Example-Based Wrinkle Synthesis for Clothing Animation

Huamin Wang Florian Hecht Ravi Ramamoorthi James O'Brien

University of California, Berkeley

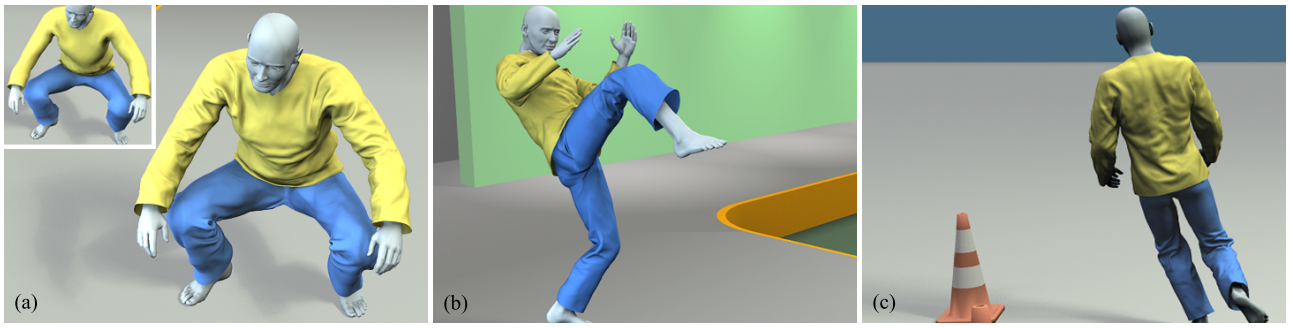


Figure 1: *Our method uses a precomputed dataset to synthesize detailed cloth wrinkles (a) that are layered onto a coarse base simulation (inset). The precomputed datset can be used to synthesize wrinkles for a wide range of poses (b and c).*

Abstract

This paper describes a method for animating the appearance of clothing, such as pants or a shirt, that fits closely to a figure's body. Compared to flowing cloth, such as loose dresses or capes, these types of garments involve nearly continuous collision contact and small wrinkles, that can be troublesome for traditional cloth simulation methods. Based on the observation that the wrinkles in close-fitting clothing behave in a predominantly kinematic fashion, we have developed an example-based wrinkle synthesis technique. Our method drives wrinkle generation from the pose of the figure's kinematic skeleton. This approach allows high quality clothing wrinkles to be combined with a coarse cloth simulation that computes the global and dynamic aspects of the clothing motion. While the combined results do not exactly match a high-resolution reference simulation, they do capture many of the characteristic fine-scale features and wrinkles. Further, the combined system runs at interactive rates, making it suitable for applications where high-resolution offline simulations would not be a viable option. The wrinkle synthesis method uses a precomputed database built by simulating the high-resolution clothing as the articulated figure is moved over a range of poses. In principle, the space of poses is exponential in the total number of degrees of freedom; however clothing wrinkles are primarily affected by the nearest joints, allowing each joint to be processed independently. During synthesis, mesh interpolation is used to consider the influence of multiple joints, and combined with a coarse simulation to produce the final results at interactive rates.

Keywords: Clothing animation, cloth simulation, example-based animation, wrinkles, precomputed animation.

Contact email: {whmin, fhecht, ravir, job}@eecs.berkeley.edu

From the ACM SIGGRAPH 2010 conference proceedings.

1 Introduction

Garments that conform approximately to the shape of the wearer's body, such as a shirt or pants, typically exhibit wrinkles and fine buckling patterns. Unlike flowing skirts and capes, most parts of these fitted garments[1] are in nearly continual contact with the body. Current techniques for cloth animation can produce highly realistic results for animated clothing with detailed wrinkle patterns. However, solving cloth dynamics on high-resolution meshes can be expensive, especially when the scenario involves a large number of collisions and intermittent contacts.

In this paper, we propose a fast clothing animation system that combines synthesized fine wrinkle details with coarse cloth dynamics. The key observation behind this work is that fine-scale wrinkles can often be approximated kinematically by matching to a database of precomputed cloth configurations, while global dynamic behavior can be captured with a coarse simulation on a low-resolution mesh.

Our method operates by first constructing a wrinkle database for a given garment, based on high-resolution cloth simulations that exercise each joint of a skinned articulated figure over its range of motion. This precomputation phase is time consuming, but it only needs to be done once for a given figure-garment pair and can subsequently be used for a wide variety of motions. Figure 1 shows an example of typical results for our method, and demonstrates a single precomputed database being used for a range of different motions.

Our online phase efficiently synthesizes wrinkle meshes for novel human poses. We first interpolate within each joint to obtain a mesh for the desired joint angle, and then merge joint wrinkles together into a fine clothing mesh for the whole pose input. Finally, we combine the output with a low-resolution cloth simulation that accounts for gravity, collisions among different clothes, and dragging and compression of clothes across different joints. All three steps can be processed efficiently using graphics hardware, with the whole system running at interactive rates.

There are three main observations that motivate the assumptions that our system relies on:

- First, clothing wrinkles are formed predominantly by the movement of the figure's articulated joints. Although the

[1] In the fashion industry a *fitted garment* describes a particular style of tailoring. Here we will use the term more generally to mean a garment that roughly follows the shape of the body parts that it covers.

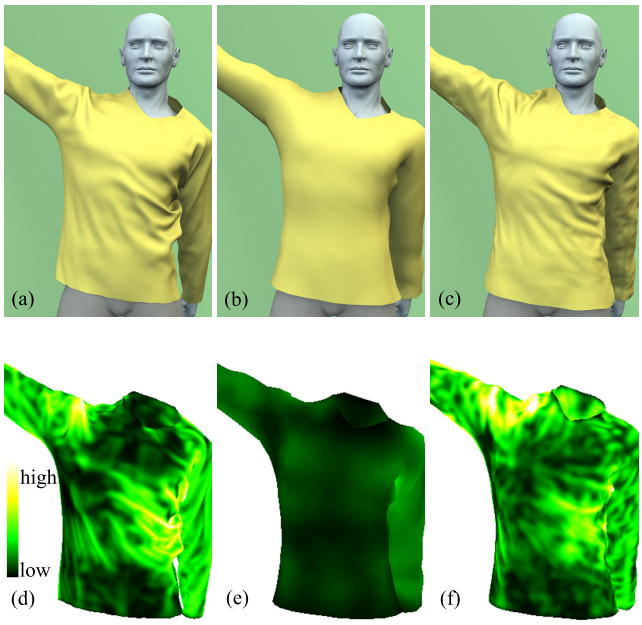


Figure 2: *The high-resolution simulation result shown in (a) used a mesh of 25k vertices, which required 2-3 minutes per frame of animation (0.007 frames per second). The coarse simulation shown in (b) on a 638 vertex mesh takes only 0.072 seconds per frame (13.9 FPS) but lacks detail. Our result, shown in (c), adds wrinkles to the coarse result and runs at 0.084 seconds per frame (11.9 FPS), and captures many of the fine-scale features and wrinkles. In (d), (e), and (f) the wrinkle patterns are emphasized by showing the vertex mean curvature magnitude of each mesh in (a), (b), and (c), respectively.*

wrinkle motion can be affected by external forces and global cloth dynamics, the wrinkle shapes, particularly at small scale, are largely determined by the amount of compression or stretching due to motion of the nearby body parts. Therefore, we can separate small-scale wrinkle behavior from the larger scale cloth motion, and use separate methods for each.

- Second, the pattern of wrinkles generated for a particular body pose looks similar each time that pose is reached. This phenomenon occurs due to a combination of the fabric's memory property, and the many kinematic constraints imposed by the large number of contacts between the cloth and body. Further, with flexion/extension the cloth gets more/less compressed, and the same wrinkles are intensified/reduced as a function of joint angles. This observation implies that a wrinkle database can be used to synthesize new wrinkles through interpolation using body poses as variables.

- Finally, joint motion usually only affects the surrounding cloth regions, and its influence is limited by friction between cloth and the human body. For instance, movement of the left arm typically has negligible effect on the right sleeve. Therefore, instead of sampling over the figure's entire pose space for the wrinkle database, we can gather samples for each individual joint separately and use blending to model areas that are affected by multiple joints. This property is crucial to make the database construction manageable, as otherwise constructing it would require computation and storage exponential in the number of degrees of freedom in the articulated skeleton.

Although these assumptions are not universally true and do not apply in all circumstances, our results show that the cloth synthesis method they motivate can produce compelling results at interactive rates. For instance, while our result in Figure 2c does not exactly match the high-resolution simulation in Figure 2a, it does capture many of the fine-scale details and wrinkle patterns missing from the coarse simulation in Figure 2b. Most importantly, our result looks plausible and is computed at a very small fraction of the cost of a comparable full resolution simulation.

2 Background

Cloth Simulation: In recent years, physically-based cloth simulation has generated highly realistic results for clothing animation [Baraff and Witkin 1998; Choi and Ko 2002; Bridson et al. 2002; Kaldor et al. 2008], and surveys articles have been written that summarize current state of the art [House and Breen 2000; Choi and Ko 2005; Nealen et al. 2006]. A key feature of realistic cloth appearance is the formation of fine-scale wrinkles. Many textiles are made from stiff threads, so that woven cloth is typically highly resistant to in-plane stretching and compression. When the otherwise flexible material is bent out of plane in a compound fashion, it forms characteristic folds and buckling patterns. Accurately simulating these features requires both high resolution to capture geometric detail, and a stiff constitutive model. A number of recent efforts have focused specifically on this challenge [Bridson et al. 2003; Goldenthal et al. 2007; English and Bridson 2008; Volino et al. 2009; Selle et al. 2009]. Although these methods can produce highly detailed results, doing so requires substantial computation time, typically several minutes per frame of animation.

Other methods have been designed for interactive applications. Some examples of successful strategies include specially designed integration methods [Jakobsen 2001; Meggs 2005], GPU acceleration [Bordes et al. 2009], and explicit constraint enforcement to avoid stiff constitutive models [Provot 1995; Müller 2008; Thomaszewski et al. 2009]. Unfortunately, all of these methods fail to resolve fine wrinkles, either because performance demands require insufficient detail in the simulation mesh and/or because the fast simulation method suppresses buckling and wrinkle formation.

Our method is designed to work with existing cloth simulation techniques, such as those described above. A highly realistic, but likely slow, method is used to precompute offline the examples for the wrinkle database. Then at runtime, a faster method is used online to produce a coarse result, that is enhanced by wrinkles synthesized from the example database. Future advances in cloth simulation can directly be used to improve both offline and online phases.

Wrinkle Generation: Other researchers have also investigated detail enhancement through wrinkle generation. Hadap and colleagues [1999] proposed a texture-based method to generate wrinkle textures from triangle deformation. They present compelling results but we believe our approach offers improved realism, particularly for fitted garments. Others have modeled fine wrinkles procedurally to approximate wrinkling phenomena on skin and cloth [Kang et al. 2001; Larboulette and Cani 2004; Decaudin et al. 2006; Eibner et al. 2009], or used a stress map to synthesize new wrinkles from a database manually created by artists [Cutler et al. 2007]. Work by Popa and colleagues [2009] used a wrinkle generation method to improve the quality of specific frames of captured cloth, based on shadows in the recorded images. In contrast, our method builds wrinkles from simulated examples, and offers indirect control over the results by choice of the simulation method and database parameters.

Cloth in Games: Outside the academic graphics literature, real-time methods have been developed for animating cloth on video game characters. Earlier examples used simple static textures, but more recent titles typically use a combination of skinning methods with static normal maps and geometry textures for fitted clothing, and a combination of procedural methods and low-resolution simulation for flowing garments like capes. Recent examples include *Batman: Arkham Asylum*, *Assassins' Creed II*, and *Star Wars: The Force Unleashed*. While the combination of skin deformers and textures can create very realistic still images, the static nature of wrinkles becomes apparent when characters move.

Skin Animation and Wrinkles: Similar to fitted clothing, skin also has wrinkles. Often used in games [Lee 2006], one way to animate skin is to use skeleton-based deformers [Capell et al. 2002; Vasilakis and Fudos 2009]. Shi and colleagues [2008] further applied a data-driven deformable model to obtain secondary deformation in real time. Alternatively, example-based methods rephrase the animation problem as a pose-space interpolation problem using existing skin and wrinkle samples. Other researchers have demonstrated how to capture skin data with cameras, and use example-based deformation methods specifically tailored to skin animation [Allen et al. 2002] and [Mohr and Gleicher 2003]. In order to interpolate arbitrary shapes in the pose space, radial basis functions can be used to control a sparse set of vertices as landmarks in [Lewis et al. 2000]. Alternatively, least-square fitting can be applied to model tightly-fit clothing deformations without many wrinkles [Wang and Phillips 2002]. A combination of eigenvalue decomposition and joint-based interpolation can be used to avoid high dimensional pose-space sampling [Kry et al. 2002]. When the computational cost is not a major concern, Laplace's equation can be used to deform skin and tightly-fit cloth [Weber et al. 2007]. Kavan and colleagues [2010] show how skin deformers can be used to efficiently compress complex cloth motion.

Kim and Vendrovsky [2008] proposed an interpolation method to generate new clothing animations by minimizing an energy functional over the pose space. However, their method can only handle a sparse set of wrinkle samples, and it is not straightforward for GPU implementation. In contrast, our method can produce more significant wrinkles from dense sampled database, and do so efficiently on the GPU. For this reason, our work is closely related to the method presented by Park and Hodgins [2008]. They separated static skin deformation, which can be animated as a function of pose, from dynamic skin deformation, which is approximated by dynamic equations. In our technique, fine wrinkles are modeled by human poses and global dynamics by a coarse cloth simulation.

Precomputed Animation: Many existing methods for example-based animation techniques have focused mainly on flowing cloth motions or other non-cloth systems [James and Fatahalian 2003; Cordier and Magnenat-Thalmann 2005; White et al. 2007; James et al. 2007]. While these techniques could potentially be applied to animating fitted clothing, they would require some modification to avoid problems with the large state space of an articulated figure. Our method avoids this problem with a simple approach that separates wrinkles from coarse cloth dynamics, and further separates wrinkles into blended sets driven by each joint.

3 Algorithm

An overview for our system is shown in Figure 3. It takes the human pose and coarse cloth simulation in each frame as the input, and generates finely detailed clothing meshes as output. Given the angle configuration for each joint, a wrinkle mesh corresponding to that joint is first interpolated using a database of precomputed wrinkles. Separate wrinkle meshes for each joint are merged together into a single mesh for the full human pose. Finally, wrinkle details are transferred to the coarse simulation, to obtain clothing animations with fine details.

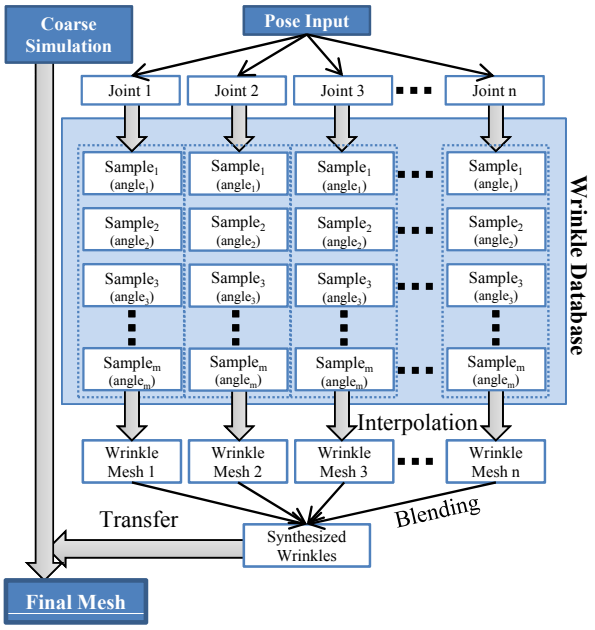
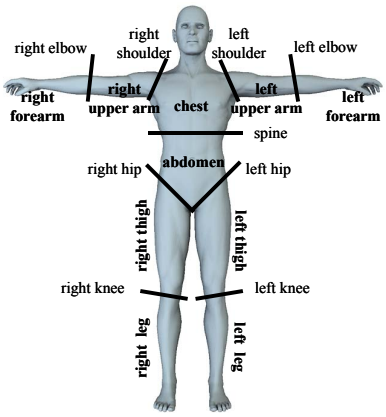


Figure 3: *The system overview.*

3.1 Data Precomputation

The wrinkle database is made of wrinkle meshes sampled separately for each joint as shown in Figure 3. Each mesh corresponds to a joint configuration, sparsely sampled over the allowed joint space. A typical database for a human body covers nine major joints: left elbow, right elbow, left knee, right knee, spine, left shoulder, right shoulder, left hip, and right hip, as shown in Figure 4. The first four joints are hinge joints with only one degree of freedom (DoF), while the last four are ball-in-socket with three DoFs each. The spine joint is simplified from a series of actual spine joints and we use three DoFs for it as well. Although this database can be easily extended to cover additional joints, for example wrists and ankles, we found doing so unnecessary because these other joints have less effect on wrinkles for most clothing designs. We sample each joint angle uniformly by 15 or 30 degrees. As an example, the spine joint uses three samples for the twisting angle, three samples for the front-back angle, and five samples for the bending angle, so we have 45 joint samples in all. Overall there are 325 samples for five joints on the upper body and 202 samples over four joints on the lower body. We emphasize that each joint is sampled independently, so the total number of configurations is just the sum of the number of configurations for each joint.

The high-resolution wrinkle mesh for each sample is generated by simulating the clothing mesh from the resting human pose to the sample pose. We use a mass-spring system to simulate cloth dynamics by adding length springs for each triangle edge, and bending springs for each opposing vertex pair in neighboring triangles as proposed by Choi and Ko [2002]. We use an implicit cloth solver similar to that use by Baraff and Witkin [1998] to calculate velocity updates in each time step and a fourth-order Runge-Kutta method to integrate over time. A signed distance function defined over a 3D regular grid is calculated ahead of time to handle cloth collisions with the human body [Bridson et al. 2003]. A friction spring proportional to the collision force is added to produce a dynamic fric-



Name	# samples
Elbow (L)	14
Elbow (R)	14
Knee (L)	11
Knee (R)	11
Spine	3×3×5
Shoulder (L)	3×6×7
Shoulder (R)	3×6×7
Hip (L)	3×3×10
Hip (R)	3×3×10

Figure 4: *The human joints and their influence regions.*

tion effect. We also decrease the magnitude of other forces applied to colliding cloth vertices to mimic a static friction effect. Self collisions are only tested for each vertex-triangle pair, and when they become close to each other, a penalty spring will be added to avoid penetration. It is possible that self penetration still happens when wrinkles collapse heavily over each other under a large bending angle. Fortunately, they are mostly occluded and not visible from the outside.

We emphasize that any fine-scale cloth solver can be used for the precomputation step, and newer algorithms in the future will provide better precomputed data. We also note that some level of artistic control could potentially be exercised at this point as stylistic variations of the precomputed cloth should appear later during synthesis.

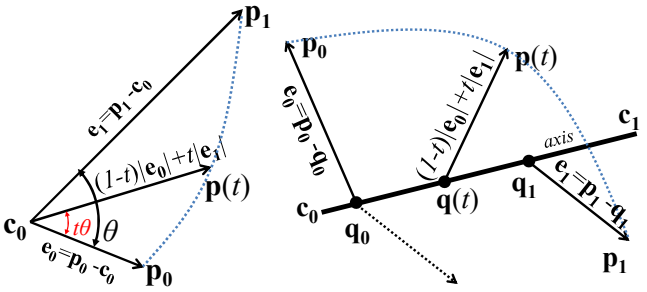
3.2 Coarse Simulation

The coarse cloth simulation must quickly simulate cloth dynamics for a simplified clothing mesh (using 600 to 800 vertices in our experiments). The input to this simulation is a human body mesh for each frame supplied from external sources, which can also be a collision proxy in a coarser form than the one used for display.

We implemented an online solver similar to the one used in Section 3.1 for data construction, except that a second-order implicit scheme replaces the RK4 method for temporal integration and the collision detection is handled differently. To speed up the solver, we ignore self collisions because wrinkle collapses have been covered mostly in the wrinkle database. Cloth-body collisions and cloth-cloth collisions among different garment meshes still need to be considered.

During the data construction step, cloth-body collisions are detected by first converting the body mesh into volumetric data as a precomputation step. Unfortunately, this is no longer acceptable for coarse simulation since volumetric conversion will be too time consuming to calculate on the fly. Instead, we assume that all triangle normals in the body mesh are defined toward the outside, so we can simply use the nearest triangle normals to do the inside-outside test, with the distance to the nearest triangle as the penetration amount.

Cloth-cloth collisions are detected in the same fashion when the clothes are worn in layers. For example, when the long shirt always covers the pants, a cloth-cloth collision happens if and only if a vertex of the pants jumps to the outside of the shirt and it is on top of its closest shirt triangle (its projection on the plane of the closest shirt triangle is within that triangle). This body penetration test fails if the body mesh becomes inside out, in which case a better yet slower collision detection algorithm should be used instead.



(a) A bending joint angle. (b) A twisting joint angle.

Figure 5: *The interpolation schemes for two types of joint angles. Blue dotted curves represent the interpolation trajectories.*

3.3 Wrinkle Synthesis

The database constructed in Section 3.1 is then used to synthesize fine wrinkle meshes for arbitrary human pose. The synthesis process is done in two steps. First, sparse mesh samples are interpolated into a wrinkle mesh, corresponding to each desired joint separately. Wrinkle meshes are then composed together into a single clothing mesh for the whole human pose.

Because all joint motions are rotational, a rotational interpolation scheme performs better than spatial linear interpolation, which causes severe shrinking artifacts especially when joint angles are sparsely sampled. The joint location $\mathbf{c_0}$ can be directly determined from the human model. We assume the joint rotates at a constant speed, and we linearly interpolate both the joint angle and the distance to the joint, as shown in Figure 5a. In this figure, $\mathbf{p_0}$ and $\mathbf{p_1}$ are positions of the same vertex in two precomputed meshes, and t is a number between 0 and 1 that controls the interpolation. Let $\mathbf{e_0} = \mathbf{p_0} - \mathbf{c_0}$ and $\mathbf{e_1} = \mathbf{p_1} - \mathbf{c_0}$ be the vectors from $\mathbf{c_0}$ to $\mathbf{p_0}$ and $\mathbf{p_1}$ respectively, and $\hat{\mathbf{e}}_0 = \mathbf{e_0}/|\mathbf{e_0}|$ and $\hat{\mathbf{e}}_1 = \mathbf{e}_1/|\mathbf{e}_1|$ be the normalized vectors. We obtain,

$$\mathbf{p}(t) = \mathbf{c_0} + ((1-t)\,|\mathbf{e_0}| + t\,|\mathbf{e_1}|)\,R(\hat{\mathbf{n}}, t\theta)\hat{\mathbf{e}}_0 \tag{1}$$

where $\mathbf{R}(\mathbf{u}, \theta)$ is the rotation matrix that transforms $\hat{\mathbf{e}}_0$ to $\hat{\mathbf{e}}_1$ by rotating around the normalized axis $\hat{\mathbf{n}} = \hat{\mathbf{e}}_0 \times \hat{\mathbf{e}}_1$ by angle θ.

This scheme works for most joint angles except for the twisting angle, which uses the bone $\mathbf{n} = \mathbf{c_1} - \mathbf{c_0}$ as the axis as shown in Figure 5b. In this case, we linearly interpolate the distance to the axis, the rotational angle, and also the distance along the axis, similar to a screw motion in dual-quaternion interpolation. Let $\hat{\mathbf{n}} = \mathbf{n}/|\mathbf{n}|$ be the normalized rotational axis and $\mathbf{q_0}$ and $\mathbf{q_1}$ be $\mathbf{p_0}$ and $\mathbf{p_1}$'s projection on the bone respectively. $\mathbf{q}(t)$ is linearly interpolated as:

$$\mathbf{q}(t) = (1-t)\mathbf{q_0} + t\mathbf{q_1}, \tag{2}$$

Given $\mathbf{e_0} = \mathbf{p_0} - \mathbf{q_0}$ and $\mathbf{e_1} = \mathbf{p_1} - \mathbf{q_1}$, $\mathbf{p}(t)$ is calculated as:

$$\mathbf{p}(t) = \mathbf{q}(t) + ((1-t)\,|\mathbf{e_0}| + t\,|\mathbf{e_1}|)\,R(\hat{\mathbf{n}}, t\theta)\hat{\mathbf{e}}_0 \tag{3}$$

Once wrinkle meshes are interpolated from the database for each joint separately, the next step is to merge them together for the whole human pose. We first segment each clothing mesh into regions according to body joints as shown in Figure 4. Typically, a long shirt covers the top six regions and pants cover the five regions on the bottom. Under the assumption that each joint only affects its two neighboring regions, we code this influence by a map between each cloth vertex $\mathbf{v}$ and the joint connecting to its residing region. While the joint is only defined as a single point for rotations in the skeleton model, the blending method treats the seam between two

adjacent regions as a practical joint. Therefore, the influence map is calculated using the distance from $\mathbf{v}$ to neighboring regions. As an example, influence values of a vertex $\mathbf{v}$ in the chest region are calculated as:

$$\begin{aligned} w_{ls}(\mathbf{v}) &= d_a(\mathbf{v})d_r(\mathbf{v})/(d_a(\mathbf{v})d_r(\mathbf{v}) + d_a(\mathbf{v})d_l(\mathbf{v}) + d_l(\mathbf{v})d_r(\mathbf{v})) \\ w_{rs}(\mathbf{v}) &= d_a(\mathbf{v})d_l(\mathbf{v})/(d_a(\mathbf{v})d_r(\mathbf{v}) + d_a(\mathbf{v})d_l(\mathbf{v}) + d_l(\mathbf{v})d_r(\mathbf{v})) \\ w_{sp}(\mathbf{v}) &= d_l(\mathbf{v})d_r(\mathbf{v})/(d_a(\mathbf{v})d_r(\mathbf{v}) + d_a(\mathbf{v})d_l(\mathbf{v}) + d_l(\mathbf{v})d_r(\mathbf{v})) \end{aligned} \quad (4)$$

in which $w_{ls}(\mathbf{v})$, $w_{rs}(\mathbf{v})$ and $w_{sp}(\mathbf{v})$ are the weights from left shoulder, right shoulder and spine joint respectively, and d_l, d_r and d_a are Euclidean distances from $\mathbf{v}$ to left upper arm, right upper arm and abdomen, respectively. Once weights are determined for each vertex, wrinkle meshes can be linearly blended using the weights in each overlapping region, and the result is a fine mesh that corresponds to the whole human pose.

3.3.1 Analysis

The assumption that each joint only affects wrinkles locally allows us to construct the wrinkle database independently for each joint, and to merge wrinkle meshes together in overlapping regions. Such an assumption is also seen in example-based skin animations, such as the method presented by Allen and colleagues [Allen et al. 2002]. Here we provide some quantitative analysis of this assumption for clothing animation, and discuss potential artifacts and solutions.

Our experiment is carried out as follows. For a joint J, we calculate the offset vector for each vertex between any two of J's sample meshes in the wrinkle database. This vector represents how the vertex moves from one joint configuration to another. We then apply a Laplacian operator onto all vectors over the mesh to remove locally-constant dragging or shifting motions, which can be modeled in coarse simulation. The remaining vector is considered to contribute more to the wrinkles, so we sum its magnitude for all J's sample mesh pairs. The results for two joints are plotted in Figure 6.

One can see that the blending weight proposed in Section 3.3 approximately matches with the influence weight from the experiment. However, the shoulder joint also has slight influence over the forearm and the abdomen shown in Figure 6e and 6f, which is not considered in our model (Figure 6a and 6b). In other words, wrinkles on the forearm and the abdomen will not be affected by the shoulder joint as they should be. Fortunately, we have not found significant artifacts as a result. To solve this problem, more sophisticated blending weights and algorithms could be used, which may however cause further computational cost. Another issue is that the space of wrinkles generated independently for each joint may not span all possible wrinkles in the blending region. An ideal solution is to build a database based on regions rather than on separate joints, and each region should be parameterized by all neighboring joints. In that case, more subtle wrinkle effects could be included but it would require constructing and storing a very large database.

Another interesting observation is that the elbow joint has more influence over the inner elbow (Figure 6g) than the outer elbow (Figure 6h). This is expected because wrinkles are formed mostly in the inner elbow.

3.4 Wrinkle Detail Transfer

Given the synthesized cloth wrinkle mesh that only takes the human pose into account, the final step is to align this wrinkle mesh with coarse simulation. Instead of using techniques proposed for large deformation [Sumner and Popović 2004; Baran et al. 2009], here we use a simple down-sampling and up-sampling method, since the deformation from the fine mesh to the coarse mesh is usually small.

Let M_f be the synthesized wrinkle mesh from the database and M_c be a coarse mesh produced by coarse simulation. We formulate the correspondence by assigning a bidirectional influence weight a_{uv} between every vertex $\mathbf{u} \in M_f$ and $\mathbf{v} \in M_c$. In practice, we find the nearest triangle in M_c for every $\mathbf{u}$ and a_{uv} is simply defined as the barycentric coordinate of $\mathbf{u}$'s projection in that triangle. Therefore, a_{uv} is nonzero only if $\mathbf{v}$ is a vertex belonging to $\mathbf{u}$'s closest triangle in M_c. We do not encourage building M_f from M_c using subdivision schemes because they can exaggerate discontinuous artifacts as will be discussed later.

The detail transfer is processed by first down-sampling the fine mesh M_f onto M_c in order to find the offset between M_c and M_f for every $\mathbf{v} \in M_c$,

$$\mathbf{o}(\mathbf{v}) = \mathbf{v} - \frac{\sum_{\mathbf{u}\in M_f} a_{uv}\mathbf{u}}{\sum_{\mathbf{u}\in M_f} a_{uv}} \quad (5)$$

$\mathbf{o}(\mathbf{v})$ is then up-sampled and applied to each $\mathbf{u} \in M_f$:

$$\mathbf{u} = \mathbf{u} + \frac{\sum_{\mathbf{v}\in M_c} a_{uv}\mathbf{o}(\mathbf{v})}{\sum_{\mathbf{v}\in M_c} a_{uv}} \quad (6)$$

One issue in using barycentric coordinates as influence weights is that the upsampled offset will not be C^1 continuous along M_c's triangle edges. This artifact can be noticed as a ghosting coarse mesh on top of the fine mesh in the form of shallow folds. For polygon meshes built from regular grids, this could probably be avoided by high-order interpolation schemes, such as bicubic interpolation. Unfortunately, formulating a similar high-order interpolation scheme for an arbitrary triangle mesh cannot be trivially implemented in an efficient way. Instead, we apply Laplacian smoothing on the up-sampled offsets over M_f so that the fine mesh can deform smoothly toward its coarse version.

Slight cloth-body penetration may be introduced when the coarse mesh varies greatly from the fine mesh. To avoid this problem, we use a safety buffering distance when testing cloth-body collisions in coarse simulation, as discussed in Section 3.2.

If two joint angles are too close to each other, the blending region in Section 3.3 may not have enough space to form a smooth transition from one mesh to another. For example, the abdomen region between the left and right hip joints. In this case, we found it is more effective to first deform each wrinkle mesh for each joint separately toward coarse simulation so they get initially aligned, and then blend them together to finalize the synthesis step.

4 Implementation and Results

We now describe our results, discussing our offline and online phases, and showing image still frames. The accompanying video demonstrates a number of clothing animations.

Precomputed Database: We built four wrinkle databases for four different clothes: men's long shirt with 25k vertices, men's long pants with 21k vertices (both seen in Figure 1), women's short shirt with 23k vertices, and women's cropped pants with 11k vertices (both seen in Figure 8). Each database typically took two to four days to construct with unoptimized code, depending on the number of samples and vertices. The memory and storage requirement for each database ranges from 50 to 100MB. Once the database is calculated for a human body model, it can be used to synthesize general human motions for the same model. The human model is represented by a triangle mesh with 30k vertices and 50k triangles.

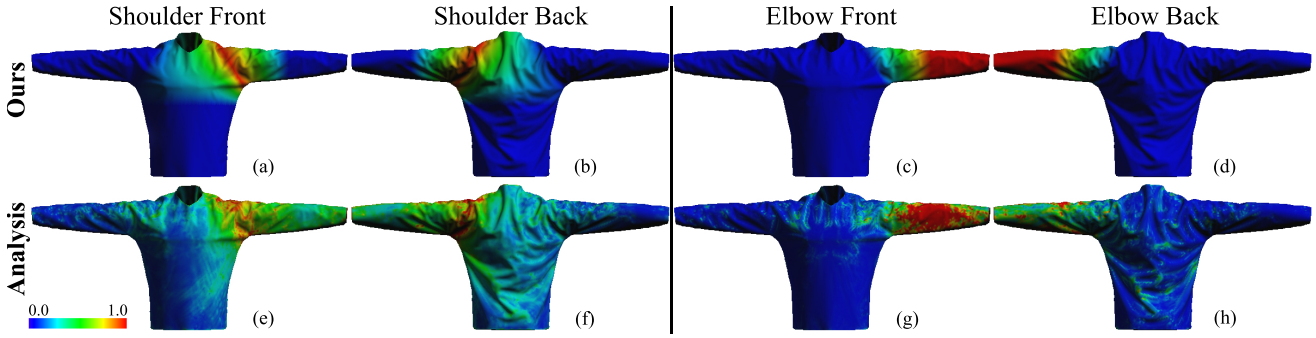


Figure 6: *The influence map over the clothing mesh for the left shoulder joint (left) and the left elbow joint (right). Pictures on the top are calculated from the blending method in Equation 4 and pictures on the bottom are from the analysis experiment.*

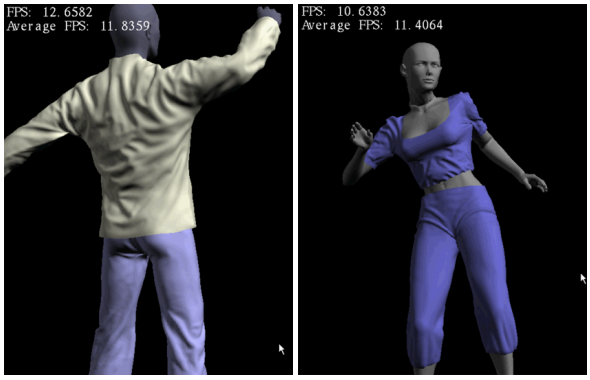


Figure 7: *Screenshots from a realtime recording of our system synthesizing cloth animations.*

Coarse Cloth Simulation: For the online coarse simulation, we use meshes with between 600 and 800 vertices. The collision detection also uses a simplified version of the human model with 11k vertices and 20k triangles. We compute the coarse simulation using three CPU threads to process each garment (shirt, pant) and detail transfer simultaneously. Although the video frames are spaced at 1/30 second intervals, the coarse cloth simulation takes four or five simulation steps per rendered frame. Synchronization is required for all simulation threads after each cycle, to guarantee the accuracy of cloth-cloth collisions. We also tried implementing the coarse simulation in graphics hardware, but found it difficult to efficiently maintain locality coherence in collision detection. The coarse simulation is not used for the short shirt in the female character examples because dynamic effects become less obvious when the shirt is tightly attached to the human body.

Wrinkle Synthesis and Performance: Wrinkle databases are loaded into GPU memory for fast access during computation. The wrinkle synthesis step and the detail transfer are implemented on both CPU and GPU (using CUDA). The system currently runs on a Dell T7500 workstation with a dual quad core 2.26 GHz processor and an Nvidia Quadro FX 5800 graphics card. The average timings over 500 frames for the male character with 46k cloth vertices, and the female character with 34k cloth vertices, are listed in Table 1. Our overall system has a speed of 12 FPS, and for the first time brings fine-scale clothing animation into the interactive realm. This is at least three orders of magnitude faster than the high-resolution simulation. Screenshots from our interactive system are shown in Figure 7.

Name	Processor	Man	Woman
Wrinkle Synthesis	GPU	0.009s	0.008s
Shirt Simulation	CPU	**0.074s**	-
Pants Simulation	CPU	0.061s	**0.078s**
Wrinkle Transfer	CPU/GPU	0.072s	0.041s
Total		0.084s	0.086s
Frame Rate		11.93	11.65

Table 1: *Average timings for the male and female character examples. Thread bottlenecks are emphasized in bold font. Times listed are for computing a video frame update for motion sampled at 1/30 second intervals.*

Animation Sequences: Any source of either interactively generated or pre-recorded human pose data can be used to create animation sequences. In our examples, we use available motion-capture data. We show three animations for the same male character using the long shirt and long pants database: *stretching* (Figure 1a), *kicking* (Figure 1b), and *running* (Figure 1c). We also demonstrate two animations for the female character: *dancing* (Figure 8a and b) and *pointing* (Figure 8c and d). Note that figures in the paper, other than Figure 7, show rendered images computed offline with ambient occlusion to facilitate visualizing the wrinkles. The video also shows an interactive session with real-time animation and hardware rendering. Natural clothing motions with fine-scale details and wrinkles are produced in all cases, with the wrinkles changing in response to the global dynamics.

Limitations: Although our experiments show that our method can generate realistic clothing animations for a number of common cases, we also observe that our result quality is not the same as the high resolution simulation (Figure 2). The differences occurs, in part, because the wrinkle database is simplified by considering each joint separately, rather than the exponential number of all possible human pose configurations. The blending method is also an approximation to the actual complicated joint influence model on wrinkles. As a result, the synthesized mesh may have more wrinkles compared with the ground truth and wrinkles appear to be more static. This can be seen in Figure 9 where the curve of the mean curvature sum in the high-resolution simulation (blue) has stronger fluctuations than the final result (red). One can also notice from Figure 9 that the curve of the final result (red) has a similar shape to the curve of the wrinkle synthesis result (green), because the coarse simulation barely introduces any wrinkles into the final result during the detail transfer step. However, it slightly reduces the wrinkle strength due to a smoothing effect, so the final result curve is slightly lower than the wrinkle synthesis curve (green).

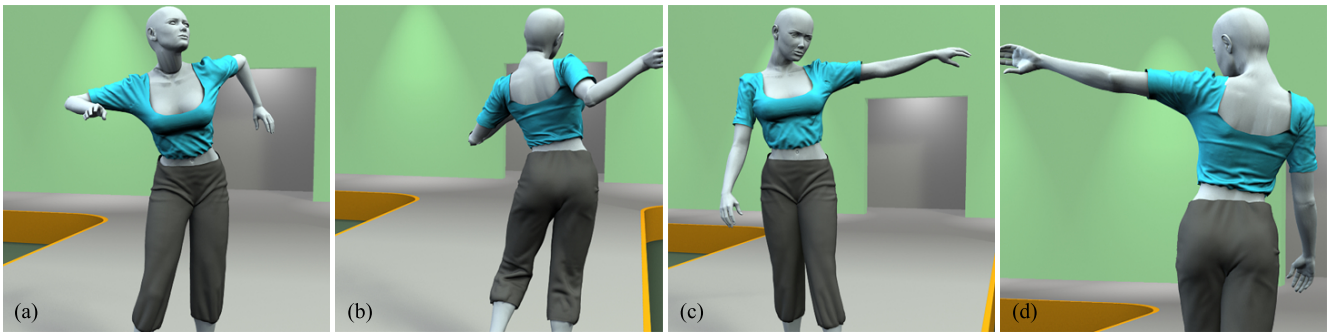


Figure 8: *A female character dancing (a and b), and pointing (c and d).*

Since the synthesis step depends on the pose configuration rather than the motion history, and the coarse simulation usually does not introduce new wrinkles, almost identical wrinkle patterns will be generated given the same pose. Indeed, similar wrinkle patterns can be noticed in the results for our method, for a repetitive human motion.

Our method is also not suitable for clothing that is loosely fit to the human body, such as skirts and capes, because the wrinkles will not be well determined by the human pose alone, and their motion is too nonlinear to be interpolated by a small precomputed database.

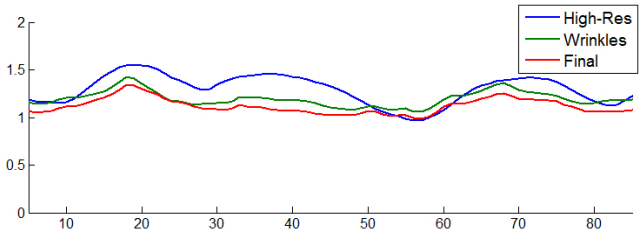


Figure 9: *The vertex mean curvature sum comparison over 80 rendered frames. One such frame is shown in Figure 2.*

5 Conclusions and Future Work

Clothing animation for fitted garments, with fine-scale details and wrinkles, remains as a difficult challenge. In this paper, we have for the first time brought this problem into the interactive realm, going from minutes per frame for high-resolution physical simulation, to several frames per second with our method in graphics hardware. Our technique combines a data-driven approach, based on interpolating a precomputed wrinkle database, with coarse cloth simulation for overall dynamics. This method is complementary to cloth simulation algorithms, and future advances in that area will benefit both precomputation and online phases of our technique. The same precomputed wrinkle database can be used for general human motions of the same character and garments, making our system of great potential interest in interactive applications like video games.

Future work can focus on many directions. The wrinkle database can be expanded to model the effects of multiple joints for greater accuracy, rather than processing each joint independently. Model reduction could be used to enable compactness and develop better wrinkle blending techniques, if more training data is available. Illumination effects like ambient occlusion or spherical harmonic lighting can also be precomputed, so that both animation and realistic rendering can be done interactively in hardware. Captured clothing data can be used, rather than precomputed fine-scale simulations. We also wish to study retargetting the same wrinkle database to different human characters and clothing designs. Finally, we believe there are a number of other problems that can benefit from tight coupling of data-driven methods to physical simulation.

Acknowledgments

We thank Michael Tao for extensive last minute video editing help, and the members of the Berkeley Graphics Group for their helpful suggestions. The reviewers deserve special mention for their thorough reading of the paper, and pointing out several references we had missed. This work was supported in part by NSF Awards IIS-0924968, IIS-0915462, ONR YIP Award N00014-10-1-0032, California Discovery Grant COM09S-156646, UC Lab Fees Research Program Grant 09-LR-01-118889-OBRJ, and by generous support and equipment donations from Intel, NVIDIA, Pixar, Adobe and Autodesk.

References

Allen, B., Curless, B., and Popovic, Z. 2002. Articulated body deformation from range scan data. In *Proc. of ACM SIGGRAPH 2002*, vol. 21, 612–619.

Baraff, D., and Witkin, A. 1998. Large steps in cloth simulation. In *Proc. of ACM SIGGRAPH 1998*, 43–54.

Baran, I., Vlasic, D., Grinspun, E., and Popović, J. 2009. Semantic deformation transfer. In *Proc. of ACM SIGGRAPH 2009*, vol. 28, 1–6.

Bordes, J. P., Maher, M., and Sechrest, M. 2009. Nvidia apex: High definition physics with clothing and vegetation. In *Game Developers Conference*.

Bridson, R., Fedkiw, R., and Anderson, J. 2002. Robust treatment of collisions, contact and friction for cloth animation. In *Proc. of ACM SIGGRAPH 2002*, vol. 21, 594–603.

Bridson, R., Marino, S., and Fedkiw, R. 2003. Simulation of clothing with folds and wrinkles. In *Proc. of SCA 2003*, 28–36.

Capell, S., Green, S., Curless, B., Duchamp, T., and Popović, Z. 2002. Interactive skeleton-driven dynamic deformations. In *Proc. of ACM SIGGRAPH 2002*, vol. 21, 586–593.

Choi, K.-J., and Ko, H.-S. 2002. Stable but responsive cloth. In *Proc. of ACM SIGGRAPH 2002*, vol. 21, 604–611.

Choi, K., and Ko, H. 2005. Research problems in clothing simulation. *Computer Aided Design 37*, 585–592.

Cordier, F., and Magnenat-Thalmann, N. 2005. A data-driven approach for real-time clothes simulation. *Comput. Graph. Forum 24*, 2, 173–183.

Cutler, L. D., Gershbein, R., Wang, X. C., Curtis, C., Maigret, E., Prasso, L., and Farson, P. 2007. An art-directed wrinkle system for CG character clothing and skin. *Graphical Models 69*, 5-6, 219–230.

Decaudin, P., Julius, D., Wither, J., Boissieux, L., Sheffer, A., and Cani, M.-P. 2006. Virtual garments: A fully geometric approach for clothing design. In *Proc. of Eurographics 2006*, vol. 25, 625–634.

Eibner, G., Fuhrmann, A. L., and Purgathofer, W. 2009. Generating predictable and convincing folds for leather seat design. In *Proc. of Spring Conference on Computer Graphics*, 93–96.

English, E., and Bridson, R. 2008. Animating developable surfaces using nonconforming elements. In *Proc. of ACM SIGGRAPH 2008*, vol. 27, 1–5.

Goldenthal, R., Harmon, D., Fattal, R., Bercovier, M., and Grinspun, E. 2007. Efficient Simulation of Inextensible Cloth. In *Proc. of ACM SIGGRAPH 2007*, vol. 26, 49.

Hadap, S., Bangerter, E., Volino, P., and Magnenat-Thalmann, N. 1999. Animating wrinkles on clothes. In *Proc. of the 10th IEEE Visualization 1999 Conference*, IEEE Computer Society.

House, D., and Breen, D. 2000. *Cloth Modeling and Animation*. AK Peters.

Jakobsen, T. 2001. Advanced character physics. In *Game Developers Conference*.

James, D. L., and Fatahalian, K. 2003. Precomputing interactive dynamic deformable scenes. In *Proc. of ACM SIGGRAPH 2003*, vol. 22, 879–887.

James, D. L., Twigg, C. D., Cove, A., and Wang, R. Y. 2007. Mesh ensemble motion graphs: Data-driven mesh animation with constraints. *ACM Trans. Graph. 26*, 4, 17.

Kaldor, J. M., James, D. L., and Marschner, S. 2008. Simulating knitted cloth at the yarn level. In *Proc. of ACM SIGGRAPH 2008*, 1–9.

Kang, Y.-M., Choi, J.-H., Cho, H.-G., and Lee, D.-H. 2001. An efficient animation of wrinkled cloth with approximate implicit integration. *The Visual Computer 17*, 3, 147–157.

Kavan, L., Sloan, P.-P., and O'Sullivan, C. 2010. Fast and efficient skinning of animated meshes. *Comput. Graph. Forum 29*, 2.

Kim, T.-Y., and Vendrovsky, E. 2008. Drivenshape: a data-driven approach for shape deformation. In *Proc. of SCA 2008*, 49–55.

Kry, P. G., James, D. L., and Pai, D. K. 2002. Eigenskin: Real time large deformation character skinning in hardware. In *Proc. of SCA 2002*, 153–159.

Larboulette, C., and Cani, M.-P. 2004. Real-time dynamic wrinkles. In *Proc. of Computer Graphics International*, 522–525.

Lee, M. 2006. Seven ways to skin a mesh: Character skinning revisited for modern GPUs. In *Proc. of GameFest, Microsoft Game Technology Conference*.

Lewis, J. P., Cordner, M., and Fong, N. 2000. Pose space deformation: a unified approach to shape interpolation and skeleton-driven deformation. In *Proc. of ACM SIGGRAPH 2000*, 165–172.

Müller, M. 2008. Hierarchical position based dynamics. In *Proc. of Virtual Reality Interactions and Physical Simulations*.

Meggs, A. 2005. Parachute pants and denim dresses: Taking real-time cloth beyond curtain. In *Game Developers Conference*.

Mohr, A., and Gleicher, M. 2003. Building efficient, accurate character skins from examples. In *Proc. of ACM SIGGRAPH 2003*, vol. 22, 562–568.

Nealen, A., Mueller, M., Keiser, R., Boxerman, E., and Carlson, M. 2006. Physically based deformable models in computer graphics. *Computer Graphics Forum 25*, 4, 809–836.

Park, S. I., and Hodgins, J. K. 2008. Data-driven modeling of skin and muscle deformation. In *Proc. of ACM SIGGRAPH 2008*, vol. 27, 1–6.

Popa, T., Zhou, Q., Bradley, D., Kraevoy, V., Fu, H., Sheffer, A., and Heidrich, W. 2009. Wrinkling captured garments using space-time data-driven deformation. In *Proc. of Eurographics 2009*, vol. 28.

Provot, X. 1995. Deformation constraints in a mass-spring model to describe rigid cloth behavior. In *Graphics Interface '95*, 147–154.

Selle, A., Su, J., Irving, G., and Fedkiw, R. 2009. Robust high-resolution cloth using parallelism, history-based collisions, and accurate friction. *IEEE Transactions on Visualization and Computer Graphics 15*, 2, 339–350.

Shi, X., Zhou, K., Tong, Y., Desbrun, M., Bao, H., and Guo, B. 2008. Example-based dynamic skinning in real time. In *Proc. of ACM SIGGRAPH 2008*, vol. 27, 1–8.

Sumner, R. W., and Popović, J. 2004. Deformation transfer for triangle meshes. In *Proc. of ACM SIGGRAPH 2004*, 399–405.

Thomaszewski, B., Pabst, S., and Strasser, W. 2009. Continuum-based strain limiting. In *Proc. of Eurographics 2009*, vol. 28, 569–576.

Vasilakis, A., and Fudos, I. 2009. Skeleton-based rigid skinning for character animation. In *Proc. of the Fourth International Conference on Computer Graphics Theory and Applications*, 302–308.

Volino, P., Magnenat-Thalmann, N., and Faure, F. 2009. A simple approach to nonlinear tensile stiffness for accurate cloth simulation. *ACM Trans. Graph. 28*, 4, 1–16.

Wang, X. C., and Phillips, C. 2002. Multi-weight enveloping: least-squares approximation techniques for skin animation. In *Proc. of SCA 2002*, 129–138.

Weber, O., Sorkine, O., Lipman, Y., and Gotsman, C. 2007. Context-aware skeletal shape deformation. In *Proc. of Eurographics 2007*, vol. 26.

White, R., Crane, K., and Forsyth, D. A. 2007. Data driven cloth animation. In *ACM SIGGRAPH 2007 sketches*, 37.