EUROGRAPHICS 2008 / G. Drettakis and R. Scopigno
(Guest Editors)

Volume 27 (2008), Number 2

Automatic Conversion of Mesh Animations into Skeleton-based Animations

Edilson de Aguiar[1†], Christian Theobalt[2], Sebastian Thrun[2], Hans-Peter Seidel[1]

[1]MPI Informatik, Saarbruecken, Germany
[2]Stanford University, Stanford, USA

Abstract

Recently, it has become increasingly popular to represent animations not by means of a classical skeleton-based model, but in the form of deforming mesh sequences. The reason for this new trend is that novel mesh deformation methods as well as new surface based scene capture techniques offer a great level of flexibility during animation creation. Unfortunately, the resulting scene representation is less compact than skeletal ones and there is not yet a rich toolbox available which enables easy post-processing and modification of mesh animations. To bridge this gap between the mesh-based and the skeletal paradigm, we propose a new method that automatically extracts a plausible kinematic skeleton, skeletal motion parameters, as well as surface skinning weights from arbitrary mesh animations. By this means, deforming mesh sequences can be fully-automatically transformed into fully-rigged virtual subjects. The original input can then be quickly rendered based on the new compact bone and skin representation, and it can be easily modified using the full repertoire of already existing animation tools.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism:Animation I.4.8 [Image Processing and Computer Vision]: Scene Analysis:Motion

1. Introduction

Recently, a variety of deformation-based approaches have been proposed that enable skeleton-less generation of computer animations [BS08]. This change of paradigm in animation production has been flanked by the appearance of new tracking approaches in computer vision that enable optical motion capture of arbitrary subjects using deformable meshes rather than rigid body hierarchies [dATSS07a]. Both of the above developments suggest that purely mesh-based animation representations have developed into an ever more popular alternative to kinematic hierarchy-based animations.

Unfortunately, although mesh-based approaches provide greater flexibility at the time of animation creation than skeleton-based algorithms, they output more space-consuming dynamic scene representations comprising of independent position data streams for every vertex. A further disadvantage is that animators are used to a large repertoire of tools for editing and rendering traditional skeletal animations, but yet lack the same set of tools for working with mesh-based dynamic scene representations.

Xu et al. [XZY*07] propose to close this gap by introducing a set of mesh-based operations to post-process surface animations in a similar manner as kinematic representations. Although their method allows for flexible post-processing of time-varying surfaces, it requires a fundamental redesign of existing animation tools and also does not explore data compaction possibilities. The latter was the focus of the work by James et al. [JT05] who aim at extracting a skinning representation from mesh sequences that is well-suited for rendering on graphics hardware but not meant to be used for editing.

In contrast, we propose a method that enables the fully-automatic conversion of an arbitrary mesh animation into a skeleton-based animation. Given as input a deforming mesh with constant surface connectivity, our algorithm first extracts a plausible kinematic bone hierarchy that closely resembles a skeleton hand-crafted by an animator, Sect. 4 and

† This work was done while the author was a visiting researcher at the Stanford University and it was sponsored by the Max-Planck Center for Visual Computing and Communication

Figure 1: *From left to right: Input animation, color-coded distribution of blending weights, and two poses of the input re-generated based on our skeleton-based version.*

Sect. 5. Thereafter, our algorithm automatically infers joint motion parameters, Sect. 6, and estimates appropriate surface skinning weights, Sect. 7, to attach the skeleton to the surface. The output of our algorithm is a fully-rigged skeletal version of the original surface-based input. We show results obtained with a variety of mesh animations, and also prove the faithfulness of the reconstructed skeletal representations to ground truth input, Sect. 8. In summary, our algorithm

- enables fully-automatic extraction of skeleton structure, skeletal motion parameters and surface skinning weights from arbitrary deforming mesh sequences, and
- thereby enables easy post-processing and fast rendering of mesh animations with standard skeleton-based tools without having to modify them.

As opposed to related methods our approach *jointly* produces a compact and easily modifiable skeletal version (Fig. 1), enables fast and accurate rendering of the original input, enables easy generation of new pose sequences for the input subject, and achieves all this without requiring any modification to already existing animation tools.

2. Related Work

In our work we jointly solve a variety of algorithmic subproblems by extending ideas from the research on mesh segmentation, skeleton reconstruction, motion estimation, character skinning and pose editing. For the sake of brevity, we refer the interested reader to overview articles on motion estimation [Pop07] and mesh segmentation methods [Sha08], and in the following highlight selected related papers from the other categories.

2.1. Skeleton Reconstruction

Different ways for performing skeleton extraction, each of them tailored to a specific data type and application, have been proposed in the literature. Some approaches extract skeletons from static meshes [KT03, LKA06, SLSK07] to gain information on topology or to perform segmentation. Thus, extraction of an animation skeleton is not the goal.

The accurate extraction of kinematically and biologically plausible animation skeletons is of great importance in optical motion capture, where the skeletal structure needs to be inferred from marker trajectories [KOF05, dATS06] or shape-from-silhouette volumes [dATM*04]. Similarly, kinematic skeletons can be reconstructed from a set of range scans of humans [AKP*04] or CT scans of the human hand [KM04]. Recently, Aujay et al. [AHLD07] and Schaefer et al. [SY07] have proposed methods to extract plausible human animation skeletons from a static surface mesh of a character, for instance by using prior knowledge about the anatomy of humans and animals.

In contrast, in our setting we explicitly make use of motion information to extract kinematic hierarchies more robustly. Our algorithm creates a plausible animation skeleton that best explains the data and that closely resembles a skeleton designed by an animator (in case such a skeleton exists for the data set).

2.2. Character Skinning

Skinning or enveloping is the problem of mapping the articulated motion of the skeleton to deformations of the character's surface. Due to its efficiency, ease of implementation and support in many commercial packages, the most widely used enveloping method is linear blend skinning [LCF00]. Unfortunately, standard linear blending has a limited modeling power and cannot reliably reproduce certain effects, such as muscle bulging. More recent blending weight estimation schemes look at several example poses and suggest methods to overcome the limitations inherent to linear blending. Wang et al. [WP02] suggest the use of multi-linear blending weights, Mohr et al. [MG03] suggest selective adding of bones to increase reconstruction faithfulness, and James et al. [JT05] suggest to use affine transformations.

In a different line of thinking, researchers recently suggested to use the motion of a kinematic skeleton to express constraints for a surface deformation method like a Laplacian deformation [YBS07] or a Poisson deformation [WPP07]. By this means, convincing animations can be obtained as one can capitalize on the full modeling flexibility of a more complex mesh deformation approach.

Recently, Baran et al. [BP07] have proposed a new approach that bears some similarity with our method. They jointly fit a template skeleton to a static character mesh and automatically compute appropriate skinning weights. In contrast to their method, our approach *extracts* a subject-specific

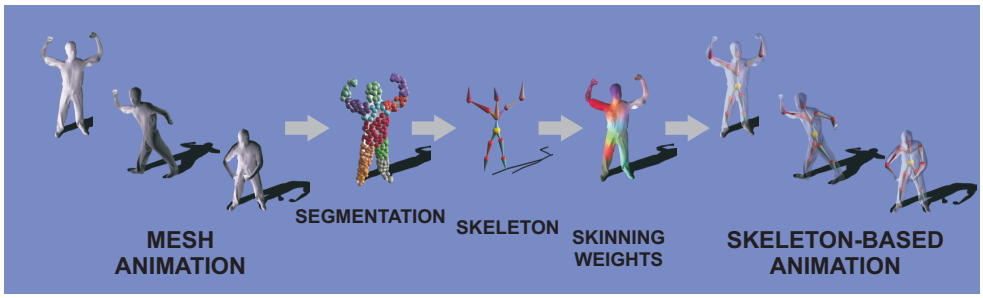

Figure 2: *Overview of our algorithm: using an animated mesh as input, our approach segments the model into plausible approximately rigid surface patches (shown in different colors), estimates the kinematic skeleton (joints shown in red) and its motion parameters, and calculates the skinning weights connecting the skeleton to the mesh. The output is a skeleton-based version of the input mesh animation.*

skeleton without knowing the structure a priori, and it does so by analyzing the input motion of the mesh. Still, we capitalize on their blending weight computation, but in contrast to their original idea, apply it to a whole series of poses in order to obtain weights that more faithfully approximate the input.

2.3. Pose and Animation Editing

Similar in spirit to our algorithm are methods that edit mesh-based deformations directly based on spatio-temporal mesh editing operators rather than by transforming the input into a skeletal representation [XZY*07, KG06]. While the flexibility of these methods is very high and the resulting edited animations are of high visual quality, these methods require a fundamental redesign of existing animation tools and they don't explore data compaction possibilities. However, in particular when the mesh animation can well be explained by a skeleton, our approach to transform a mesh animation into a skeletal one is advantageous, as it enables fast and easy post-processing using the full spectrum of already existing software.

A first step in this direction was taken in [SY07] where a skeleton and skinning weights are estimated from a set of example poses. The main differences to our method are that we exploit the full motion information in the input to robustly learn a skeleton by means of spectral clustering, that we get a full range of skeletal motion parameters for the input animation which gives us much greater flexibility during post-processing, and that we fit our skinning weights to the entire range of animation frames which leads to more reliable estimates.

In summary, while many related approaches from the literature propose alternative methods to solve subproblems of our setting, our method outputs a fully-parameterized skeletal version of the input mesh animation that can either be rapidly rendered without modification or easily be modified using standard tools.

3. Overview

An overview of our approach is shown in Fig. 2. The input to our algorithm is an animated mesh sequence comprising of N frames. We represent an animated mesh sequence by a mesh model $M = (V = vertices, T = triangulation)$ and position data $p_t(v_j) = (x_j, y_j, z_j)_t$ for each vertex $v_j \in V$ at all time steps t. By using the coordinate sets $P_t = \{p_t(v_j)\}$ and the mesh M we are able to completely describe our time-varying animated model.

In the first step of our algorithm we employ spectral clustering to group seed vertices on the mesh into approximately rigid segments. By using the clustered seed vertices we are then able to segment the moving mesh into kinematically meaningful approximately rigid patches, Sect. 4. Thereafter, adjacent body parts are determined and the topology of the kinematic structure of the mesh is found, Sect. 5. Using the estimated topology, joint positions between interconnecting segments are calculated over time. In order to eliminate temporal bone length variations due to per-time step fitting inaccuracies, joint positions are updated at all time steps and an inverse kinematics approach is applied to determine the subject's joint parameters over time, Sect. 6. In a last step we calculate appropriate skinning weights to attach the learned skeleton to the surface, Sect. 7. This way, we produce a complete skeleton-based new version of the original input.

4. Motion-driven Segmentation

The first step of our algorithm segments the animated input mesh (given by M and P_t) into spatially coherent patches that undergo approximately the same rigid transformations over time. We initialize our approach by selecting a subset of l vertices that are distributed evenly over the mesh M. For the selection of the seeds we only consider a reference pose P_{tr} (typically $tr = 0$), and employ a curvature-based segmentation method [YGZS05] to decompose the model into l surface patches. The seed vertices are chosen as the vertices closest to the centers of the patches. We typically choose l to be in the range of $0.3 - 1.0\%$ of the total vertex count of

the model, which enables reasonably fast decomposition of even large meshes.

Similar to [KOF05, dATS06] in the context of optical motion capture and [TRdAS07] for volumetric segmentation, the motion trajectories of the seed vertices throughout the whole sequence form the input to a spectral clustering approach [NJW02] which automatically groups the l seeds into k approximately rigidly moving groups. We capitalize on the invariant that mutual distances between points on the same rigid part should only exhibit a small variance while the mesh is moving.

In order to use spectral clustering we first construct a spatial affinity matrix A. We developed an affinity criterion specifically for our setting that defines the entries of A as follows:

$$A_{i,j} = e^{-\frac{\sigma_{i,j}+\sqrt{\rho_{i,j}}}{S^2}}, \qquad (1)$$

where $\rho_{i,j} = \frac{1}{N^2}\sum_t \delta(v_i, v_j, t)$ is the mutual Euclidean distance $\delta_{i,j,t}$ between seed vertex v_i and seed vertex v_j over time and $\sigma_{i,j}$ is its standard deviation. $S = \frac{1}{N^2}\sum_{i,j}(\sigma_{i,j} + \sqrt{\rho_{i,j}})$ is a scaling term controlling the convergence behavior. We construct the entries of A such that the affinity values of vertex pairs with large average mutual distance is reduced, which forces our spectral clustering algorithm to put spatially far apart groups of vertices with similar motion into separate clusters.

Spectral clustering is our method of choice as it can robustly infer complex cluster topologies as they are typical for our motion segmentation problem. Instead of grouping the vertices directly based on the individual values $A_{i,j}$, spectral clustering uses the top eigenvectors of matrices derived from A to cluster the vertices (for details see [NJW02]). This makes the clustering more robust against outliers and leads to a more robust and kinematically more meaningful segmentation. As an additional benefit, the optimal number of clusters k can be automatically calculated based on the data set's eigen-gap. In our system, we automatically cluster the seeds into k groups such that around 99.0% of the total variation of the data is explained.

Using the k optimal vertex clusters, we create triangle clusters $T_0 \ldots T_{k-1} = T$ by assigning each triangle $\Delta = (w_0, w_1, w_2) \in T$ to the closest seed vertex class considering the average Euclidean distance from a seed vertex v_u to w_0, w_1, and w_2. The resulting clusters divide the mesh into k approximately rigid surface patches, Fig 3. Note that although a structurally motivated distance measure like the geodesic distance could also be used for clustering the triangles and to determine affinities in Eq. (1), our experiments show that similar results can be achieved using simple Euclidean distance which reduces the algorithm's computation time considerably.

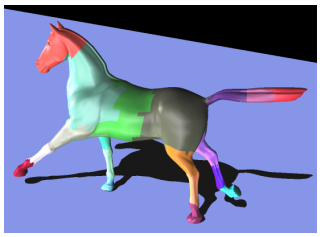

Figure 3: *Approximately rigid surface patches shown in different colors.*

5. Automatic Skeleton Extraction

Given the list of body segments, their associated seed vertices and triangle patches, we extract the kinematic skeleton structure of the animated mesh by first finding its kinematic topology (i.e. find which body parts are adjacent) and, thereafter, by estimating the positions of the interconnecting joints for the whole sequence.

To determine which body segments are adjacent, we analyze the triangles at the boundaries of the triangle patches. Body parts A and B are adjacent if they have mutually adjacent triangles in their respective patch boundaries. Unfortunately, in practice a patch may be adjacent to more than one other patch. If more than two patches are directly connected (e.g. head, torso and arm), we need to decide which segments are truly kinematically connected and which are not. Here we take a heuristic approach and consider only those patches to be adjacent that share the longest common boundary (in terms of the number of adjacent boundary triangles). For instance, if head, arm and torso are connected we calculate the number of neighboring triangles for all combinations of patch pairings (e.g. head-torso, head-arm and torso-arm) and do not assign the pair head-arm as an adjacent segment since it has less neighbors in comparison with the other two options. For any adjacent pair of patches, a joint has to be found later. Note that in our system we assume that the body part in the center of gravity of the mesh at the reference time step is the root of the hierarchy.

In order to estimate the joint positions between two adjacent body segments A and B quickly, we only consider the information from the sets of seed vertices V_A and V_B located on these segments, and not the information from all vertices of V. Instead of solving for the complete sequence of joint positions, we significantly reduce the problem's complexity by first aligning the segment poses to a reference time step tr (usually $tr = 0$), then solving for a single optimal joint position at c_{tr} in the reference pose, and finally retransforming c_{tr} into the original poses of A and B. To serve this purpose, for each time step t we first compute two rigid body transforms $T_{A_{t\to tr}}$ and $T_{B_{t\to tr}}$ that align the positions of the seed vertices in both sets with the positions of the seed vertices V_A at the reference time step [Hor87].

For finding c_{tr} we follow an idea proposed in [KOF05] and assume that a good estimate for the correct sequence of

joint positions is the sequence of locations that minimizes the variance in joint-to-vertex distance for all seed vertices of the adjacent parts at all frames. Using this assumption, [dATS06] solves for the joint location at the reference time c_{tr} by using a distance penalty based on the average Euclidean distance to regularize the solution. Alternatively, we use the regularization term proposed by [AKP*04], which makes the estimated joint position come closer to the centroid position b_t of the boundary curve between the two adjacent body parts at all time steps t. Therefore, we solve for c_{tr} by minimizing:

$$J(c_{tr}) = \frac{1}{2} * \sum_{v_a \in V_A} \sigma_a(c_{tr}) + \frac{1}{2} * \sum_{v_b \in V_B} \sigma_b(c_{tr}) + \alpha * d(c_{tr}, b_{tr}) \tag{2}$$

where $\sigma_a(c_{tr})$ and $\sigma_b(c_{tr})$ corresponds to the Euclidean distance variance over time between the joint position c_{tr} and the vertex v_a and between c_{tr} and v_b, respectively. $d(c_{tr}, b_{tr})$ is the Euclidean distance between c_{tr} and b_{tr} at the reference time step. The coefficient α is used to regularize the solution, making the joint position be located as closed as possible to the interior of the mesh. The results in Sect. 8 were generated using a value of $\alpha = 0.05$ (which we found to be satisfactory in our experiments).

After solving Eq. 2 and finding the optimal joint location c_{tr}, the joint positions at all other frames can be easily computed by $c_t = T^{-1}_{A_{t \to tr}} * c_{tr}$. By applying the above procedure to all adjacent body parts we reconstruct all joint positions for the whole sequence (see Fig. 4).

6. Motion Parameters Estimation

A consistent parameterization of the skeletal motion in terms of joint parameters is only feasible in case the skeleton structure has constant dimensions over time. However, due to possible errors generated by aligning the body parts in the reference frame (mostly caused by subtle (non-rigid) relative motion between vertices on the same segment) the lengths of the bones in the skeleton may slightly vary over time. We enforce the bone lengths to be constant by stepping through the hierarchy of the estimated skeleton from the root down to the leaves and correcting the joint positions for each pair of subsequent joints in the kinematic chain separately.

Let c_t^i be the position of a joint i and c_t^{i-1} the position of its parent joint at time t. We are able to calculate the optimal value for the length of the bone connecting joint $i-1$ and i, $l_{i-1,i}$, over time and the new positions for the joints i, nc_t^i, by minimizing the following energy:

$$S(nc^i, l_{i-1,i}) = \sum_{t=0}^{N} \|c_t^i - nc_t^i\|^2 + (\|nc_t^i - c_t^{i-1}\| - l_{i-1,i})^2 . \tag{3}$$

The first term in Eq. (3) keeps the new joint position nc_t^i as close as possible to the old position, while the second term constrains the bone length to be the same in all frames.

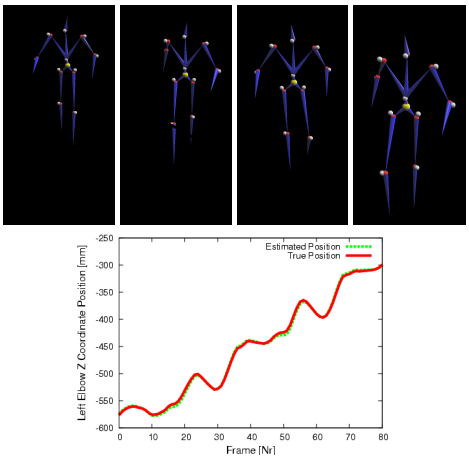

Figure 4: *(upper) Visual comparison between the reconstructed joint positions and the true positions (shown as white spheres) at four different frames. (lower) Plot of the low difference in z-coordinate between the reconstructed left elbow joint position and the ground truth position over time (biped walking sequence).*

After solving this equation for each pair of subsequent joints in the hierarchy we obtain a consistent kinematic structure of the mesh M. To infer joint motion parameters, i.e. a rotational transformation T_t^i for all joints i at all times t, we first specify the skeletal pose at the first time step as reference pose (this is the best we can do given no explicit reference). Thereafter, we apply a Cyclic-Coordinate-Descent (CCD) like algorithm [Lue73,BMB87] to infer all T_t^i relative to the reference using Euler angle parameterization. To this end, we optimize one joint variable at a time by calculating the positional error w.r.t the estimated joint positions found for that time step. Since we have all in-between joint positions of each kinematic sub-chain, our method converges quickly and reliably. Finally, the translation of the root is stored as additional parameter for each frame.

7. Skinning Weight Computation

Skinning is the process of attaching the skeleton to the surface in such a way that changes in skeletal pose lead to plausible surface deformations. Although more advanced deformation schemes exist, Sect. 2.2, we decided to use the standard linear blend skinning method [LCF00] (also called skeletal subspace deformation method - SSD) since it is widely supported in games and animation packages.

Let $p_0(v_i)$ be the position of the vertex v_i of M in the reference pose (or rest pose), let T_t^b be the transformation of the bone b from the reference to time t, and let $w^b(v_i)$ be the weight of the bone b for the vertex v_i. Note that the bone transformation T_t^b equals the transformation T_t^j of the preceding joint j from the hierarchy. SSD expresses the new position of vertex v_i at time t as $p_t(v_i) = \sum_b (w_i^b T_t^b p_0(v_i))$.

Therefore, in order to use the SSD method to re-animate the sequences using a more compact representation, we need to determine the bone weights w_i^b for each vertex v_i, i.e. we need to know how much each bone influences each vertex.

Although alternative approaches for skinning weight computation are available in the literature, we employ the method proposed in [BP07] to determine the skinning weight distribution for each bone. This method computes the distribution based on the results of a heat diffusion process rather than based on simple vertex proximity to the bone, which makes the estimation process more robust. In contrast to their work, however, we consider the entire sequence of mesh and skeleton poses from the input when finding optimal weights. In particular we first solve for the weight distributions w_f^b of each frame f separately, and thereafter average them to obtain the final distributions w^b.

When computing the weight distribution w_f^b we regard the character volume as an insulated heat-conducting body and force the temperature of the bone b to be 1 and the temperature of all others to be 0. The weight $w_f^b(v_i)$ equals the equilibrium temperature at v_i. For computational simplicity, the equilibrium equation is only solved on the mesh's surface yielding $\frac{\partial w_f^b}{\partial t} = \Delta w_f^b + H_f(p_f^b - w_f^b) = 0$. In our discrete case this can be reformulated as

$$(-\Delta_f + H_f) w_f^b = H_f p_f^b . \quad (4)$$

In this equation, Δ_f is the discrete Laplacian operator at frame f [WMKG07], p_f^b is a vector where $p_f^b(v_i)$ equals 1 in case b is the nearest bone to vertex v_i and 0 otherwise. H_f is a diagonal matrix with entries $H_{ii} = 1/dist(v_i)^2$ representing the heat contribution weight of the nearest bone to vertex v_i at frame f. Here, $dist(v_i)$ is the Euclidean distance between vertex v_i and the nearest bone in case it is contained in the character's volume and 0 otherwise. The final weight distributions w^b for each bone is the average of the weights w_f^b for all frames.

The heat diffusion solution provides smooth and realistic blending weight distributions since it respects geodesic surface proximity during weight assignment rather than error-prone Euclidean proximity [BP07]. Furthermore, our experiments show that by computing the optimal weights from all available poses our skeletal animation more faithfully reproduces the entire original mesh animation.

8. Experiments and Results

To demonstrate and validate our algorithms, we applied it to a set of mesh animations generated in a variety of different ways, see Tab. 1 for a list. Some synthetic sequences, such as the horse sequence, Fig. 1, were generated with a mesh deformation method [SP04]. Other synthetic sequences like the bird and the walking biped (see **video**) were originally created with a skeleton-based method which conveniently gives us ground truth data. We also have captured performances of humans at our disposition. The dancing and capoeira sequences, Fig. 6 and Fig. 8, were reconstructed by a mesh-based marker-less motion capture approach [dATSS07a]. The cartwheel and posing sequences (Fig. 8 and video) were obtained by using raw motion capture marker trajectories as constraints in a deformation approach [dATSS07b].

Figure 5: *Visual comparison between pre-defined skeleton embedded using [BP07] (l) and the skeleton extracted by our approach (r). Despite slight differences in the torso area, our estimated skeleton closely resembles the fitted template.*

Fig. 8 shows one original input mesh, the color-coded skinning weight distributions, and some poses of the original animation re-rendered using our skeleton-based reparametrization. A visual comparison between our result and the input, see Fig. 6 and several examples in the accompanying video, shows that our result is visually almost indistinguishable from the original mesh animation and exhibits very natural surface deformations. Furthermore, visual inspection already suggests that the estimated kinematic structures are nicely embedded into the subjects at all frames and possess biologically plausible dimensions and hierarchies. Here, we would like to note again that all these results were obtained fully-automatically. Our new representation is very compact. For the horse animation, for instance, we only need to store geometry and skinning weights once, and for each time step only store 60 motion parameters (3-DOF per joint) rather than approx. 25000 coordinate values.

To get a quantitative impression of the faithfulness and quality of our results, we analyze individual aspects of our method more closely.

Skeleton Reconstruction Since for most synthetic data we know the true sequence of joint positions, we are able to provide a quantitative estimate of the accuracy of our skeleton extraction and motion capture approach. Fig. 4(upper) shows a visual comparison between the joint positions estimated by our approach and the true joint positions, shown as white spheres, for the walking biped sequence. Fig. 4(lower) illustrates the accuracy of the reconstructed motion parameters over time. The plot shows a comparison between the z-coordinate of the true and estimated positions of the left elbow joint for the same walking sequence. The difference between true and estimated joint positions is very small and consistent which illustrates the robustness of our method. Similar low errors could be observed in all our sequences

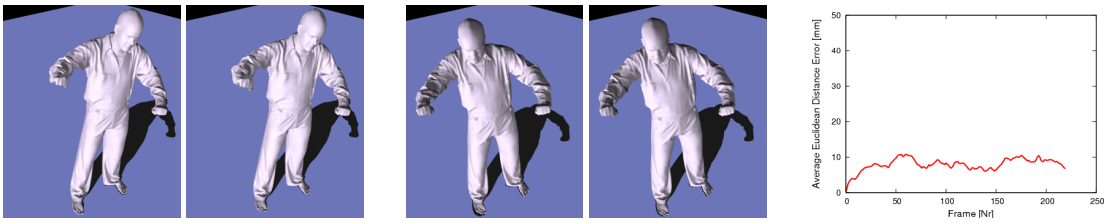

Figure 6: *(left) Two pairs of images comparing the input (left sub-image) to our skeleton-based result (right sub-image). In either case the renderings are visually almost indistinguishable. (right) Plot showing the low average Euclidean distance error between input and reconstructed vertex positions for the human-sized model.*

with available ground truth. The column JACC in Tab. 1 shows the average Euclidean distance error between estimated joint position and true joint position for all joints over time. Due to the lack of absolute coordinates, the error is given in percent of the longest bounding box (B.B.) dimension of the model. In all our examples, major misalignments between the original and the reconstructed skeleton could only be found if the part of the input corresponding to that joint was not prominently articulated.

Our automatically computed bone hierarchies closely match the skeletons that would typically be created by animators. This is shown in the comparison in Fig. 5 where we show side-by-side our result and the result of fitting a template kinematic model with the method of Baran et al. [BP07]. Only the topology of the root/spine area slightly differs.

Accuracy of reparameterized animation Fig. 6(left) shows a comparison between two frames of the input dancing sequence (left sub-images) and the generated skeleton-based animation (right sub-images). Visually, almost no difference can be seen. The faithful reproduction of the original input is also shown in the accompanying video.

Fig. 6(right) plots the consistently low average difference between the vertex positions of the input and the reparameterized output over time for the dancing sequence. For the human-sized figure, the error is mostly below one centimeter which shows the high reconstruction quality also quantitatively. Column ACCU of Tab. 1 shows that similarly low errors are observed in the other test sequences.

Pose and animation editing Using our system, we are able not only to recreate the original input based on a more compact representation, but can straightforwardly produce novel postures of the input mesh, see Fig. 7. To this end, we only need to modify the joint parameters which can easily be done in any standard animation package. Since we have a complete set of motion parameters for the input, we can also easily modify aspects of the original animation by altering the joint parameter curves of selected joints (see accompanying video).

Computation time Tab. 1 lists the run times of each processing step in our algorithm. The second and third columns

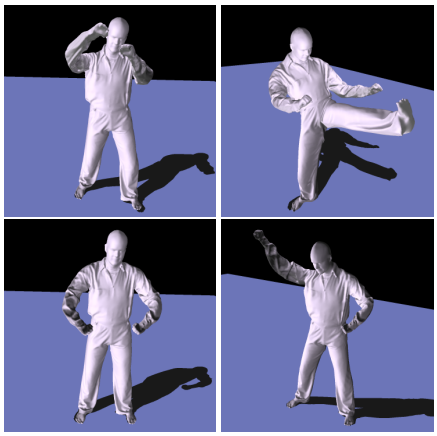

Figure 7: *New poses for the input mesh can be easily generated by simply changing the joint parameters of the extracted skeleton.*

show the number of frames in each sequence (FRAMES) and the number of triangles in each model (NUMTRI). The column SEGM lists the time needed for clustering and mesh segmentation. Column SKEL lists the time needed to build the skeleton and estimate all motion parameters, and column SKIN lists the time needed to find the blending weights. With the exception of SKIN which shows per-frame times, all times given are for processing entire sequences. All run times were measured on a Laptop featuring an Intel Core Duo CPU with 1.7 GHz.

Discussion Our approach is subject to a few limitations. During skeleton extraction, it is impossible to locate a joint if there is no relative motion between adjacent body parts. Therefore, in some of our sequences hands and feet are missed due to insignificant relative motion. However, we consider this to be a principal problem of any data-driven skeleton extraction method, and user interaction is feasible in this case.

Most remaining reconstruction inaccuracies are due to non-rigid deformation components in the input that are not well explainable by a rigid skeleton and linear skinning weights. As part of future work, we plan to investigate if alternative skinning methods lead to an even further reduc-

SEQUENCE	FRAMES	NUMTRI	SEGM	SKEL	SKIN	JACC	ACCU
Horse	47	17K	4s	17s	7s/frame	N/A	0.56% of B.B.
Bird	60	55K	5s	49s	10s/frame	2.9% of B.B.	0.42% of B.B.
Biped	80	32K	5s	36s	14s/frame	1.7% of B.B.	0.52% of B.B.
Cartwheel	120	7K	4s	63s	3s/frame	N/A	0.81% of B.B.
Posing	280	7K	9s	261s	4s/frame	N/A	0.43% of B.B.
Capoeira	150	106K	44s	244s	28s/frame	N/A	0.47% of B.B.
Dancing	220	106K	37s	312s	28s/frame	N/A	0.37% of B.B.

Table 1: *Given an animated mesh sequence with T triangles (NUMTRI) and N frames (FRAMES), the processing times for segmenting the mesh into triangle patches (SEGM), to extract its kinematic structure and reconstruct its motion parameters (SKEL), and to calculate the skinning weights based on the input data (SKIN) are shown. Also, the low average difference between estimated joint positions and true joint locations - in percent of the maximal side length of the overall bounding box (JACC) and the average difference between original and reconstructed vertex position (ACCU) are indicated.*

tion of the residual errors, e.g. [MMG06, KCZO07, AS07, WPP07].

Furthermore, skeletal reparametrization works very well for subjects whose motion is largely due to a skeleton such as humans and most animals. In largely non-rigidly moving animations, such as a deforming piece of cloth, our algorithm would still determine a skeleton, but it is not physically plausible. Therefore, mesh-based editing approaches might be preferred in this case [XZY*07, KG06].

Despite these limitations, we have presented a simple, practical, robust, and easy-to-implement approach to automatically transform an arbitrary mesh animation into a fully-rigged kinematic skeleton animation.

9. Conclusion

We presented a fully-automatic method to extract a kinematic skeleton, joint motion parameters, and surface skinning weights from an animation of a single triangle mesh. The result is a compact representation of the original input that can be easily rendered and modified in standard skeleton-based animation tools without having to modify them in the slightest. This way, we are able to preserve the great modeling flexibility of purely mesh-based approaches while making the resulting skeleton-less animations straightforwardly available to the animator's repertoire of processing tools. Our results show that the efficient combination of skeleton learning and temporally-coherent blending weight computation enables us to effectively bridge the gap between the mesh-based and skeleton-based animation paradigms.

Acknowledgments: This work is supported by EC within FP6 under Grant 511568 with the acronym 3DTV.

References

[AHLD07] AUJAY G., HÉTROY F., LAZARUS F., DEPRAZ C.: Harmonic skeleton for realistic character animation. In *SCA '07* (2007), pp. 151–160.

[AKP*04] ANGUELOV D., KOLLER D., PANG H.-C., SRINIVASAN P., THRUN S.: Recovering articulated object models from 3d range data. In *In Proc. of AUAI '04* (Virginia, USA, 2004), pp. 18–26.

[AS07] ANGELIDIS A., SINGH K.: Kinodynamic skinning using volume-preserving deformations. In *Symposium on Computer Animation* (2007), pp. 129–140.

[BMB87] BADLER N. I., MANOOCHEHRI K. H., BARAFF D.: Multi-dimensional input techniques and articulated figure positioning by multiple constraints. In *SI3D* (New York, NY, USA, 1987), pp. 151–169.

[BP07] BARAN I., POPOVIĆ J.: Automatic rigging and animation of 3d characters. *ACM Trans. Graph. 26*, 3 (2007), 72.

[BS08] BOTSCH M., SORKINE O.: On linear variational surface deformation methods. *IEEE Transactions on Visualization and Computer Graphics 14*, 1 (2008), 213–230.

[dATM*04] DE AGUIAR E., THEOBALT C., MAGNOR M., THEISEL H., SEIDEL H.-P.: M3 : Marker-free model reconstruction and motion tracking from 3d voxel data. In *PG 2004* (Seoul, Korea, October 2004), pp. 101–110.

[dATS06] DE AGUIAR E., THEOBALT C., SEIDEL H.-P.: Automatic learning of articulated skeletons from 3d marker trajectories. In *Proc. ISVC* (2006), pp. 485–494.

[dATSS07a] DE AGUIAR E., THEOBALT C., STOLL C., SEIDEL H.-P.: Marker-less deformable mesh tracking for human shape and motion capture. In *Proc. IEEE CVPR 2007* (June 2007).

[dATSS07b] DE AGUIAR E., THEOBALT C., STOLL C., SEIDEL H.-P.: Rapid animation of laser-scanned humans. In *IEEE Virtual Reality 2007* (Charlotte, USA, 2007), pp. 223–226.

[Hor87] HORN B.: Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America 4(4)* (1987), 629–642.

[JT05] JAMES D., TWIGG C.: Skinning mesh animations. *ACM Transactions on Graphics (Proc. SIGGRAPH) 24*, 3 (2005).

[KCZO07] KAVAN L., COLLINS S., ZARA J., O'SULLIVAN C.: Skinning with dual quaternions. In *ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games* (2007), pp. 39–46.

[KG06] KIRCHER S., GARLAND M.: Editing arbitrarily deforming surface animations. In *SIGGRAPH '06* (2006), ACM Press, pp. 1098–1107.

[KM04] KURIHARA T., MIYATA N.: Modeling deformable human hands from medical images. In *SCA '04* (Aire-la-Ville, Switzerland, 2004), pp. 355–363.

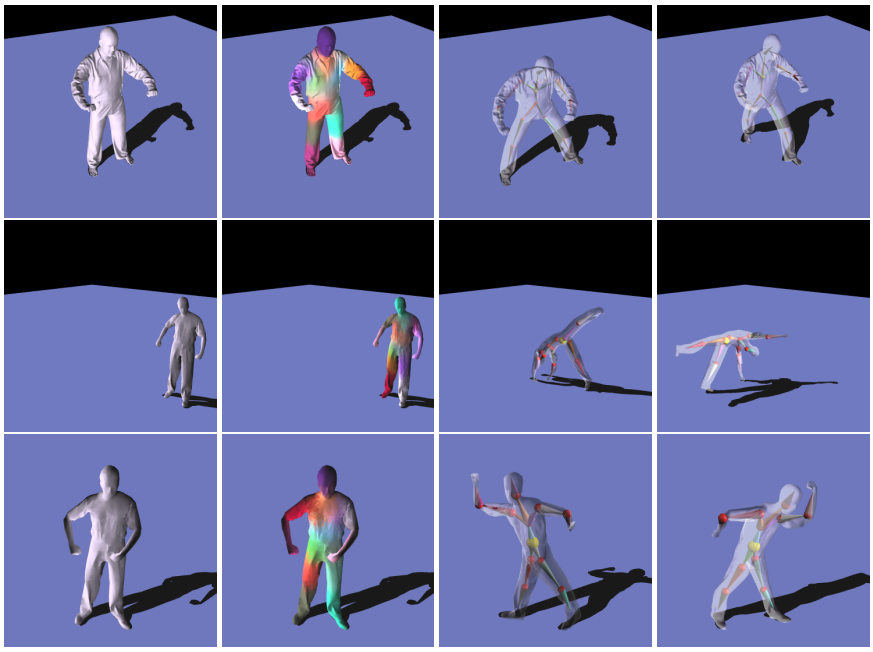

Figure 8: *Results obtained with different input animations (left to right in each row): One frame of the input animation, color-coded blending weight distribution, and two poses of the input recreated with our new skeleton-based representation. Our method efficiently and fully-automatically converts mesh animations into skeleton-based animations.*

[KOF05] KIRK A. G., O'BRIEN J. F., FORSYTH D. A.: Skeletal parameter estimation from optical motion capture data. In *CVPR 2005* (June 2005), pp. 782–788.

[KT03] KATZ S., TAL A.: Hierarchical mesh decomposition using fuzzy clustering and cuts. In *SIGGRAPH '03* (New York, NY, USA, 2003), pp. 954–961.

[LCF00] LEWIS J. P., CORDNER M., FONG N.: Pose space deformation: a unified approach to shape interpolation and skeleton-driven deformation. In *SIGGRAPH '00* (New York, NY, USA, 2000), pp. 165–172.

[LKA06] LIEN J.-M., KEYSER J., AMATO N. M.: Simultaneous shape decomposition and skeletonization. In *SPM '06* (New York, NY, USA, 2006), pp. 219–228.

[Lue73] LUENBERGER D. G.: *Introduction to Linear and Nonlinear Programming*. New York, NY, USA, 1973.

[MG03] MOHR A., GLEICHER M.: Building efficient, accurate character skins from examples. In *SIGGRAPH '03* (New York, NY, USA, 2003), pp. 562–568.

[MMG06] MERRY B., MARAIS P., GAIN J.: Animation space: A truly linear framework for character animation. *ACM Trans. Graph. 25*, 4 (2006), 1400–1423.

[NJW02] NG A. Y., JORDAN M., WEISS Y.: On spectral clustering: Analysis and an algorithm. In *Proc. NIPS* (2002).

[Pop07] POPPE R.: Vision-based human motion analysis: An overview. *Computer Vision and Image Understanding 18*, 1 (2007).

[Sha08] SHAMIR A.: A survey on mesh segmentation techniques. *Computer Graphics Forum* (2008).

[SLSK07] SHARF A., LEWINER T., SHAMIR A., KOBBELT L.: On–the–fly curve-skeleton computation for 3d shapes. In *Eurographics* (Prague, Czech, September 2007), pp. 323–328.

[SP04] SUMNER R. W., POPOVIC J.: Deformation transfer for triangle meshes. In *SIGGRAPH '04* (2004), ACM Press, pp. 399–405.

[SY07] SCHAEFER S., YUKSEL C.: Example-based skeleton extraction. In *SGP '07* (Aire-la-Ville, Switzerland, 2007), pp. 153–162.

[TRdAS07] THEOBALT C., RÖSSL C., DE AGUIAR E., SEIDEL H.-P.: Animation collage. In *Proc. SCA* (2007), pp. 271–280.

[WMKG07] WARDETZKY M., MATHUR S., KAELBERER F., GRINSPUN E.: Discrete laplace operators: No free lunch. In *Symposium on Geometry Processing* (2007), pp. 33–37.

[WP02] WANG X. C., PHILLIPS C.: Multi-weight enveloping: least-squares approximation techniques for skin animation. In *SCA '02* (New York, NY, USA, 2002), pp. 129–138.

[WPP07] WANG R. Y., PULLI K., POPOVIĆ J.: Real-time enveloping with rotational regression. *ACM Trans. Graph. 26*, 3 (2007), 73.

[XZY*07] XU W., ZHOU K., YU Y., TAN Q., PENG Q., GUO B.: Gradient domain editing of deforming mesh sequences. *ACM TOG 26*, 3 (2007), 84.

[YBS07] YOSHIZAWA S., BELYAEV A. G., SEIDEL H.-P.: Skeleton-based variational mesh deformations. In *Eurographics* (Prague, Czech, September 2007), pp. 255–264.

[YGZS05] YAMAUCHI H., GUMHOLD S., ZAYER R., SEIDEL H.-P.: Mesh segmentation driven by gaussian curvature. *The Visual Computer 21*, 8-10 (2005), 649–658.