Multi-layer skin simulation with adaptive constraints

Pengbo Li Paul G. Kry

School of Computer Science, McGill University

Abstract

We present an approach for physics based simulation of the wrinkling of multi-layer skin with heterogeneous material properties. Each layer of skin is simulated with an adaptive mesh, with the different layers coupled via constraints that only permit wrinkle deformation at wavelengths that match the physical properties of the multi-layer model. We use texture maps to define varying elasticity and thickness of the skin layers, and design our constraints as continuous functions, which we discretize at run time to match the changing adaptive mesh topology. In our examples, we use blend shapes to drive the bottom layer, and we present a variety of examples of simulations that demonstrate small wrinkles on top of larger wrinkles, which is a typical pattern seen on human skin. Finally, we show that our physics-based wrinkles can be used in the automatic creation of wrinkle maps, allowing the visual details of our high resolution simulations to be produced at real time speeds.

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

Keywords: physics based simulation, thin shells, heterogeneous material properties, constraints, adaptive meshing, multi-layer skin

1 Introduction

Current methods for producing computer animation of wrinkled skin and clothing can be categorized into three groups. Artists can draw wrinkle patterns in texture maps to vary the surface normal during shading calculations [Blinn 1978; Jimenez et al. 2011] and design wrinkle curves to modify mesh geometry in an art-directed manner [Cutler et al. 2007]. Alternatively, procedural approaches can produce mesh deformations that closely resemble the physical phenomenon [Rohmer et al. 2010]. In the third category are techniques that rely entirely on physics-based simulation to animate the surface mesh according to a set of assigned material properties [Rémillard and Kry 2013; Chen et al. 2013]. Regardless the technique used, wrinkles are important visual details used in the design of realistic and expressive characters, for faces and bodies.

In this paper we present a physics based approach for simulating wrinkles, and we focus specifically on skin as opposed to clothing. Whereas cloth can be simulated as a single thin shell that interacts with other shells or surfaces through contact, skin consists of layers of different materials that are *attached* to one another. Skin wrinkling occurs when a thin stiff layer buckles instead of compressing to accommodate deformation of the underlying volume; the elastic energy of the wrinkled shape is lower than when there is compression alone. With skin composed of multiple layers of thin material attached to a deformable volume, it is possible to have fine wrinkles on top of larger ones, as can be seen on human skin undergoing deformation. Figure 1 shows an example of this phenomenon. Physics based simulation automates the creation of these fine geometric details, thus the artist or animator need only design the underlying blend shapes and the material properties of the different skin layers. Because of the cost of simulating such fine details, we compute wrinkle maps from our multi-later simulation, which permits physically plausible fine wrinkles to be added to interactive applications using a normal-map fragment shader.

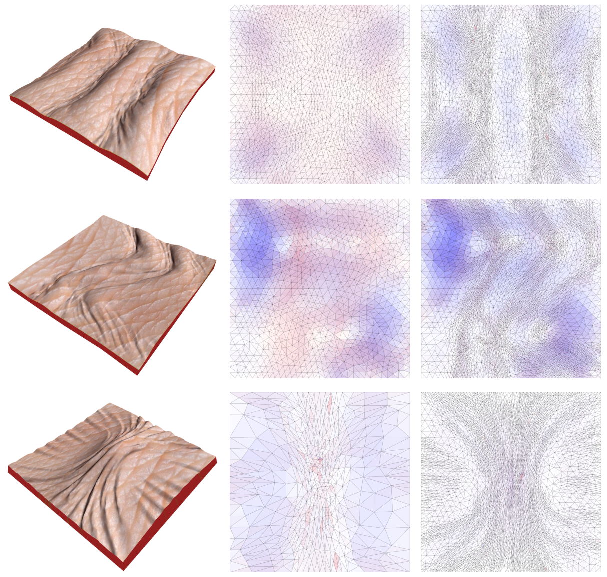
Figure 1: *An example showing wrinkles forming with a two layer skin model undergoing compressing (top), twisting (middle) and pinching (bottom). Center and right shows the adaptive meshes in material space of the two layers.*

We take inspiration from the embedded thin shells approach of Remillard and Kry [2013], which has the benefit of using only a small number of degrees of freedom in comparison to a full volumetric simulation of a thin stiff layer on top of a softer volume. In our work, we similarly construct constraints that attach the different layers, but we also deal with multiple layers and modify the constraints to account for the distance between each layer. Because we drive the underlying layer with blend shapes, the multi-layer skin contains all the degrees of freedom. While this simplifies the simulation, there is still the major challenge of simulating the very high mesh resolutions necessary for fine wrinkles. Therefore, we employ the adaptive mesh technique of Narain et al. [2012] to add degrees of freedom to the simulation only where they are needed. But adaptive meshes also require a careful modification of the constraints. Our solution is to define the constraints as continuous functions and use an adaptive sampling technique that ensures that the constraints remain satisfied after each remeshing. Furthermore, we address the simulation of heterogeneous materials, such as varying skin thickness and stiffness, both during construction of the constraints, and with a method for resampling material properties during remeshing.

Our contributions can be summarized as follows. We introduce a *multi-layer model* to approximate the layered structure of human skin. Our model takes into account *heterogeneous material properties* in the construction of the constraints between layers that allow the formation of wrinkles at frequencies predicted by the material properties. We present a method for *resampling constraints* such that they remain valid when different layers are adaptively remeshed to produce fine details. We similarly *resample material properties* for the simulation to accommodate remeshing of the different layers. Finally, we automate the creation of *physics-based wrinkle maps* for blend-shape driven simulation, which can be used to enrich real time animations with fine geometric details.

2 Related work

Physically based simulation is a critical tool used to produce computer animation that would be otherwise extremely difficult to create by artist. The problem of simulating deformation of multi-layer skin is not too different from cloth simulation, a topic which has received a vast amount of attention. Early work on modelling the dynamics of cloth used mass-spring systems [Provot 1995], while other seminal work demonstrated the use of continuum methods and implicit stepping [Baraff and Witkin 1998]. While these methods continue to be the widely-used, more recent research has addressed modeling issues [Grinspun et al. 2003; Wang et al. 2011], collision handling [Bridson et al. 2002; Harmon et al. 2008], strain limiting [Thomaszewski et al. 2009; Wang et al. 2010b], and resolving fine details such as wrinkles and folds [Bridson et al. 2003; Chen et al. 2013].

The popularity of mass spring systems is due to the ease of implementation and inexpensive computational costs. However, for higher fidelity simulations, continuum-based approaches are typically employed with finite element methods (FEM). Most of the existing FEM approaches are based on the geometrically exact thin shell formulation presented by Simo and Fox [1989]. The characteristic folding and buckling behavior of cloth highly depends on bending properties, which plays a central role in the appearance of real textiles. The models of bending forces are typically characterized into two main approaches. One is to use crossover springs that extend the surface [Provot 1995]. The other one is to build a discrete hinge model and evaluate precisely the angle between adjacent mesh elements, thus creating plausible bending forces [Bridson et al. 2003; Grinspun et al. 2003]. Furthermore, subdivision finite elements and a co-rotational strain formulation have been used to produce smooth deformations, while likely avoiding the locking artifacts that can occur with linear shape functions [Thomaszewski et al. 2006]. For the simulation of single layer skin, Remillard and Kry [2013] also use higher order shape functions for smoothness in the coarsely discretized underlying volume. However, the embedded thin shell uses linear shape function, and is coupled to the underlying volume using constraints designed to only permit deformation and the desired wrinkling frequency. This use of constraints is similar to the method of Bergou et al. [2007], which provides a mechanism for a generating fine details in a high resolution simulation that track the general shape and motion of a previously computed coarse simulation. Our work uses a similar constraint based approach, and builds on the embedded thin shell technique, but results in a more realistic multi-layer human skin model.

Wrinkles and folds are important visual features of deformable surfaces. The simulation of high resolution meshes is able to capture fine details, but it is computationally expensive. Therefore, a number of recent approaches prefer to add visual details during a post-processing step on the simulate result of a coarser model. Techniques to generate additional wrinkles on the reduced model can be divided into four main categories: procedural generation [Bando et al. 2002; Cutler et al. 2007], learning from high-resolution simulation data [Wang et al. 2010a; Kavan et al. 2011], sampling from examples of real cloth [Zhou et al. 2012], and physics simulation with simplified models [Müller and Chentanez 2010; Rohmer et al. 2010]. An alternative approach prefers to animate cloth through adaptive refinement, allowing local control of the resolution of meshes to conform to the complex shapes that arise during simulation [Narain et al. 2012; Narain et al. 2013]. This is beneficial in making a balanced trade-off between level of detail and computational cost, and thus we implement our technique using Narain's provided code for adaptive mesh simulation.

Human skin is composed of several layers, each with unique properties and functions. In our physically based simulation, we care about the mechanical behavior of these skin layers. Earlier work by Thalmann et al. [2002] establishes layered skin models based on the physical structure of epidermis and dermis, and explores the formation of folds and wrinkles on the skin. More recently, there have been significant efforts devoted to measuring the elastic properties and thickness of each layer [Gennisson et al. 2004; Geerligs 2010]. Cerda et al. [2003] build theory of thin elastic sheets offering insight into mechanism of wrinkling in human skin. We build on this model in order to identify the critical wrinkling frequency of different layers in order to build constraint functions that couple our adaptively meshed thin shell models.

3 Multi-layer skin

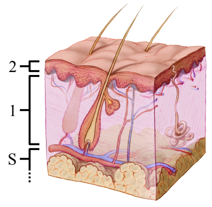

Figure 2: *Illustration of skin cross-section.*

Human skin is made up of multiple layers. Figure 2 shows an illustration of a cross section of human skin (by Don Bliss, National Cancer Institute). The subcutaneous region is labeled S, the dermis is the first layer, and the epidermis is the second layer. The epidermis consists of many sub-layers. Likewise, while the dermis is composed primarily of collagen and elastic fibers, it also contains a variety of other components, such as blood vessels, mechanoreceptors, and sweat glands. Despite the variation of composition, we make the common assumption that a continuum mechanics model can be suitably defined for each layer, and furthermore we use a thin shell model. Dermis and epidermis layers can be quite different across the body (e.g., eye lids in comparison to hands). Using the thin shell model, we can set the mechanical properties to correspond to the local properties, and we likewise accommodate the heterogeneous properties of the skin by allowing the parameters to vary across our layer models.

We assume that the deformation of the subcutaneous layer is provided as a blend shape, thus we only simulate a two layer model based on the dermis and epidermis. We couple neighboring layers by local constraints with a spatial frequency that corresponds to their mechanical properties. This follows a construction similar to the embedded thin shells work of Remillard and Kry [2013], except that we use different model for predicting wrinkle frequencies for different layers that takes into account the mutli-layer model, as described in the following subsection.

3.1 Critical wavelength model

The previous work on embedded thin shells [Rémillard and Kry 2013] applied a simple model describing the mechanical behavior of a thin film resting on top of a soft elastic foundation [Timoshenko and Gere 2009]. This model gives the critical wavelength of the top layer as a function of the the skin's thickness and elastic properties. In other words, once these parameters have been identified, the wavelength of wrinkles can be predicted. Human skin can likewise be approximated as a multi-layer composite material, where a thin and relatively stiff epidermis is attached to a soft dermis. In this case, the wrinkling frequencies can be described by using another simple physical model, which differs slightly from the single layer model. A general form of the wrinkle periodicity is given by Cerda and Mahadevan et al. [Cerda and Mahadevan 2003]. Given the thickness of the upper layer h_k and lower layer h_{k-1}, and the Young's modulus of each layer (E_k and E_{k-1} respectively), the critical wavelength is given by

$$\lambda_k \sim (h_k h_{k-1})^{1/2} \left(\frac{E_k}{E_{k-1}} \right)^{1/6}. \tag{1}$$

Once the critical wrinkling wavelength is known, we can determine a set of spatially distributed local constraints so that our multi-layer thin shell model produces wrinkles with the desired wavelengths. Note that when the skin layers have varying elasticity and thickness, this approximation of the critical wavelength will be local and we must take this into consideration when constructing constraints that allow for the appropriate variation of wrinkle wavelengths.

4 Constraints

Using a scheme similar to embedded thin shells, we build constraints to couple the thin shell simulations in different layers. A set of constraints must be defined between each pair of adjacent layers. The constraints are local and overlap in a manner that ensures that the higher layer will follow the lower layer at a spatial frequency lower than its critical wrinkling wavelength. We couple the bottom layer with the blend shape through position constraints, which constrains this first layer shell to the embedded positions in the blend shape.

Using $k \in \{1, 2, \cdots\}$ as a layer index, consider an upper layer defined by positions $\mathbf{x}_{k-1} \in \mathbb{R}^m$ and a lower layer with positions $\mathbf{x}_k \in \mathbb{R}^n$. In general, the number of vertices in the upper layer will be higher than the lower layer, and the positions will correspond to a different discretization of material space. We define constraints which include a normal-based term to account for the layer thickness,

$$H_k \mathbf{x}_k = H_{k-1}(\mathbf{x}_{k-1} + t_k \mathbf{n}_{k-1}), \tag{2}$$

where $\mathbf{n}_{k-1}$ provide the lower layer vertex normals, t_k is the layer thickness, and $H_k : \mathbb{R}^n \rightarrow \mathbb{R}^c$ and $H_{k-1} : \mathbb{R}^m \rightarrow \mathbb{R}^c$ form a sparse constraint matrix across all degrees of freedom. The weighted combination of vertex positions defined by the constraints must be computed to address different layer mesh discretizations that arise in using adaptive meshes for the layers and we address this below in Sections 4.2 and 4.3. For heterogeneous materials, the thickness of a layer is non-constant, and we replace the scalar t_k with a diagonal matrix to account for the thickness at different vertices. The thickness must be sampled from a property map based on the current layer discretizations (see Section 5), and similarly the normals must be computed (we use an area weighted average of adjacent face normals).

Below, we first address the construction of the constraints, and then describe our approach for consistently sampling the constraint functions when the adaptive layer mesh changes.

4.1 Construction by clustering

Simulating skin layers with heterogeneous material properties requires that we establish non-uniform constraints according to the critical wavelength of each vertex. Each vertex can be assigned a wavelength based on its material space position and a material space property map. By using the critical wavelength, we produce a weighted clustering where the clusters have different sizes. Small clusters form in regions of low wavelength, and large clusters form in regions of high wavelength.

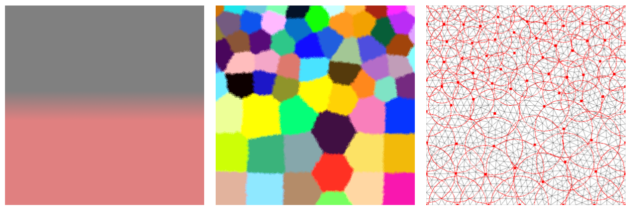
Figure 3: *Constraint construction visualized in material space. Left shows the varying property map of the skin. Middle shows the clusters formed based on the expected wrinkle properties with smaller areas higher in material space to accomondate smaller wrinkles. Right shows the continuous truncated Gaussian supports overlaid with an adaptive meshing of material space.*

We use a greedy approach to create clusters. We approximate the distance of a vertex to a cluster center by the shorted path following mesh edges, similar to Remillard and Kry [2013], except that we divide the mesh edge lengths by the average critical wavelength of the adjacent vertices. This effectively encourages regions where we expect high frequency wrinkles to form smaller clusters because the edge traversal costs in these regions will be larger.

We perform the clustering on a fine meshing of the material space so as to best approximate the variation of wrinkle wavelengths across the continuous regions of material space. We start by choosing a random vertex as the first center, then compute the shortest weighted edge traversal distance to all other vertices using Dijkstra's algorithm. Given the critical wavelength at the randomly selected vertex, we do not want to include another vertex cluster within radius r equal to $2/\pi$ the critical wavelength (see Remillard and Kry [2013] but here r is not constant) Then, among the vertices that are sufficiently far from other existing cluster centers, we choose the farthest vertex as the next center. The process repeats until all vertices are within half a wavelength of a cluster center.

Once clusters are identified, the critical wavelength computed for the vertex in the center of the cluster determines the radius of the continuous constraint function, specifically, a truncated Gaussian. Because the layer meshes are adaptive, we do not identify in advance which vertices will be included in the constraint. An example of the clustering can be seen in Figure 3. Given uniform elastic properties, the red brightness indicates the thickness of the skin layer. The darker region is thinner resulting in a smaller critical wavelength and thus smaller clusters and smaller radii for the continuous constraint functions. The extent of the truncated Gaussian constraint functions can be seen as circles in the right most image of the figure.

4.2 Area weighted constraint sampling

Because the layers do not have a fixed meshing, we must sample the continuous functions to create constraints for a given discretization of the layers. Within a cluster of vertices belonging to the upper layer, each constraint requires the weighted position average of those vertices to match a corresponding weighted position average of vertices in the lower layer. For a given constraint function and a given material space mesh for each layer, the set of vertices influenced by the constraint is defined by the radius of the continuous constraint function, which is equal to half the critical wavelength. The constraint in Equation 2 with the normal term omitted can be written in an expanded form as

$$\sum_{i \in C_k} \omega_i x_i^k = \sum_{j \in C_{k-1}} \omega_j x_j^{k-1}, \tag{3}$$

where C_k and C_{k-1} are the sets of vertices involved at each layer, and ω_i are weights given by a truncated Gaussian function,

$$\omega_i = \sigma_c a_i e^{-\frac{d_i^2}{2\sigma^2}} \text{ if } d_i < 2\sigma, \ \ 0 \text{ otherwise.} \tag{4}$$

Here σ_c normalizes the sum of weights to one ensuring an affine combination, a_i is the area corresponding to vertex i, which equals the average of one third the area of the adjacent faces, $\sigma = \frac{1}{2}r$, and d_i is the distance from vertex i to the constraint's center. Note again that the truncation at distance 2σ determines the vertex presence in the cluster, namely, it is within the radius of constraint center.

Figure 4 shows a 2D representation of the constraint sampling. The constraint function is a truncated gaussian, and each sample is weighted differently given the distance to its neighbors (small ticks on the horizontal axis denote the midpoints between samples). Note that we must renormalize these weighted samples to produce an affine combination, and without an additional modification, the affine combination of the sample locations does not correspond to the center of the constraint (compare the thick gray and black vertical lines). Using area weights is important because the adaptive mesh may create scenarios where there are small triangles on one side of a constraint and larger ones on the other side. But given that the normalized area weighted sampling does not sum to give the center of the constraint function, we make an adjustment as described in the next section.

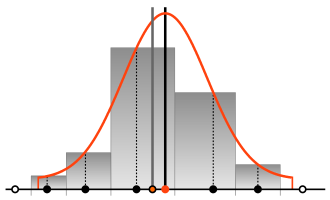
Figure 4: *2D illustration of constraint sampling.*

4.3 Constraint Adjustment

The weights of vertices in a constraint are determined by its area and distance to the center of the cluster, and the constraint coefficients for vertices are resampled each time a layer is remeshed. Anisotropic adaptive remeshing leads to changes in the sampled constraint value, and produces unstable simulations if these changes are not addressed. Therefore, we propose a method to adjust the weights of each vertex such that the weighted average positions of the material space coordinates remains at the exact center of its constraint after adaptive remeshing.

Given a vertex with material space coordinates $u^i = (u_x^i, u_y^i)$, we compute an adjusted weight ω_i^* as

$$\omega_i^* = \omega_i + \alpha_1 m_1(u_x^i) + \alpha_2 m_2(u_y^i), \tag{5}$$

where α_1 and α_2 are coefficients of functions that alter the weight depending on the material space location. We use simple linear functions for m_1 and m_2, which evaluate zero at the cluster center,

$$m_1(u_x^i) = u_x^i - u_{c_x}^*, \tag{6}$$
$$m_2(u_y^i) = u_y^i - u_{c_y}^*. \tag{7}$$

where u_c^* is the coordinates of the center of the cluster. To solve for α_1 and α_2, we note that the sum of material space coordinates with adjusted weights should equal the cluster center, that is,

$$\sum_{i \in C} (\omega_i + \alpha_1 m_1(u_x^i) + \alpha_2 m_2(u_y^i)) u^i = u_c^*. \tag{8}$$

Bringing the sum of old weighted positions to the right hand side, we can create and solve a 2×2 matrix system as $A\alpha = b$ where

$$A = \sum_{i \in C} [m_1(u_x^i) \ \ m_2(u_y^i)]^T u^i, \quad b = u_c^* - \sum_{i \in C} \omega_i u^i. \tag{9}$$

While the system solve is trivial, there is a cost to assembling A and b proportional to the number of vertices in the cluster. However, the cost is small relative to the initial area weighted constraint sampling and normalization.

5 Simulation

Our implementation is based on the ARCSim code developed by Narain et al. [2012]. We naturally inherit the methods from their work for deformable surface simulation. We use the piecewise linear model described by Wang et al. [2011], the co-rotational finite element method [Müller and Gross 2004] for in-plane stretching, and discrete hinge model [Bridson et al. 2003] for bending energy. In addition, a bounding volume hierarchy [Tang et al. 2010] is used to detect collisions and non-rigid impact zones [Harmon et al. 2008] is used to resolve collisions. The linear system is solved using direct solver in the TAUCS library.

At each simulation step, we add a sampling step after the adaptive remeshing to resample the material property and weights of each vertex with respect to the new discretization. Furthermore, we modify the solver step as follows. There is a step to build the constraint matrix, and we've changed the framework to solve the constrained system with an implicit Euler method. In addition, we must update the position of vertices in blend shapes at the beginning of each frame. We have implemented both a statics position based solver and dynamic velocity based solver. The position based solver computes a quasi-static equilibrium of each layer from the configuration of lower layer, without considering the mass of skin and velocity damping. Buckling in this case happens suddenly once the skin reaches the critical strain. In contrast, the velocity based solver take inertia into consideration, and with sufficient damping we can observe smooth transitions between energy minima. The comparison between them can be seen in the supplementary video.[1]

5.1 Property map sampling

We assign a texture map to the material space to allow varying layer material properties across the surface. Properties are encoded in the different color channels of the texture map, allowing an artist to simply paint varying thickness or stiffness. The red channel represents the critical wavelength, the green channel is for Young's modulus, and the blue channel indicates thickness. For each model, having a reference material parameter with default thickness and Young's modulus, we convert the sampled RGB values to corresponding relative values, then scale the reference material to real value. Our two-layer skin model has a reference material of Young's modulus $E = 4$ MPa and thickness $h = 1.3$ mm for the first layer, and a reference material with $E = 600$ MPa and $h = 0.12$ mm for the second layer. For the soft substructure we use $E = 0.5$ MPa and $h = 15$ mm.

When the a skin layer is adaptively remeshed, we set the elastic properties of the new discretization by sampling the property map. For instance, a new edge will have its bending stiffness computed from values obtained from the property map at the midpoint of the edge. Note that this assumes that heterogeneous properties in the material map are smooth and effectively constant at the resolution of mesh. This is discussed in Section 6.1.

[1]`http://www.cs.mcgill.ca/~kry/pubs/mlw`

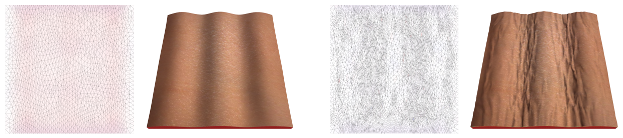

Figure 5: *Comparison between one layer simulation (left) and two layer simulation (right) under the same compression.*

5.2 Wrinkle map construction

The simulation of very fine wrinkles with our multi-layer model can be quite costly, but we need not compute the shapes every time if we are only using a small number of blend shapes. Instead, we see our model as an interesting means for automating the creation of wrinkle maps. The creation of a wrinkle map is straightforward as we only need to drawing the material space mesh with the vertex color set to the normal direction. We map each components of the normal to the $[0, 1]$ interval and store the result in an 8 bit texture map (we find that quantization errors are not noticeable in our application). We create wrinkle maps for a few key frames that capture the different wrinkling modes, and then use a texture shader to interpolate the normal maps based on the parameter of the blend shape model. This produces plausible results that resemble those of the physics-based simulation, as can be seen in an example of a two-layer skin model in the supplementary video. While we demonstrate this process for a blend shape model, we note that it can also be combined with skinning methods where the angle of the joint would provide the interpolation parameter.

6 Results and Discussion

Table 1 shows a breakdown of computation time for the different examples in this paper. We evaluated both an iterative MINRES solver and a direct solver, and while both are time consuming we prefer the latter because we do not need to worry about preconditioning to facilitate the solver progression. Note that we set the minimum allowable triangle size of adaptive remeshing to be 0.5% side length of squre patches, which produces a large number of degree of freedom and slower solves.

Figure 5 is a good example that helps justify our multi-layer concept. We can easily observe small wrinkles standing on larger wrinkles in the two-layer model, while the single layer model does not produce this effect. Attaching the second layer increases the realism of skin simulation. Meanwhile, the material space meshes demonstrate the superiority of adaptive meshing, which focuses the resolution on the regions where small wrinkles appear. An example simulation accounting for heterogeneous material properties can be seen in Figure 6. A one-layer example showing the changes of wavelength in accordance with the varying thickness and stiffness can be see in the supplementary video.

Aiming to produce realistic animation, we design three blend shape models to mimic typical human skin deformations (see the video). This includes compressing, twisting, and a pinch while pulling up. Figure 1 shows the plausible results for wrinkled human skin, and we use a skin texture to render this example in order to provide increased visual realism. In addition, to show a different visualization of the changing geometry, we include a uniform color Phong shading example in the supplementary video.

6.1 Limitations

Sampling of material properties assumes that they only change smoothly. Note that it would be an interesting challenge to compute appropriate material properties for a very coarse mesh such that the mechanical behavior matched the finer version. Because our base mesh resolution is high enough to accurately model the variation on the material property maps we assume that we do not have artifacts due to numerical coarsening. Alternatively, if sharp material boundaries were to be introduced in the material property map, these could be handled by constraining the adaptive mesh such that it always places edges along the discontinuity.

				Avg. time/frame			
Figure	N_c	Avg. N_v	t_c	t_I	t_R	t_S	Total
5 right	436	7693	1.10	107.4	0.24	1.44	109.10
6 middle	189	3104	0.08	76.1	0.05	0.80	76.89
6 left	139	2696	0.05	71.81	0.06	0.63	72.50
1 top	421	11529	1.01	269.8	0.65	3.42	273.87
1 middle	421	6770	1.01	85.57	0.19	1.06	86.82
1 bottom	401	5294	0.98	81.45	0.22	1.21	82.88

Table 1: *Performance measurements for our examples, where N_c is the number of constraints, N_v is the number of vertices, t_c is the time for constructing constraints, t_I is the time for solving and integrating the system, t_R is the time for remeshing, and t_S is the time for constraint sampling, material property sampling, and constraint adjustment. All timing is in seconds and was performed on a machine with 2.4 GHz Intel Core i7 CPU.*

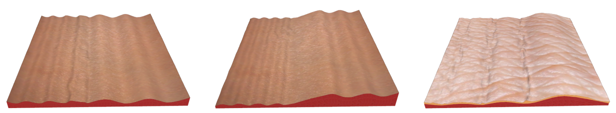

Figure 6: *Examples of one and two layer skin models under compression with different heterogeneous material properties. From left to right, single layer with varying stiffness, single layer with varying thickness, attaching second layer on one layer model with varying stiffness.*

We assume that the material space embedding has low distortion, and therefore that we can reasonably define the continuous constraint functions in material space. When there is high distortion in the embedding, it is difficult to assign a meaningful truncation distance and weights.

Finally, we note that remeshing can produce temporally unstable visual artifacts for the fine wrinkles that our simulations generate. This is most evident in the Phong shaded example where highlights jump when the remeshing produces new vertex normals. We plan to address this problem in future work.

7 Conclusions

Our multi-layer physics-based skin model improves upon the single layer model for wrinkle simulation. Permitting heterogeneous material properties is important for improving simulation fidelity, and is addressed by our model. Simulating each layer with an adaptive mesh allows for reasonable computation times with respect to the size of the fine details that are achieved. By designing continuous constraint functions that respect the predicted wrinkling wavelengths, we transform the problem of assigning constraints into a sampling problem. By using blend shapes to drive the bottom later, and using texture maps to define the heterogeneous elastic properties and varying layer thickness, we have a system that can be used to produce fine physical wrinkling details into standard character animation models. While our demonstrations focus on small patches of skin, we believe our technique is useful for the creation of wrinkles on faces, necks, fingers, and arms. Finally, because we can easily construct wrinkle maps from our high resolution simulations, we expect that this technique can find use in interactive applications, such as games and movies.

Acknowledgements

This work was supported by NSERC, CFI, and GRAND NCE.

References

BANDO, Y., KURATATE, T., AND NISHITA, T. 2002. A simple method for modeling wrinkles on human skin. In *Proc. of the 10th Pacific Conference on Computer Graphics and Applications*, 166–175.

BARAFF, D., AND WITKIN, A. 1998. Large steps in cloth simulation. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '98, 43–54.

BERGOU, M., MATHUR, S., WARDETZKY, M., AND GRINSPUN, E. 2007. Tracks: toward directable thin shells. *ACM Trans. Graph. 26*, 3 (July), 50:1–50:10.

BLINN, J. F. 1978. Simulation of wrinkled surfaces. *SIGGRAPH Comput. Graph. 12*, 3 (Aug.), 286–292.

BRIDSON, R., FEDKIW, R., AND ANDERSON, J. 2002. Robust treatment of collisions, contact and friction for cloth animation. *ACM Trans. Graph. 21*, 3 (July), 594–603.

BRIDSON, R., MARINO, S., AND FEDKIW, R. 2003. Simulation of clothing with folds and wrinkles. In *Proc. of the 2003 Symposium on Computer animation*, 28–36.

CERDA, E., AND MAHADEVAN, L. 2003. Geometry and physics of wrinkling. *Phys. Rev. Lett. 90* (Feb), 074302:1–4.

CHEN, Z., FENG, R., AND WANG, H. 2013. Modeling friction and air effects between cloth and deformable bodies. *ACM Trans. Graph. 32*, 4 (July), 88:1–88:8.

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. Special Issue on SCA 2005.

GEERLIGS, M. 2010. *Skin layer mechanics*. PhD thesis, Technische Universiteit Eindhoven.

GENNISSON, J. L., BALDEWECK, T., TANTER, M., CATHELINE, S., FINK, M., SANDRIN, L., CORNILLON, C., AND QUERLEUX, B. 2004. Assessment of elastic parameters of human skin using dynamic elastography. *Ultrasonics, Ferroelectrics and Frequency Control, IEEE Trans. on 51*, 8 (Aug), 980–989.

GRINSPUN, E., HIRANI, A. N., DESBRUN, M., AND SCHRÖDER, P. 2003. Discrete shells. In *Proc. of the 2003 Symposium on Computer animation*, 62–67.

HARMON, D., VOUGA, E., TAMSTORF, R., AND GRINSPUN, E. 2008. Robust treatment of simultaneous collisions. *ACM Trans. Graph. 27*, 3 (Aug.), 23:1–23:4.

JIMENEZ, J., ECHEVARRIA, J. I., OAT, C., AND GUTIERREZ, D. 2011. *GPU Pro 2*. AK Peters Ltd., ch. Practical and Realistic Facial Wrinkles Animation, 15–27.

KAVAN, L., GERSZEWSKI, D., BARGTEIL, A. W., AND SLOAN, P.-P. 2011. Physics-inspired upsampling for cloth simulation in games. *ACM Trans. Graph. 30*, 4 (July), 93:1–93:10.

MAGNENAT-THALMANN, N., KALRA, P., LUC LEVEQUE, J., BAZIN, R., BATISSE, D., AND QUERLEUX, B. 2002. A computational skin model: fold and wrinkle formation. *IEEE Trans. on Information Technology in Biomedicine 6*, 4 (Dec.), 317–323.

MÜLLER, M., AND CHENTANEZ, N. 2010. Wrinkle meshes. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '10, 85–92.

MÜLLER, M., AND GROSS, M. 2004. Interactive virtual materials. In *Proceedings of Graphics Interface 2004*, 239–246.

NARAIN, R., SAMII, A., AND O'BRIEN, J. F. 2012. Adaptive anisotropic remeshing for cloth simulation. *ACM Trans. Graph. 31*, 6 (Nov.), 152:1–152:10.

NARAIN, R., PFAFF, T., AND O'BRIEN, J. F. 2013. Folding and crumpling adaptive sheets. *ACM Trans. Graph. 32*, 4 (July), 51:1–51:8.

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

RÉMILLARD, O., AND KRY, P. G. 2013. Embedded thin shells for wrinkle simulation. *ACM Trans. Graph. 32*, 4 (July), 50:1–50:8.

ROHMER, D., POPA, T., CANI, M.-P., HAHMANN, S., AND SHEFFER, A. 2010. Animation wrinkling: Augmenting coarse cloth simulations with realistic-looking wrinkles. *ACM Trans. Graph. 29*, 6 (Dec.), 157:1–157:8.

SIMO, J. C., AND FOX, D. D. 1989. On stress resultant geometrically exact shell model. Part I: formulation and optimal parametrization. *Comput. Methods Appl. Mech. Eng. 72*, 3 (Mar.), 267–304.

TANG, M., MANOCHA, D., AND TONG, R. 2010. Fast continuous collision detection using deforming non-penetration filters. In *Proc. of the 2010 Interactive 3D Graphics and Games*, 7–13.

THOMASZEWSKI, B., WACKER, M., AND STRASSER, W. 2006. A consistent bending model for cloth simulation with corotational subdivision finite elements. In *Proc. of the 2006 Symposium on Computer animation*, 107–116.

THOMASZEWSKI, B., PABST, S., AND STRASSER, W. 2009. Continuum-based strain limiting. *Computer Graphics Forum 28*, 2, 569–576.

TIMOSHENKO, S. P., AND GERE, J. M. 2009. *Theory of Elastic Stability*. Dover Civil and Mechanical Engineering Series.

WANG, H., HECHT, F., RAMAMOORTHI, R., AND O'BRIEN, J. 2010. Example-based wrinkle synthesis for clothing animation. *ACM Trans. Graph. 29*, 4 (July), 107:1–107:8.

WANG, H., O'BRIEN, J. F., AND RAMAMOORTHI, R. 2010. Multi-resolution isotropic strain limiting. *ACM Trans. Graph. 29*, 6 (Dec.), 156:1–10.

WANG, H., O'BRIEN, J. F., AND RAMAMOORTHI, R. 2011. Data-driven elastic models for cloth: modeling and measurement. *ACM Trans. Graph. 30*, 4 (July), 71:1–71:12.

ZHOU, Z., SHU, B., ZHUO, S., DENG, X., TAN, P., AND LIN, S. 2012. Image-based clothes animation for virtual fitting. In *SIGGRAPH Asia 2012 Technical Briefs*, SA '12, 33:1–33:4.