Model-Based Silhouette Extraction for Accurate People Tracking

Ralf Plaenkers and Pascal Fua*

Computer Graphics Lab (LIG)
Computer Graphics Lab, EPFL, CH-1015 Lausanne, Switzerland
pascal.fua@epfl.ch

Abstract. In this work, we introduce a model-based approach to extracting the silhouette of people in motion from stereo video sequences. To this end, we extend a purely stereo-based approach to tracking people proposed in earlier work. This approach is based on an implicit surface model of the body. It lets us accurately predict the silhouette's location and, therefore, detect them more robustly. In turn these silhouettes allow us to fit the model more precisely. This allows effective motion recovery, even when people are filmed against a cluttered unknown background. This is in contrast to many recent approaches that require silhouette contours to be readily obtainable using relatively simple methods, such as background subtraction, that typically require either engineering the scene or making strong assumptions.
We demonstrate our approach's effectiveness using complex and fully three-dimensional motion sequences where the ability to combine stereo and silhouette information is key to obtaining good results.

1 Introduction

In recent years, much work has been devoted to tracking people from video sequences. Many of the techniques that have been proposed rely on extracting silhouettes and fitting body models to them. See [1,11,13] for recent reviews. While this may be practical in some cases—for example, because the background is both static and known, thus allowing background subtraction—silhouette extraction is in general a difficult task.

Here, we present a model-based approach to silhouette extraction that allows us to overcome this problem and to simultaneously recover 3–D body shape and motion as well as 2–D outlines. We use the "articulated soft objects" we proposed in earlier work [14] to represent and track human bodies: We use a conventional articulated skeleton but replace the simple geometric primitives—typically, cylinders or ellipsoids—that are usually attached to it by implicit volumetric primitives. Each one 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 three key strengths:

– **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.

* This work was supported in part by the Swiss Federal Office for Education and Science

A. Heyden et al. (Eds.): ECCV 2002, LNCS 2351, pp. 325–339, 2002.

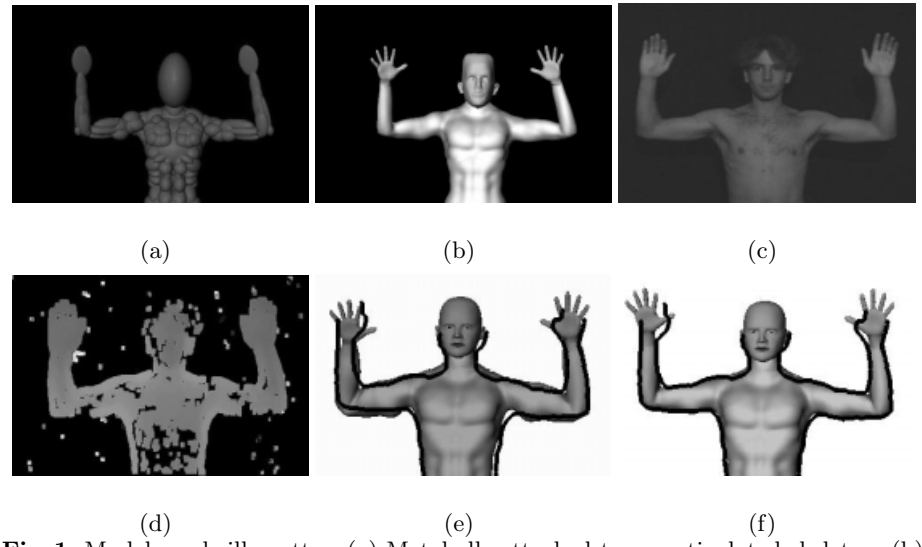


Fig. 1. Models and silhouettes. (a) Metaballs attached to an articulated skeleton. (b) Skin surface computed by ray casting. (c) One image of a stereo pair used to estimate the parameters of the model in (b). (d) Corresponding disparity map. (e) The real body outlines overlaid on the skin surface. In this case the model was fitted using stereo only. As a result, it ends up too far from the actual data points and the system compensates by incorrectly enlarging the primitives. (f) Using the silhouettes during the fitting process provides stricter constraints that yield a better result.

- **Accurate shape description by a small number of parameters:** Varying a few parameters 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.

Our model is depicted by Figure 1(a,b). In an earlier publication [14], we used it to demonstrate robust tracking using stereo only. Here, we show that its quality and 3-D nature allow us to effectively predict silhouette locations from frame to frame, constrain the search and, thus, reliably extract them from a cluttered background without having to make *a priori* assumptions about it. These silhouettes are used in turn to constrain model reconstruction, thereby further increasing its precision and reliability.

This is in contrast to many recent approaches that rely either on noisy edge or region information, such as image gradient and optical flow, e.g. [3,9], or on silhouette contours that are assumed to be readily available or obtainable through statistical background subtraction techniques e.g. [12,6,8,7], both of which require either an engineering of the scene or a static background. Furthermore, as noted in [9], the simple rigid volumetric primitive-based models that are often used tend to be a source of inaccuracy. Our approach addresses both of these issues: Our improved models yield a better accuracy that, in turn, gives us the predictive power required for effective silhouette extraction.

Combining stereo and silhouette data is valuable because they are complementary sources of information. The former works best when a body part faces the cameras but becomes unreliable when the surface slants away, which is precisely where silhouettes can be used. Figure 1(e,f) illustrates this complementarity. In this example, we used a single stereo pair. In Figure 1(e) only stereo-data, in the form of a cloud of 3–D points derived from the disparity map, was used. The stereo data is too noisy and shallow to sufficiently constrain the model. As a result, the fitting algorithm tends to move it too far away from the 3–D data and to compensate by inflating the arms to keep contact with the point cloud. Using the silhouettes in addition to the stereo data, however, sufficiently constrains the fitting problem to obtain the much improved result of Figure 1(f).

In the remainder of this paper, we first introduce both our human body model and the optimization framework we use to instantiate its degrees of freedom given stereo and silhouette data. We then turn to our model-based automated silhouette extraction approach and show that it yields precise outlines and that combining silhouettes and stereo yields much better results than using stereo alone.

2 Articulated Model and Surfaces

The human body model we use in this work [17] 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* 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. 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 by ray casting [17].

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.

2.1 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 Θ is broken into sub-vectors that control the following model components:

- Shape
  - $\Theta^w = \{\theta_b^w \mid b = 1..B\}$, the width of body parts.
  - $\Theta^l = \{\theta_b^l \mid b = 1..B\}$, the length of body parts.
- Motion
  - $\Theta^r = \{\theta_{j,f}^r \mid j = 1..J, f = 1..N\}$, the rotational degree of freedom of joint j of the articulated skeleton for all frames f
  - $\Theta^g = \{\theta_f^g \mid f = 1..N\}$, 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.

2.2 Metaballs

Metaballs [2] are generalized algebraic surfaces that are defined by a summation over n 3-dimensional Gaussian density distributions, each called a *primitive*. The final surface $\mathcal{S}$ is found where the density function F equals a threshold T, taken to be 0.5 in this work:

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

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

$$f_i(x, y, z) = exp(-2d_i(x, y, z)) \, , \tag{4}$$

where d_i represents the algebraic ellipsoidal distance described below. 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.

2.3 3–D Quadratic Distance Function

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. To express simply the transformations of these implicit surfaces that is caused by their attachment to an articulated skeleton, we write the ellipsoidal distance function d of Eq. 4 in matrix notation as follows. For a specific metaball and a state vector Θ, we define the 4×4 matrix

$$\mathbf{Q}_\Theta = \mathbf{L}_{\Theta^{w,l}} \cdot \mathbf{C}_{\Theta^{w,l}} \ . \tag{5}$$

where $\mathbf{L}$ and $\mathbf{C}$ are radii and position of the primitive respectively. The skeleton induced transformation $\mathbf{S}_\Theta$ is introduced as the rotation-translation matrix from the world frame to the frame to which the metaball is attached. These matrices will be formally defined in the appendix.
Given the $\mathbf{Q}_\Theta$ and $\mathbf{S}_\Theta$ matrices, we combine the quadric and the articulated skeleton transformations by writing the distance function of Eq. 3 as:

$$d(\mathbf{x}, \Theta) = \mathbf{x}^T \cdot \mathbf{S}_\Theta^T \cdot \mathbf{Q}_\Theta^T \cdot \mathbf{Q}_\Theta \cdot \mathbf{S}_\Theta \cdot \mathbf{x} \ . \tag{6}$$

This formulation will prove key to effectively computing the Jacobians required to implement the optimization scheme of Section 3.

We can now compute the global field function F of Eq. 3 by plugging Eq. 6 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 $\mathbf{Q}_\Theta$ and $\mathbf{S}_\Theta$ matrices, and so can its derivatives as will be shown in the appendix. These matrices will therefore constitute the basic building blocks of our optimization scheme's implementation.

3 Optimization Framework

Our goal is to instantiate the degrees of freedom of our model so that it conforms as faithfully as possible to the image data derived from motion sequences such as the ones shown in Figure 3 and Figure 4. The expected output of our system is the instantiated state vector Θ of Eq. 1 that describes the model's shape and motion. This is a highly non-linear problem: The model consists of an articulated set of implicit surfaces. As a result it contains rotations in Euclidean space as well as quadratic and exponential distance functions. Simplifying the volumetric models, replacing the perspective transform by an orthographic one, and using a different representation for rotational joints can be used to linearize parts of the problem [3]. Such approaches, however, tend to lose in generality. Therefore, we chose to use a non-linear least squares estimator (LSE) to minimize the distance between the observations and the model. We implemented a variant of the standard Levenberg-Marquart least-squares solver [15] that can handle large number of unknowns by using sparse matrices.

In practice, we use the data to write n_{obs} observation equations of the form

$$F(\mathbf{x}, \Theta) = T - \epsilon_i \ , \ 1 \leq i \leq n_{obs} \ , \tag{7}$$

where F is the global field function of Eq. 3, T is the threshold of Eq. 2, $\mathbf{x}$ is a data point, and ϵ_i is an error term. We then minimize $v^T P v$ where $v = [\epsilon_1, \ldots, \epsilon_{n_{obs}}]$ is the vector of residuals and P is a diagonal weight matrix associated to the observations. Because F is both well-defined and differentiable, these observations and their derivatives can be estimated both simply and without search using the matrix formalism of Section 2.3. This is valuable because our least-squares solver takes advantage of differential information for faster and more robust optimization, as do most powerful optimizers. The computation is outlined briefly

in the appendix and we refer the interested reader to our earlier publication [14] for additional details.

We now turn to the detailed implementation of the 3–D point and 2–D silhouette observations which are the main cues we obtain from the image sequences.

3.1 3–D Point Observations

Disparity maps such as those of Figure 1(d) are used to compute clouds of noisy 3–D points. Each one is used to produce one observation of the kind described by Eq. 7. Minimizing the corresponding residuals tends to force the fitted surface to be as close as possible to these points.

The properties of the chosen distance function allow the system to naturally deal with outliers and to converge even from rough initializations or estimates. The smooth shape of the inverted exponential that is used in our field function is responsible for both effects. It approaches zero asymptotically and, thus, provides an upper limit on the error resulting from distance between model and observation.

3.2 2–D Silhouette Observations

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

$$\mathcal{S}(\theta) = \{[x, y, z]^T \in \mathbb{R}^3, F(x, y, z, \theta) = T\} \ . \tag{8}$$

Assuming that the line of sight is tangential to $\mathcal{S}(\theta)$, let $[x(\theta), y(\theta), z(\theta)]$ be the contact point that is both on the line and on the surface. By definition, $[x(\theta), y(\theta), z(\theta)]$ satisfies two constraints:

1. The point is 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 along the line of sight to find the point $\mathbf{x}$ that is closest to the model in its current configuration. This point is used to add one of the observations described by Eq. 7. By construction, the point on the ray with the lowest value of field function F satisfies the second constraint as depicted by Figure 2(a).

In order to keep the second constraint satisfied during the optimization process, the Jacobian has to be constructed accordingly. A change in model position or size induces a motion of $\mathbf{x}$ along the ray in order to remain the closest point on the ray with respect to the model. This involves computing first and second order derivatives for the Jacobian entries as will be discussed in the appendix. We have already seen in Figure 1 that silhouettes are crucial to constrain the search space. In Figure 2(b) we show a similar behavior in a 2–D context.

4 Robust Silhouette Tracker

Because our models are fully three-dimensional, given a position estimate at a particular time, we can predict where we expect to see silhouettes in a particular

(a) (b)

Fig. 2. Silhouette constraints. (a) Two lines of sight going through the camera optical center. In both cases, at the point that minimizes F along the line, the gradient of F is perpendicular to the line. But only in the case of the dashed line that is actually tangent to the surface, is this point on the surface and thus satisfies both criteria of Section 3.2. (b) The top row depicts the results of fitting an ellipse to slightly noisy 2–D points, shown as white crosses. The computed ellipses are much too large. The bottom row depicts the fit to the same points, but under the constraint that the ellipse be tangent to the white lines. The results are much improved.

image and we use this prediction to constrain the search for the silhouettes to restricted areas in the image. In Section 4.1, we discuss this approach when run *a posteriori*, that is, by first tracking the person's body using stereo only, computing the model's outline in each frame and using it as an initial guess for the silhouette's location. This "naive" approach yields good results if the initial tracking was accurate enough but fails otherwise. In Section 4.3, we therefore introduce a more effective approach in which, for each frame, we use the model derived from the previous frame to estimate the silhouette and immediately use this silhouette to guide the recovery of the new current model position. We will show that this increases the tracker's robustness and prevents it from making mistakes in ambiguous situations.

4.1 Simple Model-Based Approach for Silhouette Extraction

Given a first fit of the model to the data, one can take advantage of it to extract additional information from the images. In this case, we can first track using stereo alone. In the absence of gross tracking errors, the projected model's outlines can be used as an initial estimate for the silhouette location. We ran a number of experiments in which we used this estimate to initialize an active contour. We found that, in practice, when the background is cluttered, the active contour does not in general converge towards the desired outline because it gets stuck into undesirable local minima. To remedy this situation, instead of running the snake on the original image, we first filtered it using the technique described in Section 4.2 and, then, used the result to deform the contour.

Figures 3 and 4 depict the results of running this algorithm on the 600 frames of two different video sequences. For most frames, deforming the predicted outline using the filtered the image resulted in an outline that was very close to the true silhouette. We saw errors in only 19 frames. Given the fact that the motions are complex and that the background is cluttered, this shows the robustness of our algorithm. Note also, that we did not use an image of the background without the subject to perform any kind of background subtraction.

Fig. 3. Extracting the subject's silhouette using the simple model-based approach of Section 4.1. Frames 20, 40, 80, 100 and 110 of the a 300-frame sequence are shown. In the upper row are the disparity maps, in the second row are the outlines of the projected model, which is shown in the third row. This outline was fitted to the filtered image gradient in the last row. The system correctly tracks until frame 100 and then recovers before frame 110. The more sophisticated approach of Section 4.3 will overcome the error in the 100th frame.

The result for frame 100 in Figure 3 shows typical problem areas. The errors around the left arm are most interesting. The model was relatively far away from the actual position. A second circular silhouette curve at the inside of the model's arm was extracted as well. It corresponds to nothing in the original image and evolves arbitrarily during snake optimization. The problem is compounded by the fact that the table in the background is very close to the subject. It is not removed totally by the low level filtering steps but it is considered as foreground instead.

A different problem occurred around the subject's right elbow. It is cut because the model projected slightly inside the real contour so that the smoothness coefficient of the snake forced it to retract from the sections of high curvature.

Fig. 4. Automatically extracted silhouette contours in a few frames of another 300-frame sequence.

Tuning the ratio of the snake parameters can delay the onset of such problems, but not prevent them altogether.

This particular example clearly demonstrates the problems that arise when trying to extract higher-level information from image data in an unsupervised environment. Using the model is a tremendous advantage but, still, it is not enough to ensure correct behavior. We address this problem in Section 4.3.

4.2 Disparity-Based Gradient Filtering

As discussed above, in the presence of a cluttered and unknown background, even a good initial body-outline estimate does not guarantee convergence of an active contour towards the correct solution. To improve the snake algorithm's performance, we filter the image to eliminate edges that are unlikely to be silhouettes but can still trap the active contour into an undesirable local minimum. To this end we combine gradient and depth information as follows.

The main input to our system is 3–D data derived from disparity maps. It is therefore natural to also use those maps to perform a form of background subtraction. This, however, is non trivial: Because of noise, absence of texture, specularities and non fronto-parallel surfaces, these maps are typically neither dense nor reliable enough to robustly separate foreground and background. Furthermore, objects located at similar distances as the subject may not be eliminated that way. This is certainly true of the disparity map of Figure 5(b).

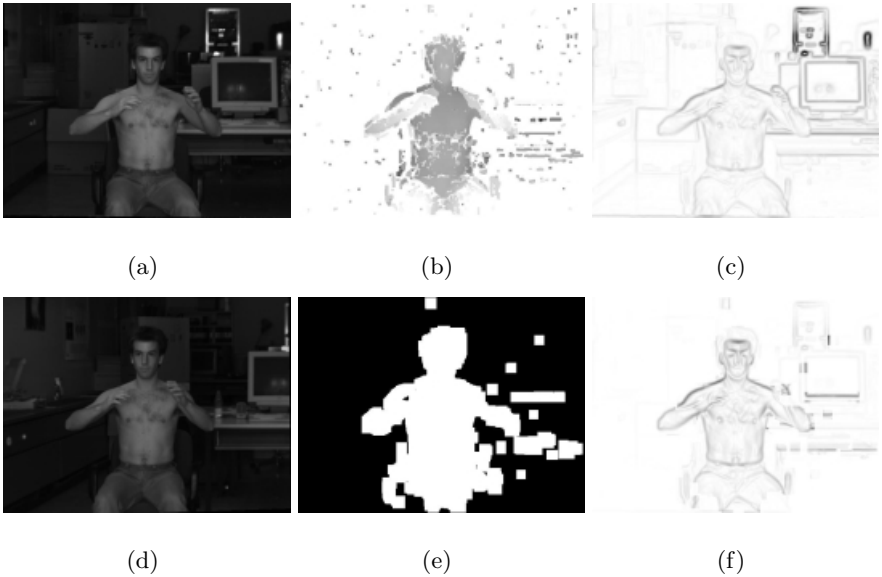


Fig. 5. Disparity-based gradient filtering. (a) and (d) is a original stereo pair and (b) is the corresponding disparity map which is binarized and morphologically "cleaned" in (e). (c) is the gradient magnitude of (a), and (f) the result of filtering (c) with (e).

We can nevertheless use such disparity maps to find image pixels with a large probability to correspond to the subject: When computing the map, we chose a range of disparities that eliminates distances that correspond to the far background. Those areas for which no acceptable disparity value can be found appear mostly in black. The correlation-based stereo algorithm we use [10] can be limited to a specified reconstruction volume and a Right-Left Check eliminates most erroneous matches. Therefore, by binarizing of the disparity map and cleaning it up using morphological operators, we obtain binary masks such as the one shown in Figure 5(e). Disparity computation was done only on a given volume excluding the far background. However, objects close to the subject are still included and parts of the subject are missing due to bad texture or shadows.

Applying the filter to the gradient image eliminates most parts of the background as shown in Figure 5(f). Finally, we perform a hysteresis thresholding that is comparable to the one used in Canny's edge-detector [4]: Only gradient image pixels are accepted that are above an upper threshold or which are above a lower threshold and have a point that passes the previous test in their immediate neighborhood.

4.3 Joint Shape and Silhouette Extraction

To solve problems such as those shown in Figure 3, we take the temporal evolution of the contour into account, instead of entirely relying on a good fit of the model. We therefore modify the scheme for silhouette extraction as follows:

1. Silhouette of previous frame serves as initialization for current frame
2. Optimize using active contours on disparity-filtered gradient image
3. Fit body model to stereo data constraint by current silhouette estimate
4. Optimize silhouette of fitted model using again active contours

The initial guess of the silhouette position is now taken to be the result of the silhouette extraction in the previous frame. Because, at frame rate, differences between successive images are small the actual silhouette is close to the previous one. Assuming a correctly recovered position of the silhouette in the previous frame we can directly feed this silhouette as initialization to an active contour algorithm for the current frame.

Again, this step alone does not ensure robust silhouette extraction. The active contour may still miss the silhouette outline of the person. We therefore rerun our fitting algorithm on the 3–D stereo data augmented by this—possibly incorrect—silhouette outline. Thanks to correct segments of the new silhouette, the implicit treatment of outliers, and the strong stereo information, in our experience the system is able to find the correct pose of the model.

In a last step the model in its correct pose is projected into the camera frame. Using the model silhouette as input to a second run of an active contour algorithm results in the final silhouette outline of the person. Results of this model-based silhouette tracker are shown in Figure 6. The original image together with the extracted silhouette of the person are shown in the left column.

The recovered model pose as well as the silhouettes are shown in the right column. These results show that this new algorithm is able to overcome the errors that occurred during the simple model-based approach of Section 4.1.

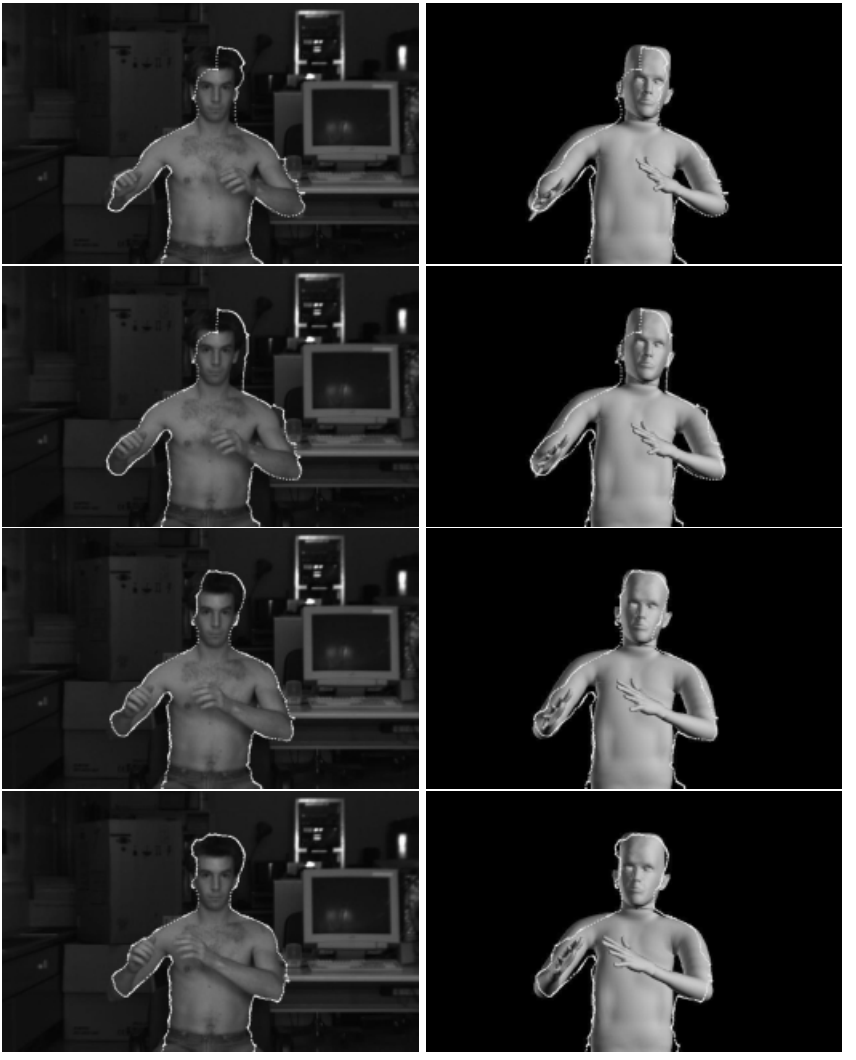


Fig. 6. Tracking results using the model-based silhouette extraction method of Section 4.3. Frames 97, 98, 99 and 100 of the sequence in Figure 3 are shown. Snake-optimized contours are overlaid on the original images on the left as well as the resulting model on the right.

5 Conclusion

We have presented a model-based technique for robustly extracting silhouette contours of people in motion. By integrating the model early on in the processing

pipeline our system overcomes the common pitfalls of low-level image processing. We do not have to engineer the background and the system could robustly extract the silhouette of the person even in a dynamic scene. 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. These silhouettes are used in turn to constrain model reconstruction, thereby further increasing its precision and reliability.

A key component of our implementation is our ability to analytically and precisely compute all required derivatives. This helps the optimizer to accurately estimate the actual variation of the distance between model and data according to parameter change and, thus, to minimize it with a minimal number of iterations. In contrast to most other work where the derivatives are obtained by perturbing the parameters and re-evaluating the distance function, computing the Jacobian analytically necessitates fewer computations. Furthermore, the derivatives in our modular matrix notation contain many identical parts and intermediate computation results can be reused to further speed-up the process.

In future work, we intend to test our system on sequences with highly cluttered and dynamic background. Our results are promising but more tests have to be effected to analyze the limits of the system. Currently, no provisions for occlusion and limb self-occlusion are given. The depth information from the disparity maps was sufficient to successfully track in our test sequences but explicit occlusion detection would be needed to robustly track more complex motions.

Appendix: Differentiating the Global Field Function

To illustrate the efficient computation of the Jacobian entries for the least squares estimator we present the computation of the metaball distance function as well as its first and second order derivatives with respect to a rotational joint paramter.

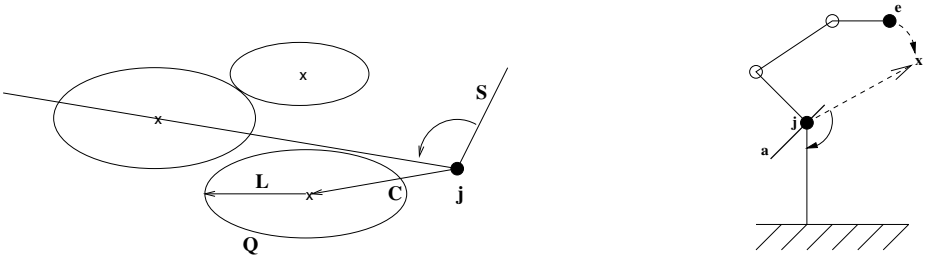
To express simply the transformations of the implicit surfaces caused by their attachment to an articulated skeleton, we write the ellipsoidal distance function d of Eq. 3 in matrix notation as follows. This formulation will prove key to effectively computing the Jacobians required to implement the optimization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

Recall that the distance function of Equation 3 can be written as follows:

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

$d(\mathbf{x}, \Theta)$ defines an ellipsoidal quadratic distance field. For a given primitive and the state vector Θ, the 4×4 matrix $\mathbf{Q}_\Theta$ contains the location of the center $\mathbf{C}$ and the scaling $\mathbf{L}$ along the principal axes respectively. Figure A.1(a) illustrates the concept.
$\mathbf{LC}_{\Theta^{w,l}} = \mathbf{L}_{\Theta^{w,l}} \cdot \mathbf{C}_{\Theta^{w,l}}$ is the scaling and translation along the principal axes:

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



(a) (b)

Fig. A.1. (a) illustrates the definition of quadric **Q**. The hierarchical skeleton transformation **S** up to joint **j** is followed by the centroid transformation **C** and by **L** describing the quadric's volume. (b) depicts the articulated structure **S**. The distance between model surface **e** and observation **x** is to be minimized subject to rotation θ^r around axis **a** of joint **j**.

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 $\theta^r \in \Theta$ of a joint J, we write:

$$\mathbf{S}_\Theta = \mathbf{E} \cdot \mathbf{J}_{\theta^r} = \mathbf{E} \cdot \mathbf{R}_{\theta^r} \cdot \mathbf{J}_{0^r} \ , \tag{A.1}$$

where $\mathbf{E}$ is the homogenous 4×4 transformation from the joint frame to the metaball frame. $\mathbf{J}_{\theta^r}$ is the transform from world frame to joint frame, including the rotation parameterized by θ^r and $\mathbf{R}_{\theta^r}$ is the homogenous rotation matrix of θ^r around axis **a** with $\mathbf{J}_{0^r} = \mathbf{R}_{\theta^r}^{-1} \cdot \mathbf{J}_{\theta^r}$. A configuration of an articulated structure is depicted by Figure A.1(b).

We can now compute the global density function F of Equation 3 by plugging Equation 6 into the individual field functions of Equation 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.

A.1 Derivatives of the Distance Function

The generic derivative of the distance function wrt. to any parameter $\theta \in \Theta$ from the state vector can be computed as:

$$\frac{\partial}{\partial\theta} d(\mathbf{x}, \Theta) = 2 * \mathbf{x}^T \cdot \mathbf{S}_\Theta^T \cdot \mathbf{Q}_\Theta^T \cdot \left[\frac{\partial}{\partial\theta} \mathbf{Q}_\Theta \cdot \mathbf{S}_\Theta \right] \cdot \mathbf{x} \tag{A.2}$$

The computation includes parts of the distance function and intermediate results can be cached and reused.

The model pose is exclusively defined by the state of the underlying articulated structure $\mathbf{S}$ of Eq. A.1. A rotational parameter θ^r defines the angle between two body parts. For differentiation classical robotics methods can be applied, such as [5]. Combined with the quadric the rotational derivative of Eq. A.2 can be shown to be:

$$\left[\frac{\partial}{\partial\theta^r}\mathbf{Q}_\Theta \cdot \mathbf{S}_\Theta\right] \cdot \mathbf{x} = \mathbf{Q} \cdot \mathbf{E} \cdot \mathbf{a} \times \mathbf{jx} \ . \tag{A.3}$$

with rotational axis $\mathbf{a}$ and $\mathbf{jx} = \mathbf{J}_\theta \cdot \mathbf{x}$ being the vector from joint center to observation. $\mathbf{E}$ is the transformation from the joint frame to the metaball frame. See Figure A.1(b) for an illustration. Equation A.3 can be efficiently implemented because it only consists of a simple vector cross-product transformed into the quadric's frame. For more details we refer the interested reader to our earlier publication [14].

A.2 Silhouette Constraint

As introduced in Section 3.2 the motion of the silhouette point $\mathbf{x}$ along the ray has to be taken into account during optimization. For non-articulated implicit surfaces this has been shown by [16]. This involves computing first and second order derivatives for the Jacobian entries. This turns out to be prohibitively expensive when done in a brute force manner. In contrast, our modular matrix formulation allows an elegant description of these derivatives because they retain their modularity and can be computed similarily to the first order derivatives. Again, intermediate results of the function evaluation as well as the computation of the first order derivatives can be reused to speed up the process.

Second order derivatives with respect to spatial coordinates can be computed according to Eq. A.4 and those with respect to a spatial coordinate as well as a parameter of the state vector according to Eq. A.5. Please see our earlier publication [14] for a full derivation of the Jacobian for the field functions.

$$\frac{\partial^2 d}{\partial x_i \partial x_j} = 2 * \frac{\partial \mathbf{x}}{\partial x_i}^T \cdot \mathbf{S}_\Theta^T \cdot \mathbf{Q}_\Theta^T \cdot \mathbf{Q}_\Theta \cdot \mathbf{S}_\Theta \cdot \frac{\partial \mathbf{x}}{\partial x_j} \ , \tag{A.4}$$

$$\frac{\partial^2 d}{\partial x_i \partial \theta} = 2 * \mathbf{x}^T \cdot \left(\left[\frac{\partial}{\partial\theta}\mathbf{S}_\Theta^T \cdot \mathbf{Q}_\Theta^T\right] \cdot \mathbf{Q}_\Theta \cdot \mathbf{S}_\Theta + \mathbf{S}_\Theta^T \cdot \mathbf{Q}_\Theta^T \cdot \left[\frac{\partial}{\partial\theta}\mathbf{Q}_\Theta \cdot \mathbf{S}_\Theta\right]\right) \cdot \frac{\partial \mathbf{x}}{\partial x_i} \tag{A.5}$$

These derivatives are necessary to correctly integrate 2–D silhouette observations in a 3–D optimization framework. When replacing them with simpler, first order ones we experienced incorrect estimation of the closest point on the ray and the optimizer did not converge. Using the complete derivatives allows the optimizer to more precisely estimate the system's reaction to paramter change and, thus, to find the optimal state with a minimal number of iterations.

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. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6), 1986.
5. J.J. Craig. *Introduction to robotics: mechanics and control*, chapter 5. Electrical and Computer Engineering. Addison-Wesley, 2nd edition, 1989.
6. L. Davis, E. Borovikov, R. Cutler, D. Harwood, and T. Horprasert. Multi-perspective analysis of human action. In *Third International Workshop on Co-operative Distributed Vision*, November 1999.
7. Q. Delamarre and O. Faugeras. 3D Articulated Models and Multi-View Tracking with Silhouettes. In *International Conference on Computer Vision*, Corfu, Greece, September 1999.
8. J. Deutscher, A. Blake, and I. Reid. Articulated Body Motion Capture by Annealed Particle Filtering. In *CVPR*, Hilton Head Island, SC, 2000.
9. T. Drummond and R. Cipolla. Real-time tracking of highly articulated structures in the presence of noisy measurements. In *International Conference on Computer Vision*, Vancouver, Canada, July 2001.
10. P. Fua. From Multiple Stereo Views to Multiple 3–D Surfaces. *International Journal of Computer Vision*, 24(1):19–35, August 1997.
11. D.M. Gavrila. The Visual Analysis of Human Movement: A Survey. *Computer Vision and Image Understanding*, 73(1), January 1999.
12. I. Kakadiaris and D. Metaxas. 3D Human Body Model Acquisition from Multiple Views. In *International Conference on Computer Vision*, 1995.
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. Articulated Soft Objects for Video-based Body Modeling. In *International Conference on Computer Vision*, pages 394–401, Vancouver, Canada, July 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. 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.
17. 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.