Appears in ACM Transactions on Graphics (Special Issue for SIGGRAPH 2010)

A Deformation Transformer for Real-Time Cloth Animation

Wei-Wen Feng Yizhou Yu Byung-Uck Kim

University of Illinois at Urbana-Champaign

Abstract

Achieving interactive performance in cloth animation has significant implications in computer games and other interactive graphics applications. Although much progress has been made, it is still much desired to have real-time high-quality results that well preserve dynamic folds and wrinkles. In this paper, we introduce a hybrid method for real-time cloth animation. It relies on data-driven models to capture the relationship between cloth deformations at two resolutions. Such data-driven models are responsible for transforming low-quality simulated deformations at the low resolution into high-resolution cloth deformations with dynamically introduced fine details. Our data-driven transformation is trained using rotation invariant quantities extracted from the cloth models, and is independent of the simulation technique chosen for the lower resolution model. We have also developed a fast collision detection and handling scheme based on dynamically transformed bounding volumes. All the components in our algorithm can be efficiently implemented on programmable graphics hardware to achieve an overall real-time performance on high-resolution cloth models.

CR Categories: I.3.7 [**Computer Graphics**]: Three Dimensional Graphics and Realism—Animation

Keywords: Deformation Transform, Regression, Skinning, Collision

1 Introduction

Achieving interactive performance in cloth animation has significant implications in computer games, online fashion shows and other interactive graphics applications. Indeed, much progress has been made to achieve a reasonable quality in real time. Nevertheless, it is still much desired to have real-time high-quality results that better preserve dynamic folds and wrinkles.

It is challenging to achieve this goal for the following reasons. First, physically based cloth simulation involves two expensive steps, PDE integration and collision handling. Even the latest GPUs have a hard time performing both tasks in real time on high resolution cloth models. Second, data-driven model reduction can be potentially applied to improve cloth simulation performance. However, unlike elastic materials and fluids, cloth primarily has secondary motion driven by collisions against an animated body. It is not obvious how model reduction could accelerate collision detection and handling for high resolution cloth models.

In this paper, we explore a different approach that tries to decouple the spatial dimensions from the temporal dimension. The temporal dimension is in charge of dynamics and the two spatial dimensions provide a domain to define spatially varying details, such as folds, over the cloth surface. Spatial details across a cloth surface are highly correlated. For example, at least dozens of vertices over a high-resolution cloth need to move together in a coherent way to create a single fold. This means it is possible to generate all the spatial details from a lower dimensional space.

Figure 1: Cloth animations generated with our method.

Based on this observation, we introduce a hybrid method for real-time cloth animation. The dynamics of a cloth model is generated by a low-resolution physically based simulation, and the high-resolution spatial details over the cloth surface are generated by a data-driven model from a low-dimensional space. We integrate the data-driven model with the dynamics model to produce complete high-quality cloth animations. This integration is achieved by training additional data-driven models that accurately capture the relationship between simulated coarse deformations and data-driven high-resolution spatial details. Such data-driven models transform simulated deformations at the lower resolution into high-resolution cloth deformations with dynamically introduced fine details. One of the major contributions of this paper is the identification and extraction of rotation invariant quantities that are well suited for training high-quality data-driven models.

We have developed a new animation pipeline to make the aforementioned hybrid cloth animation possible. Each time step starts with a one-step simulation of a low-resolution cloth, followed by deformation transformation and collision handling, and finally ends with high-resolution cloth surface reconstruction and rendering. The most important part is deformation transformation, which includes two nonlinear mappings that are respectively responsible for mid-scale and fine-scale deformations in the high-resolution model. To maintain a low collision detection and handling cost, we have developed a fast and effective scheme based on dynamically transformed bounding volumes. Every step of our run-time stage has been carefully designed to fully utilize the parallel processing power of modern GPUs. As a result, we achieve hundreds of frames per second when generating a high quality cloth animation from a synchronous coarse simulation.

2 Related Work

Data Driven Deformation. Skeleton subspace deformations (SSD) attach a skin to bones and each vertex of the skin is deformed according to a weighted sum of nearby bone transformations. It has been generalized to prevent potential artifacts [Lewis et al. 2000; Wang et al. 2007]. Meanwhile, fully automatic techniques have been developed to compute proxy bones and their influence weights from existing mesh animations [James and Twigg 2005].

Data-driven approaches have been able to produce realistic and

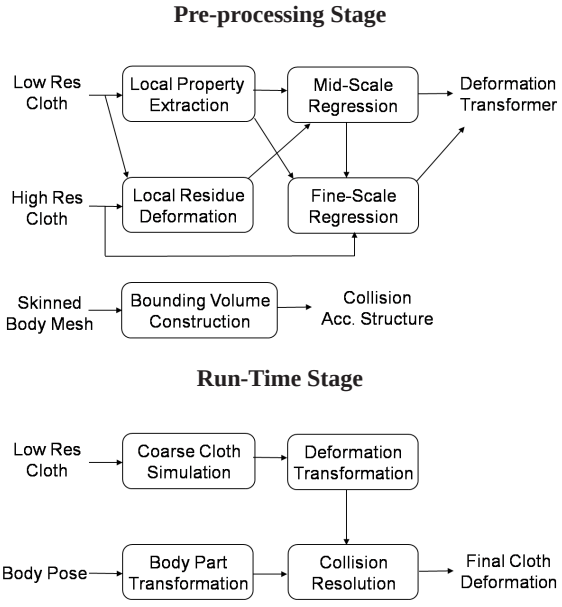


Figure 2: The workflow of our method.

detailed results. Anguelov *et al.* [2005] built a data-driven model of both the shape and deformation of full body surfaces to generate novel subjects animated with novel motion. Park and Hodgins [2006] performed motion capture of skin deformation with a large number of markers. High quality skin deformations are reconstructed via a second-order skinning scheme followed by interpolation of the residual errors. Marker-based techniques for facial motion capture and synthesis have been reported in [Bickel et al. 2007; Ma et al. 2008]. Detailed facial geometry, such as wrinkles, can be successfully synthesized from local deformation. Feng *et al.* [2008] introduced an example-based regression model for static surface deformation driven by sparse control points. This paper uses the same regression method as in [Feng et al. 2008] but in a different context for medium to fine-scale cloth deformation transformation. [Feng et al. 2008] only deals with mid-scale bone transformations. Thus it cannot produce detailed wrinkles without an excessive number of bones. More importantly, unlike [Feng et al. 2008], this paper relies on carefully designed rotation-invariant quantities to achieve superior regression results.

Producing high-quality dynamic data either via simulation or from a motion capture setup is a costly and time-consuming process. Much research has been performed to obtain dynamical models from existing data. This is typically achieved using dimension reduction and model fitting techniques [Barbič and James 2005; Park and Hodgins 2008]. Barbič and James [2005] derive a reduced dynamical model from an original one by applying dimension reduction to the state vectors while Park and Hodgins [2008] adopt an empirical dynamical model for reduced dimensions and solve its parameters by fitting the model to motion capture data.

Interactive Cloth Simulation. Extensive research has been performed on cloth simulation [Baraff and Witkin 1998; Choi and Ko 2002; Bridson et al. 2003; Goldenthal et al. 2007]. Much effort has also been devoted to making cloth simulation reach interactive performance. A common approach to faster cloth simulation replaces large in-plane forces with constraints [Fuhrmann et al. 2003; Müller et al. 2007; Goldenthal et al. 2007] and iteratively solve the system of governing equations. Physics-based geoemtric subdivision methods are applied in [Tsiknis 2004] and [Oshita and Makinouchi 2001] to produce additional folds in a coarse cloth mesh. These methods can generate more interesting high resolution cloth than applying traditional geometric subdivision. However, it is hard to produce many wrinkles within one coarse triangle as the subdivision is driven by minimizing membrane energy or enforcing the edge length constraint.

Cordier *et al.* [Cordier and Magnenat-Thalmann 2002; Cordier and Magnenat-Thalmann 2004] developed real-time techniques for animating clothes dressed on virtual characters. Their data-driven technique uses pre-simulated cloth animations as examples to correct the simulation run in real time on a much coarsened cloth mesh. Vertex positions or local cloth details are interpolated from the examples with the closest neighborhood configurations. In comparison, our technique does not need to look up closest neighborhoods. It is based on an advanced kernel regression method that takes a global configuration of the coarse cloth as the input. Results from our method have better visual quality. Stumpp *et al.* [2008] proposed a shape-matching cloth model which can plausibly reproduce folds and wrinkles from a physically based simulation. A GPU-based implementation of a finite element method [Etzmuß et al. 2003] has also been reported in [Rodriguez-Navarro and Susin 2006]. An interactive performance up to 30fps on a cloth with 10K vertices has been reported using these techniques.

3 Overview

We introduce a data-driven framework for transforming low-resolution cloth simulations into higher resolution cloth animations in real time. Unlike deformation transfer [Sumner and Popović 2004] and motion retargetting [Gleicher 1998] where the source animation has a sufficient quality and the goal is to integrate the source animation with a new hosting model, the source animation in our framework may have a low quality and our goal is to transform it to a higher-quality one. This is made possible by including high-quality high-resolution cloth deformations as part of the training data. The workflow in our method, including both a preprocessing stage and a run-time stage, is summarized in Figure 2.

Preprocessing Stage The input to the preprocessing stage is a set of n triplets, $\{(A^i, H^i, D^i)\}_{i=1}^n$, where A^i is a skinned character model, H^i is a deformed high-resolution cloth model for the character, and D^i is a corresponding deformed low-resolution cloth model. In this paper, these three types of models are all represented as triangle meshes, and meshes with the same type but a differing superscript share the same topology. Our goal is to train nonlinear mappings that can approximately transform every D^i to its corresponding H^i. Note that the underlying mechanisms for generating D^i and H^i may be different. For example, one could choose a fast low-quality simulator to generate D^i, and a slower, more expensive simulator to generate H^i.

We represent the deformation in the high-resolution model H^i at two different scales, a mid scale and a fine scale. Mid-scale deformations treat the cloth surface as a set of local patches smoothly joined together. Internal variations of the patches, such as small folds and wrinkles, are modeled as fine-scale deformations. Mid-scale deformations are represented using a skinning model while fine scale deformations are represented as residual vectors at individual vertices. We perform regressions to obtain two separate mappings for these two types of deformations. These two mappings are collectively called a *deformation transformer*. Our training procedure for the mappings is independent of the simulation technique chosen for the low-resolution model.

To accelerate collision handling at the run time, we also preprocess the cloth and character model pairs (H^i, A^i) and build a two-level bounding volume structure for collision detection.

Run-Time Stage During the run-time stage, a coarse cloth model is simulated in real time. At every frame, local surface properties of the deformed coarse model are used as the input to the trained

nonlinear mappings. Once mid-scale and fine-scale deformations have been generated from the mappings, we perform collision detection against the character model and further update the cloth deformation to resolve the collisions. The final deformation is used to reconstruct a high-resolution cloth surface for the current frame. Every step of our run-time process has been designed to fully utilize the parallel processing power of modern GPUs.

4 Cloth Deformation Transformation

Since a cloth is a highly deformable surface, the deformation transformer needs to be carefully designed to capture its intricate movements and folds. Moreover, a cloth can easily have multiple collisions with the animated character. How to resolve these collisions efficiently also affects our choice of representing cloth deformations.

4.1 Mid-Scale Deformations

A physically based cloth simulation usually needs to model a cloth using thousands of vertices to generate interesting animations. However, once fine-scale details have been removed, the remaining cloth deformation becomes more spatially coherent within local regions. Therefore, we can choose to represent mid-scale cloth deformations using linear blend skinning with a set of proxy bones [James and Twigg 2005] each of which represents a local region. Note that these proxy bones are solely used for cloth deformation and are independent of the character's skeleton. Because cloth is mostly inextensible and its deformation is mostly local bending, the deformation of proxy bones is restricted to rigid transformations with no scaling.

We need to decide next which quantity regression should be performed on. Although it is tempting to perform regression directly on global bone transformations, there exist several drawbacks. First, a global transformation is not rotation invariant. The trained mapping needs to produce distinct results for rotated versions of the same deformation, which makes the training stage much harder. Second, the dynamic range of absolute rotation and translation of a proxy bone could be very large, which makes it hard for our mapping to accurately predict them. Finally, any inaccuracies in the mapped global transformations directly affect the resulting cloth geometry, leading to obvious visual artifacts. To resolve these issues, we observe that although the low-resolution cloth simulation does not produce high-quality deformations, it does provide us with a simple overall shape of the deformed cloth. Therefore, we choose to use the cloth geometry in the low-resolution simulation to obtain a first-order approximation of every bone transformation in the high-resolution cloth, and then perform nonlinear regression to estimate the residual transformation of this first-order approximation.

Given n example deformations of the high-resolution cloth mesh, $\{H^i\}_{i=1}^n$, we partition the mesh model into a set of m local patches P_k, $k = 1..m$, using a face clustering algorithm similar to hierarchical clustering in [Wang et al. 2007] (Figure 3). We set each patch P_k as a proxy bone and extract its corresponding rotation, R_k^i, from the deformed model H^i. This rotation is defined with respect to the corresponding patch in the rest-pose high-resolution mesh. We also extract a rotation G_t^i for each triangle f_t in the deformed coarse mesh D^i. G_t^i is defined with respect to the triangle corresponding to f_t in the rest-pose coarse mesh.

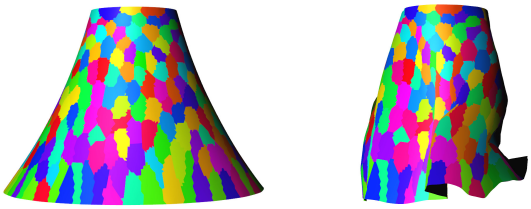


Figure 3: Proxy bones are used to model mid-scale cloth deformations. (Left) Bone clusters visualized on a rest-pose cloth model. (Right) Same bone clusters shown on a deformed cloth model. Note that cluster boundaries roughly follow cloth folds.

The first-order approximation of R_k^i is formulated as $\bar{T}_k^i = \sum_j w_t^k G_t^i$, where w_t^k's are linear blending coefficients for patch P_k and can be obtained by minimizing the following least-squares objective function, $\sum_{i=1}^n \| \sum_t w_t^k G_t^i - R_k^i \|^2$. Since $\bar{T}_k^i$ is not necessarily a rotation, we apply the polar decomposition to extract the rotation matrix $\bar{R}_k^i$ from $\bar{T}_k^i$. The residual transformation $\tilde{R}_k^i$ is the difference between $\bar{R}_k^i$ and R_k^i, defined as $\tilde{R}_k^i = (\bar{R}_k^i)^{-1} R_k^i$. It is advantageous to perform a regression on $\tilde{R}_k^i$ than R_k^i because it is local, rotation invariant, and encodes intrinsic differences between coarse and high resolution cloth deformations. In practice, we use the exponential form of a rotation and perform regression on the logarithm of $\tilde{R}_k^i$, which becomes the axis-angle representation of a rotation. This is because the axis-angle representation is less ambiguous than a quaternion and also achieves more accurate regression results. We demonstrate the benefit of performing regression on residual transformations in Figure 4. A mapping directly trained on global transformations tends to produce highly distorted deformations because of the ambiguity and a large fitting error. Note that the mapping obtained in [Feng et al. 2008] also predicts global transformations. Therefore it would suffer from the same type of problems.

Since we have chosen to use rotation-invariant quantities from the high-resolution model, we need to use rotation-invariant properties of the coarse cloth as well during regression. There are a few possible choices available, including linear rotation invariant coordinates (LRI) [Lipman et al. 2005] and local connection maps [Kircher and Garland 2008]. In our system, we choose dihedral angles between adjacent triangle pairs because it is both compact and rotation invariant. Using all dihedral angles from each example can capture static deformations, but may still miss the dynamic aspect of a cloth simulation. Therefore we use the complete set of dihedral angles, $\boldsymbol{\Theta}$, and their velocity, $\dot{\boldsymbol{\Theta}}$, during regression. In our experiments, we observed 10% to 20% improvement in numerical accuracy when using $\dot{\boldsymbol{\Theta}}$ during regression. Figure 5 shows a comparison of deformation results with and without using velocity of dihedral angles during regression. Although the visual differences are not significant, it is still desired to include $\dot{\boldsymbol{\Theta}}$ in the regression to better distinguish ambiguous examples and improve the accuracy.

We seek a mapping $g_k(\boldsymbol{\Theta}, \dot{\boldsymbol{\Theta}}) \rightarrow \omega_k$ for each P_k, where ω_k is the three-dimensional axis-angle form of the residual rotation of P_k. The regression method we use to train the mapping is simi-

Figure 4: A comparison of cloth deformations using global and rotation invariant bone transformations as regression targets. Global bone transformations are more difficult to generalize in the training stage and produce obvious artifacts in the resulting cloth deformations.

With Velocity Without Velocity

Figure 5: A comparison of transformed deformations with and without using velocity of dihedral angles during regression.

lar to the one in [Feng et al. 2008]. Since the mapping is likely to be nonlinear, to adequately account for the nonlinearity while avoiding overfitting, we perform nonlinear dimension reduction on $(\mathbf{\Theta}, \dot{\mathbf{\Theta}})$ by applying kernel canonical correlation analysis [Melzer et al. 2003] between $(\mathbf{\Theta}, \dot{\mathbf{\Theta}})$ and ω_k. Since our regression target is the residual rotation with an axis-angle representation, which is a 3D vector, the reduced dimension is set to three. This dimension reduction is followed by a linear regression between ω_k and the reduced coordinates for $(\mathbf{\Theta}, \dot{\mathbf{\Theta}})$. At run time, given a novel coarse cloth mesh, our mapping predicts the residual rotation $\tilde{R}_k$ and combines that with $\bar{R}_k$ to reconstruct R_k. Since $\tilde{R}_k$ is a local rotation, it does not have a bone translation necessary for high-resolution cloth surface reconstruction. We compute bone translations by solving a coarse-grain Poisson equation in real time as in [Feng et al. 2008].

4.2 Fine-Scale Deformations

Detailed folds and wrinkles are difficult to model with proxy bone transformations since they are high-frequency features that may change rapidly even within a local patch. Increasing the number of proxy bones may alleviate this problem, but using too many proxy bones would affect the overall performance and defeat our original purpose for real-time applications. Thus we revise the eigenskin method [Kry et al. 2002] to represent these high-frequency details, and then train a second mapping for predicting such details. The original eigenskin technique was proposed to model details in articulated skin deformation that cannot be captured by proxy bones. It partitions a mesh into a set of influence regions associated with joints and performs dimension reduction on their residue offsets. However, in our problem, these influence regions are not well-defined since a cloth model has its own dynamics and is not tightly coupled to joints in the articulated character.

To adapt the eigenskin technique to solve our problem, we need to define suitable criteria for partitioning the cloth surface into local regions. We notice that although cloth wrinkles tend to be quite complicated, wrinkle offsets at individual vertices are usually correlated with each other. That is, offset patterns at one vertex may give rise to similar offset patterns at some other vertices. Therefore our goal is to identify regions that have coherent vertex offsets throughout the training examples. We propose a clustering scheme based on both hierarchical clustering and clustered principal component analysis (CPCA) [Sloan et al. 2003] to find the coherent regions.

Given n deformed high-resolution cloth meshes, $\{H^i\}_{i=1}^n$, as training examples, we represent the fine-scale deformation at vertex v_j on mesh H^i as a 3D displacement vector u_j^i on the rest-pose mesh. Let the original and skinned position of v_j on mesh H^i be v_j^i and $\bar{v}_j^i$, respectively. According to linear blend skinning, $\bar{v}_j^i = \sum_k w_k^j M_k^i v_j^0$ where v_j^0 is the rest-pose position of v_j, M_k^i is the transformation of bone P_k at mesh H^i, and w_k^j represents the influence weight of bone P_k on vertex v_j. Thus, $u_j^i = (\sum_k w_k^j M_k^i)^{-1} v_j^i - v_j^0$. The set of displacement vectors at the same vertex but across all n training examples, $\{u_j^i\}_{i=1}^n$, are concatenated to form a $3n$-dimensional *per-vertex residual vector*. This residual vector represents per-vertex displacement vectors across all training examples instead of across different vertices.

At the beginning, our clustering algorithm sets up a face cluster Γ_s for each triangle f_s. It performs hierarchical merging on these clusters afterwards. A PCA basis is computed for all the per-vertex residual vectors within a cluster. We can determine whether a per-vertex residual vector outside the cluster can be well represented by the PCA basis of the cluster by computing the projection error of the residual vector. The error induced by merging cluster Γ_{l_a} to Γ_{l_b} is therefore defined as

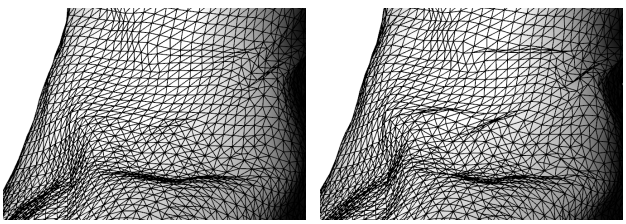
$$\sum_i \sum_{j \in \Gamma_{l_a}} \|u_j^i - \tilde{u}_j^i\|,$$

where $\tilde{u}_j^i$ is the approximation of per-vertex displacement u_j^i in cluster Γ_{l_a} by the PCA basis of cluster Γ_{l_b}. This error metric tends to merge clusters with similar per-vertex displacements across all training examples. We greedily merge the cluster pair with the lowest merging error. Whenever two clusters are merged, the PCA basis is recomputed from all the per-vertex residual vectors in the merged cluster. This process is repeated until a desired number of clusters has been reached. Each of the final clusters represents a region with similar per-vertex residual vectors.

Now let us define a *per-example residual vector*, Υ_l^i, for a final cluster Γ_l on mesh H^i by concatenating all 3D per-vertex displacement vectors in $\{u_j^i\}_{j \in \Gamma_l}$. The per-example residual vector is $3m$-dimensional if cluster Γ_l has m vertices. A PCA basis is also computed for all the per-example residual vectors corresponding to the same final cluster. For example, there is a PCA basis for the set of residual vectors, $\{\Upsilon_l^i\}_{i=1}^n$. This new PCA basis plays a similar role as the eigenskin basis in [Kry et al. 2002] and will be used for reconstructing wrinkle patterns within a cluster from given PCA coefficients.

With this new PCA basis for each final cluster Γ_l, we train a second mapping, $h_l(\mathbf{\Theta}, \dot{\mathbf{\Theta}}) \rightarrow c_l$, for fine-scale deformations in Γ_l. Here c_l represents the PCA coefficients of a per-example residual vector from Γ_l. As in the previous section, the mapping is also trained using both kernel canonical correlation analysis and linear regression. The advantage of face clustering before regression is that we can obtain local mappings for the fine-scale details within each region, instead of a global mapping for details over the entire surface. As shown in Figure 6, a global mapping may miss certain fine-scale details, while a set of local mappings produce deeper folds and wrinkles. Note that the total size of PCA basis vectors are identical between a global PCA and a clustered PCA.

At run time, the residual vector within every cluster is reconstructed from predicted PCA coefficients to produce detailed folds over the high-resolution cloth surface. We show the benefits of adding fine-scale deformations in Figure 7. The deformation results with our eigenskin-based mapping have more detailed folds that cannot be captured by bone transformations alone.



Without Clustering With Clustering

Figure 6: A comparison of cloth deformations with and without face clustering in the modified eigenskin technique.

Mid-Scale Only Mid+Fine Scale

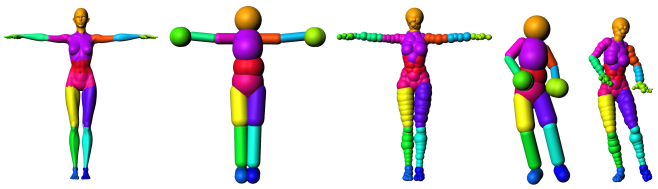
Figure 7: A comparison of deformation transformation results with and without fine-scale deformations. Mid-scale results produce a relatively smooth cloth surface while more interesting folds are generated by fine-scale deformation transformation.

5 Collision Detection and Handling

Collision detection and handling is also a critical and expensive step for cloth simulation over an articulated body. Accelerated collision detection is usually achieved via bounding volume hierarchies (BVH) set up on both the cloth and articulated body. However, for a deforming cloth, it would be too expensive to rebuild its BVH at every time step. We need to develop an efficient collision detection and resolution scheme that can easily utilize the parallel processing power of GPUs for our real-time cloth animation.

We propose a collision detection scheme between the high-resolution cloth model and the body mesh using the proxy bones originally for mid-scale cloth deformations. We attach a bounding volume to every proxy bone at its rest pose. When the cloth deforms, these proxy bones follow rigid transformations. Therefore, we can update the position and orientation of each bounding volume very quickly using the rigid transformation of the bone it is attached to. The bounding volumes of the articulated body can be updated similarly by attaching a bounding volume to every body part. Although this collision scheme is slightly less accurate, its main advantage is that the number of collision tests can be significantly reduced because collision can now be tested against proxy bones instead of mesh triangles.

There are many types of bounding volumes available, and we choose the most suitable ones according to the shape of proxy bones and body parts. The shape of a local cloth patch represented by a proxy bone is similar to a thin plate. We approximate it with a lozenge L_k [Larsen et al. 2000] by sweeping a sphere within a rectangle. Since a body part of an articulated character has an elongated shape, it can be more tightly enclosed by an elongated bounding volume, a bounding capsule C_k, whose shape can be obtained by sweeping a sphere along a line segment.



(A) (B) (C) (D)

Figure 8: An overview of our two-level bounding volume hierarchy. Given a skinned body mesh in (A), with every bone cluster rendered in a different color, we construct a bounding capsule in (B) for every bone cluster. We also build a sphere set approximation for the body mesh and assign each sphere to a corresponding bone cluster. The bounding capsules and spheres can be transformed on the fly according to body movements, as shown in (D).

However, a single bounding volume for each body part is usually not sufficiently accurate for collision detection and resolution. To improve the collision accuracy while maintaining a high level of efficiency, we use a set of spheres, S_{C_k}, to build a second more accurate level of shape approximation for each body part [Bradshaw and O'Sullivan 2002], as shown in Figure 8. We spatially index the bounding spheres according to their relative position to the bounding capsule. Since a capsule can be regarded as a cylinder joined with a hemisphere at each end, we can divide the middle cylindrical surface into small regions using cylindrical coordinates and divide the two hemispheres into small regions using polar coordinates. A spatial index table can be set up for the second-level spheres by registering every sphere with the overlapping regions on the capsule. This precomputed index table ensures a constant number of collision tests against each body part, which makes collision detection suitable for parallel processing on the GPU. We further notice that a local patch P_k on the cloth usually only intersects with a few nearby body parts throughout a simulation because the cloth is wrapped around the body and a cloth patch cannot move around freely to reach every body part. Therefore, to further reduce the number of collision tests, we associate a fixed number of nearest bounding capsules, C_{P_k}, with each cloth patch P_k by computing the sum of distances between body parts and P_k in all training examples.

The detailed steps of collision detection and resolution are summarized as follows:

1. At each time step, transforming all capsules (C_k) and lozenges (L_k) using their corresponding bone transformations.
2. For each L_k, perform collision test against the set of capsules C_{P_k} and find the one, C_{k_d}, with the deepest penetration. Locate the intersection point, p_{int}, on the surface of C_{k_d}.
3. Use p_{int} and the index table on C_{k_d} to find out the nearby sphere set $S_{p_{int}}$. Then test intersection between L_k and $S_{p_{int}}$ to obtain a penetration depth d_{L_k}.
4. Resolve the collision by setting a new bone translation $t'_k = t_k + d_{L_k}$ for patch P_k.

The above collision detection and handling scheme works well in all our experiments. Note that a collision test between a capsule and a lozenge can be reduced to finding the shortest distance between a line segment and a rectangle in the 3D space. In our real-time simulation of the coarse cloth model, we adopt a similar collision handling process but replacing lozenges with cloth faces in the low resolution mesh. Since deformed high-resolution cloth usually has an overall shape similar to the deformed coarse cloth, which has its own collision handling, cloth-body inter-penetrations in the high-resolution model are usually modest and easy to resolve.

6 Run-Time Implementation

We carefully choose operations involved in our run-time algorithms to make sure all of them can be implemented efficiently on programmable graphics hardware to achieve real-time performance on high-resolution cloth models. The overall run-time stage has four steps: low-resolution cloth simulation, deformation transformation, collision detection and resolution, and high-resolution cloth surface reconstruction. In our implementation, we use CUDA for the first three steps, and GLSL for final surface reconstruction and rendering.

Coarse Cloth Simulation We implemented the mass-spring model using Verlet integration [Zeller 2005] in the low-quality coarse cloth simulation. Since an explicit integration scheme is used, the spring lengths need to be constrained to ensure stable integration. A few iterations of Jacobi relaxation is applied in parallel

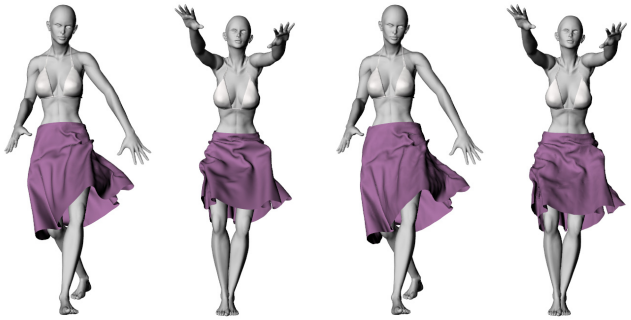


Figure 9: Comparisons between ground truth and final deformation results from our method. The left images show the ground truth, and the right images show our final deformation results.

Most Responsive Less Responsive Most Damped

Figure 10: Deformation transformation results with distinct styles. Three high-resolution cloth simulations with different material properties are used as training examples. All of them are trained with the same coarse cloth simulation. Our method can adapt to different cloth material properties in the training examples and produce distinct cloth animation styles.

to every cloth spring to enforce such length constraints [Müller et al. 2007]. Cloth-body collision detection is performed by testing every cloth triangle against the body mesh via two levels of bounding volumes as discussed in the previous section.

Deformation Transformation Once we have obtained the coarse cloth deformation, dihedral angles are extracted from the current vertex positions. $f_k(\mathbf{\Theta}, \dot{\mathbf{\Theta}})$ and $g_l(\theta, \dot{\mathbf{\Theta}})$ are used to predict residual rotations for mid-scale deformations and PCA coefficients for fine-scale offsets, respectively, in the high-resolution cloth model. This process involves matrix-vector multiplications and kernel function evaluations, which can be implemented efficiently on the GPU. Once we have obtained bone rotations, we need to solve a linear system (a Poisson equation) to obtain bone translations. The solution matrix for the linear system can be precomputed, and thus only a matrix-vector multiplication is needed at the run-time stage.

Collision Detection and Resolution Before testing collisions for the high-resolution cloth model, we transform in parallel all first-level bounding volumes according to corresponding bone transformations. We test collisions between all lozenges and their nearby body part capsules C_{P_k} in parallel. Penetration depths are also computed in parallel for every lozenge to update the corresponding bone translation.

High-Resolution Cloth Reconstruction The final cloth surface is reconstructed via eigenskin-based offsetting and matrix-palette skinning. Using the predicted PCA coefficients c_l for per-example residual vectors, the displacement u_j for vertex v_j is obtained by $u_j = V_{jl}c_l$, where V_{jl} has the components associated with u_j in the PCA basis V_l. We upload V_l as a vertex texture in advance, and reconstruct u_j in the vertex shader before skinning. The final vertex position v_j^f on the cloth surface is

$$v_j^f = \sum_k w_k^j M_k(\bar{v}_j^0 + u_j),$$

where w_k^j is a bone influence weight at v_j, and $\bar{v}_j^0$ is the rest-pose position of v_j. Here both w_k^j and $\bar{v}_j^0$ can be preloaded to the GPU memory via vertex buffer objects (VBO). Since cloth wrinkles are highly dynamic, we recompute vertex normals on the fly every frame to guarantee accurate normals for wrinkle rendering.

7 Experimental Results

We have successfully tested our deformation transformer on different types of clothing and body movements (Figure 1). We constructed low-resolution cloth models using Maya, and then generated high-resolution ones via mesh subdivision. It is also feasible to generate low-resolution cloth models by simplifying high-resolution ones. We then use Poser, a commercial software package for 3D character animation, to generate skinned character animations and their associated high resolution cloth simulations as training data. The cloth simulator in Poser is capable of simulating different types of cloth materials with different settings of simulation parameters such as cloth density, folding resistance, damping, etc. We utilize these features to produce high quality cloth animations as our training examples. For each high-resolution cloth animation, as part of the training data, we also generate a corresponding low-resolution one following the method in [Müller et al. 2007; Zeller 2005].

The number of triangles in a coarse cloth model is typically between 200 to 300 triangles. We found this resolution sufficient to capture dynamic characteristics of the cloth without much performance penalty during runtime simulation. Using more triangles in the coarse model would improve the results as the residual deformations between low-resolution cloth models and high-resolution ones would be smaller, and thus make regression easier to succeed. Since coarse cloth simulation is only a fraction of the total runtime computation, there would be only modest performance penalty as long as the number of triangles in the coarse cloth is not excessive.

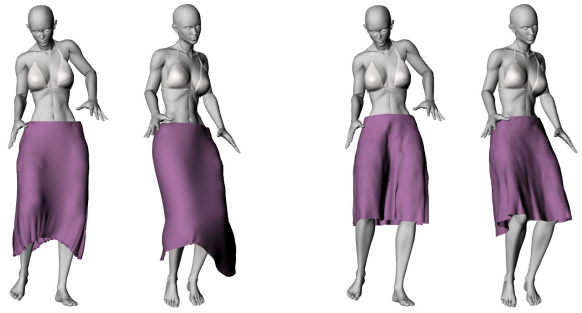
Training data from a high-resolution cloth simulation generated by Poser needs to be preprocessed and converted to our mid-scale and fine-scale deformation representations. In our experiments, we found that 300 to 400 proxy bones were suitable for capturing important mid-scale deformations in the training examples; and fine-scale details could be captured by 80 to 120 CPCA clusters with 20 basis vectors per cluster in our modified eigenskin technique (Table 1). In all our experiments, we applied the 3rd degree inhomogeneous polynomial kernel for CCA-based regression.

Our runtime stage includes both coarse cloth simulation and deformation transformation. Our runtime algorithm can achieve more than 250 frames per second on an nVidia Geforce GTX275 GPU when generating high quality cloth animations with a resolution between 25,000 and 38,000 triangles, as shown in Table 1. As can be seen, the number of triangles in our high-resolution cloth models is more than 100 times the number of triangles in our coarse cloth models.

We have performed two types of experiments to validate our method. In the first type of experiments, we choose a small fraction of the frames from a given animation as training examples and test our data-driven model on the rest of the frames. For each animation, we choose the training examples automatically by performing K-means clustering on the dihedral angles of the low-resolution cloth. Within each cluster, the frame with a configuration closest to

Cloth	Motion	# Tri	# Low Tri	Bones	#Cluster	#RotCCA	#EigPCA	#Train	#Test	Error	fps	Prep
Skirt 1	Dance 1	38,502	200	330	100	3	20	120	601	2.5%	268	13 min
Skirt 1	Jump	38,502	200	330	100	3	20	60	314	2.5%	271	11 min
Skirt 2	Dance 2	25,693	180	320	80	3	20	48	224	2.4%	280	8 min
Dress	Dance 3	27,402	220	350	120	3	20	124	600	1.8%	261	15 min
Dress	Walk	27,402	220	350	80	3	20	47	239	1.6%	266	10 min
TShirt	Dance 1	25,726	322	350	120	3	20	120	601	2.8%	251	15 min

Table 1: Statistics and Timing. All performance measurements were taken from a 3.0GHz Core 2 Duo processor with a nVidia Geforce GTX275 Graphics Processor. '#Tri' and '#Low Tri' refer to the number of triangles in the high and low resolution cloth models, respectively. '#Cluster' means the number of face clusters used for Eigenskin, '#RotCCA' means the number of CCA basis vectors used for mid-scale bone residual transformation regression, and '#EigPCA' means the number of PCA basis vectors used for representing fine-scale deformations within each cluster. '#Train' means the number of training examples, '#Test' means the number of testing examples, and 'Prep' means the total amount of time for all preprocessing steps. Error is computed using the average per-vertex error divided by the radius of the bounding sphere of the cloth.



Explicit Strain Limiting Our Method

Figure 11: A comparison between cloth animations generated from our method and the method in [Müller et al. 2007], where explicit integration with iterative strain limiting is used to produce non-stretchy cloth. Compared with explicit integration with strain limiting, our method can produce more inextensible cloth deformations similar to the training examples.

the cluster center is selected as a training example. Compared with random selection, clustering finds more representative cloth deformations in a given animation. Our trained transformer can produce high quality cloth deformations for the testing frames. A comparison between novel cloth deformations from our transformer and the ground truth are shown in Figure 9. Our transformed cloth deformations are not only visually appealing, but also numerically accurate, as shown in the "Error" column in Table 1. Our method is also capable of adapting to training examples with different cloth material properties. As shown in Figure 10, three different data-driven models were trained using examples with different material properties to show varying responsiveness and wrinkles. The trained transformers are capable of capturing intrinsic material properties from the training data and produce cloth deformations with the corresponding material properties. To further verify the generalization capability of our method, in the second types of experiments, we use sparsely chosen frames from one animation as training examples and test the trained model on a different animation with a similar style. For example, in Figure 12(A), we use frames from one dancing sequence for training and another sequence with a similar dancing style for testing. As long as the motion styles in the two sequences are reasonably similar, our method can still produce reasonably good results.

We have also compared our results with a direct simulation of high resolution cloth on the GPU using our CUDA implementation of the method in [Müller et al. 2007], which is similar to the method implemented in NVIDIA PhysX. The performance of direct simulation is about 80 FPS for the cloth model with 38K triangles. We found the major performance bottleneck was its iterative scheme for limiting spring length. While it takes only a few iterations for a coarse cloth, it may take many more iterations to generate acceptable results for a higher resolution cloth. As shown in 11, the simulated cloth still appears stretchy after 40 iterations, but its performance has already dropped to 60 FPS. Increasing the number of iterations could improve the simulation results, but would also further affect the overall performance. On the other hand, our hybrid method produces more inextensible cloth deformations with less computational cost.

8 Conclusions and Discussion

We have introduced a hybrid method for real-time cloth animation. It relies on data-driven models to capture the relationship between cloth deformations at two resolutions. Such data-driven models are responsible for transforming low-quality simulated deformations at the low resolution into high-resolution cloth deformations with dynamically introduced fine details. Our data-driven transformation is trained using rotation invariant quantities extracted from the cloth models, and is independent of the simulation technique chosen for the lower resolution model. Our method achieves hundreds of frames per second when generating a high quality cloth animation from a synchronous coarse simulation.

Limitations Like other data-driven methods, our proposed method requires a preprocessing stage where a data-driven model is trained. In addition, once deformations from a coarse simulation fall outside the deformation subspace defined by the training examples, numerical accuracy cannot be guaranteed. Nevertheless, we found experimentally that except for extreme cases, reasonable visual results could still be generated, as shown in Figure 12(A). Another limitation of our current method is that cloth self-intersections are not handled. We would like to add this capability in future using a spatial partition scheme, such as a uniform voxel grid. When two nonadjacent proxy bones on the cloth intersect with the same voxel, a potential self-intersection occurs and can be resolved by pulling the bones along their negative normal directions. Such a self-intersection resolution scheme is not expected to consume many GPU cycles.

Acknowledgments

We would like to thank Nathan Wesling for help in generating the cloth models and locating the training animations. We would also like to thank the anonymous reviewers for their valuable comments.

References

ANGUELOV, D., SRINIVASAN, P., KOLLER, D., THRUN, S., RODGERS, J., AND DAVIS, J. 2005. Scape: Shape completion and animation of people. *ACM TOG 24*, 3, 408–416.

BARAFF, D., AND WITKIN, A. 1998. Large steps in cloth simulation. In *Proc. of SIGGRAPH'98*, 43–54.

BARBIČ, J., AND JAMES, D. L. 2005. Real-time subspace integration for st. venant-kirchhoff deformable models. *ACM Trans. Graph. 24*, 3.

BICKEL, B., BOTSCH, M., ANGST, R., MATUSIK, W., OTADUY, M., PFISTER, H., AND GROSS, M. 2007. Multi-scale capture of facial geometry and motion. In *SIGGRAPH '07: ACM SIGGRAPH 2007 papers*, 33.

BRADSHAW, G., AND O'SULLIVAN, C. 2002. Sphere-tree construction using dynamic medial axis approximation. In *SCA '02: Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*, ACM, New York, NY, USA, 33–40.

BRIDSON, R., MARINO, S., AND FEDKIW, R. 2003. Simulation of clothing with folds and wrinkles. In *SCA '03: Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*.

CHOI, K.-J., AND KO, H.-S. 2002. Stable but responsive cloth. *ACM Trans. Graphics 21*, 3.

CORDIER, F., AND MAGNENAT-THALMANN, N. 2002. Real-time animation of dressed virtual humans. *Computer Graphics Forum 21*, 3, 862–870.

CORDIER, F., AND MAGNENAT-THALMANN, N. 2004. A data-driven approach for real-time clothes simulation. In *PG '04: Proceedings of the Computer Graphics and Applications, 12th Pacific Conference*.

ETZMUSS, O., KECKEISEN, M., AND STRASSER, W. 2003. A fast finite element solution for cloth modelling. In *PG '03: Proceedings of the 11th Pacific Conference on Computer Graphics and Applications*.

FENG, W.-W., KIM, B.-U., AND YU, Y. 2008. Real-time data driven deformation using kernel canonical correlation analysis. *ACM Transactions on Graphics 27*, 3, 91:1–9.

FUHRMANN, A., GROSS, C., AND LUCKAS, V. 2003. Interactive animation of cloth including self collision detection. In *WSCG '03*.

GLEICHER, M. 1998. Retargetting motion to new characters. In *SIGGRAPH 98 Proceedings*, 33–42.

GOLDENTHAL, R., HARMON, D., FATTAL, R., BERCOVIER, M., AND GRINSPUN, E. 2007. Efficient simulation of inextensible cloth. *ACM Trans. Graph. 26*, 3, 49.

(A) (B)

Figure 12: Deformation transform results by taking training examples from one sequence and testing on another. In (A), the training sequence has a similar style as the testing sequence and our method is able to produce reasonably good results. However, our method does not generalize well and give rise to artifacts and penetrations in a more extreme case shown in (B), where frames from a sequence with a very different style are used as training examples.

JAMES, D., AND TWIGG, C. 2005. Skinning mesh animations. *ACM TOG 24*, 3, 399–407.

KIRCHER, S., AND GARLAND, M. 2008. Free-form motion processing. *ACM Transactions on Graphics 27*, 2, 1–13.

KRY, P., JAMES, D., AND PAI, D. 2002. Eigenskin: Real time large deformation character skinning in hardware. In *ACM SIGGRAPH Symp. on Computer Animation*, 153–159.

LARSEN, E., GOTTSCHALK, S., LIN, M., AND MANOCHA, D. 2000. Fast distance queries with rectangular swept sphere volumes. In *IEEE International Conference on Robotics and Automation*.

LEWIS, J., CORDNER, M., AND FONG, N. 2000. Pose space deformation: A unified approach to shape interpolation and skeleton-driven deformation. In *Computer Graphics Proceedings, Annual Conference Series*, 165–172.

LIPMAN, Y., SORKINE, O., LEVIN, D., AND COHEN-OR, D. 2005. Linear rotation-invariant coordinates for meshes. *ACM Transactions on Graphics 24*, 3.

MA, W.-C., JONES, A., CHIANG, J.-Y., HAWKINS, T., FREDERIKSEN, S., PEERS, P., VUKOVIC, M., OUHYOUNG, M., AND DEBEVEC, P. 2008. Facial performance synthesis using deformation-driven polynomial displacement maps. *ACM Trans. Graph. 27*, 5.

MELZER, T., REITERA, M., AND BISCHOFB, H. 2003. Appearance models based on kernel canonical correlation analysis. *Pattern Recognition 36*, 9, 1961–1971.

MÜLLER, M., HEIDELBERGER, B., HENNIX, M., AND RATCLIFF, J. 2007. Position based dynamics. *J. Vis. Comun. Image Represent. 18*, 2.

OSHITA, M., AND MAKINOUCHI, A. 2001. Real-time cloth simulation with sparse particles and curved faces. In *Proc. of Computer Animation*, 62–83.

PARK, S., AND HODGINS, J. 2006. Capturing and animating skin deformation in human motion. *ACM Transactions on Graphics 25*, 3, 881–889.

PARK, S. I., AND HODGINS, J. K. 2008. Data-driven modeling of skin and muscle deformation. *ACM Trans. Graph. 27*, 3, 96:1–6.

RODRIGUEZ-NAVARRO, J., AND SUSIN, A. 2006. Non structured meshes for cloth gpu simulation using fem. In *Workshop on Virtual Reality Interaction and Physical Simulation*.

SLOAN, P.-P., HALL, J., HART, J., AND SNYDER, J. 2003. Clustered principal components for precomputed radiance transfer. *ACM TOG 22*, 3, 382–391.

STUMPP, T., SPILLMANN, J., BECKER, M., AND TESCHNER, M. 2008. A geometric deformation model for stable cloth simulation. In *Workshop on Virtual Reality Interaction and Physical Simulation*.

SUMNER, R., AND POPOVIĆ, J. 2004. Deformation transfer for triangle meshes. *ACM Transactions on Graphics 23*, 3, 397–403.

TSIKNIS, K. D. 2004. *Better Cloth Through Unbiased Strain Limiting and Physics-Aware Subdivision*. Master's thesis, University of British Columbia.

WANG, R., PULLI, K., AND POPOVIĆ, J. 2007. Real-time enveloping with rotational regression. *ACM Transactions on Graphics 26*, 3, 1174–1179.

ZELLER, C. 2005. Cloth simulation on the gpu. In *SIGGRAPH '05: ACM SIGGRAPH 2005 Sketches*, ACM, New York, NY, USA, 39.