Extending Implicit Skinning with Wrinkles

Fabio Turchet
MPC London, Bournemouth University, UK
fabio-tu@moving-picture.com

Oleg Fryazinov
The National Centre for Computer Animation, Bournemouth University, UK
ofryazinov@bournemouth.ac.uk

Marco Romeo
MPC London, UK
marco-ro@moving-picture.com

ABSTRACT

We propose a wrinkle system that takes as input the fields created in the implicit skinning framework, calculates the angle between their gradients and builds a scalar angle field. Its gradient resembles plausible wrinkle directions. The system is procedural and works as a post process by projecting vertices in a wrinkle field constituted of convolution surfaces.

Categories and Subject Descriptors

I.3.7 [**Computer Graphics**]: Three-Dimensional Graphics and Realism - Animation

Keywords

Implicit skinning, convolution surfaces, HRBF, wrinkles simulation

1. INTRODUCTION

Skinning is one of the most important areas in the field of computer animation and visual effects, where character realism and believability are essential. A large number of methods exist and are implemented in modern animation systems [6]. One of the recent advancements in the area which produces quite realistic results is the Implicit Skinning technique [14]. Unlike other methods which only use surface (vertices and triangles) data, Implicit Skinning makes use of implicit fields, in particular Hermite Radial Basis Functions (HRBF). This helps solving the drawbacks that linear blend skinning and dual quaternion skinning comport: loss of volume, lack of collisions and unwanted bulging.
During animation, the fields associated to each skeleton bone are rigidly transformed and the mesh vertices march in the direction of the gradient of the combined field until they reach the isovalue they had in rest pose. This allows to better preserve the initial shape and deal with self collisions in the folding region, either by stopping the marching at gradient discontinuities or by using a contact operator [15]. The framework also allows the creation of interesting effects through the use of gradient-based operators, for example the bulge in contact which approximates volume preservation. Nevertheless, to our knowledge important secondary effects such as wrinkling have not been previously investigated for this technique.

In this paper we propose to extend implicit skinning with wrinkles. Our technique benefits from the available transformed scalar fields: from their interaction a vector field that resembles plausible wrinkle directions is generated, hence creating a hybrid polygonal-implicit approach.
The main contributions of the paper are:
1) A technique for the creation of wrinkle curves based on the implicit skinning deformation,
2) A large set of parameters to allow users to procedurally tune the behaviour of the wrinkles and thus obtain the desired result.

2. RELATED WORK

Despite being considered a secondary effect in skin simulation systems [5], wrinkling as a separate effect attracted the attention of both academia and industry as an important detail to add realism to CG characters.

Elastic thin sheets like cloth or skin create wrinkles when compression forces are applied, to preserve material and isometry (the property of being inextensible) of the rest-pose object.
Current approaches can be categorized in artist-driven, in which drawn wrinkles maps are blended [7]; procedural based on textures [8] and physics based, often using a bending energy formulation in a specialized cloth solver [3].
An accurate way to measure stretching and compression is via the stretch tensor as presented in [12]. Their work is focused on augmenting coarse cloth mesh simulations with fine details as wrinkles and refine the tessellation in localized areas through adaptivity. The rest mesh is parametrized to a plane in order for the triangles to have a common frame of reference and the stretch tensor is calculated as defined in [13]. This is then diagonalized and its eigenvalues and eigenvectors give the magnitude and direction of the wrinkle vector field, respectively. Wrinkles are then generated as convolution surfaces from this field's streamlines by growing curves from seeds with the highest vector magnitude. Our technique follows the same procedural approach for curves generation but substitutes the stress field by the gradient of a scalar angle field, without using planar parametrization.

Recent works on skin in particular make use of physics-based simulation. In [11] is presented an approach in which thin shells are embedded in coarser finite element meshes to simulate the wrin-

kling effects of hard skin surrounding soft objects. The key idea is the use of frequency based position constraints, modeled as discretized continuous functions, that allow wrinkle formation only at wavelengths matching physical material properties. The work by [9] extends this technique by adding multi-layer support for heterogeneous materials and creating extremely fine and realistic skin details trough the use of adaptive meshes, for which normal or displacement maps can be exported.

Wrinkles as part of a larger simulation framework were presented in Weta's proprietary Tissue system [2] that uses finite elements to simulate fascia/fat/skin layers and constraints to solve wrinkling problems.

3. METHOD

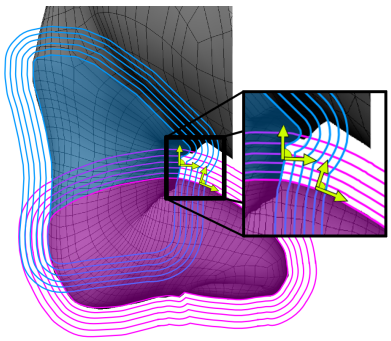


Figure 1: In pink and blue two HRBF fields; angles between their gradients are shown in yellow.

Our method works in the Implicit Skinning framework [14], as a post-process following the projection step that brings the vertices reaching their rest-pose isovalue (0.5 in our implementation). The following subsections detail the steps to generate plausible wrinkles for a mesh deformed with the implicit skinning algorithm. The required inputs are therefore the implicit fields of the segmented mesh of which we use value and gradient.

3.1 Angle Fields Preparation

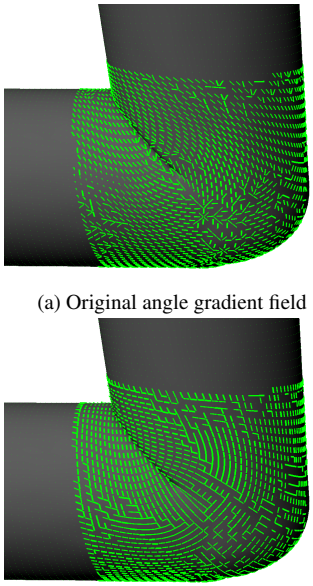
At the very core of the technique lies the generation of a discrete scalar field for each pair of adjacent joints in the skeleton hierarchy. Let f_1 and f_2 be two consecutive fields with compact support. During animation of a bending joint chain f_1 and f_2 will rotate rigidly towards each other. For each vertex in $f_1 \cap f_2$ we can then calculate straightforwardly the angle between the gradients of f_1 and f_2 as (see Figure 1):

$$angle(f_1, f_2) = \arccos\left(\nabla f_1 \cdot \nabla f_2\right)$$

Once an angle field f_A is defined, we approximate its gradient with finite differences by searching in the one-ring neighbours the position with minimum value p_{min}. For a source vertex p the gradient is then:

$$\nabla f_A = p_{min} - p$$

In practice, the user can choose to use a version of this field in which the gradients are biased towards one of the connected edge



(a) Original angle gradient field

(b) Topology-biased gradient field

Figure 2: Angle field generation using one-ring neighbours (a) and connected vertices (b)

vectors. This helps in some cases to achieve more realistic anisotropic patterns (see Figure 2). To generate fields less dependent on the mesh resolution we can also search in ring neighbours of levels greater than one, which are pre-stored for efficiency.
Considering three hierarchical fields f_1, f_2 and f_3, we allow them to interoperate in such a way that f_1 can create wrinkles with field f_2, but not with f_3, child of f_2. Being the angle calculation independent per each vertex, this step can be parallelized.

3.2 Curves Creation

At each frame every vertex is processed and inserted in a priority queue based on its angle value. The curve generation algorithm begins by selecting a seed from the top of the queue (red dots in Figure 5(f)); starting from it, vertices are added to the curve by iteratively choosing the next candidate in the one-ring neighbourhood N that has edge vector most similar to the gradient of the field at the source point. Process is repeated for the newly added point.
As in [12] the curve is grown in both directions, with the addition that for each seed we invert the gradient at that point to grow the second half of the curve. By indicating with q a candidate neighbour and by $\mathbf{e} = q - p$ the edge vector, the next vertex to add to the curve is calculated as:

$$p_{next} = \min_{q \in N}(\arccos\left(\mathbf{e} \cdot \nabla f_A\right))$$

The generated angle field has gradients pointing in plausible wrinkles directions because the angles decrease moving further away from the folding region, where their values are similar to the joint bending angle.
Every time a new point is added, its Euclidean distance is checked against all the current curves to enforce the constraint that all of them must stay at a minimum user-defined distance. The curve points loop terminates when a point's angle is under a user-set threshold or when it ends up outside of the allowed internal region of the mesh. Due to the fact that for dense meshes the number

of curve points can be high and would negatively influence performance, a resampling of the curve can be performed which in practice corresponds to a simple pruning.
In order to maintain temporal coherence between consecutive frames, the active curve list is kept from a frame to another and a curve removed from it if its corresponding seed ends up outside one of the associated fields (i.e. its isovalue differs from 0.5). Therefore the curves are not grown again from their associated seed after they are generated. This is in line with the observation that for human skin in particular, wrinkles tend to form at the same position for multiple repetitions of the same movement (fingers are a typical example), a feature exploited for instance by [4] for their wrinkle likelihood map generation.

During animation, fields rigidly rotate and interpenetrate one another which causes configurations for which a field (the one associated to a finger phalanx for example) happens to be "immersed" into another field. This becomes problematic because seeds will be selected from the whole mesh segment and wrinkle would appear in unwanted regions, like the nail. Therefore, as a precomputation only the vertices on the same side of the joints' bending direction are selected for wrinkle generation. This is done easily by thresholding based on the dot product between the vertex normal and the up vector of the associated bone's local rotation frame.

3.3 Wrinkles Field

The curves created as described in section 3.2 are a set of line segments connecting vertices of the mesh. We transform these segments into convolution surfaces w_i and threshold the sum of their field contributions:

$$v_{wrinkles} = \sum_i (w_i) - T$$

In our method we use convolution surfaces with line segments as skeletons and Cauchy kernel as a potential function. This surface for line segment with position vector $\mathbf{a}_i$, normalized direction $\mathbf{d}_i$ and length l has the following closed-form formulation [10]:

$$w_i(\mathbf{x}) = \frac{n}{2p^2(p^2+s^2n^2)} + \frac{l-n}{2p^2q^2} + \frac{1}{2sp^3}(atan[\frac{sn}{p}] + atan[\frac{s(l-n}{p}])$$

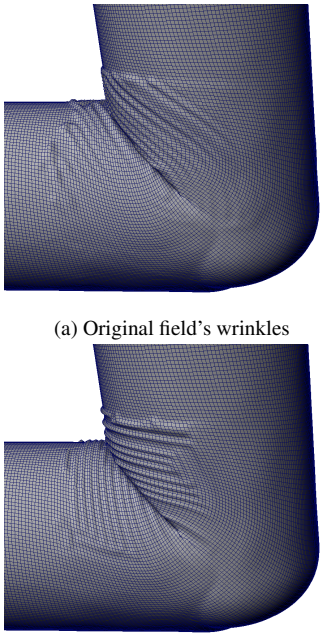
where $n = (\mathbf{x}-\mathbf{a}_i)\cdot\mathbf{d}_i$, $p^2 = 1+s^2(|\mathbf{x}-\mathbf{a}_i|^2 - n^2)$ and $q^2 = 1 + s^2(|\mathbf{x}-\mathbf{a}_i|^2 + l^2 - 2ln)$

The following default parameters are used for the convolution surface: $T = 0.5$ and $s = 0.85$.

The advantage of using convolution surfaces is double: the segments smoothly blend in and they can be integrated straightforwardly with the existing HRBF fields.
Because we don't have stretch information per face in our method, we modulate the radius of each segment by a factor that depends on the angle magnitude at each curve point. This in general makes the wrinkle thicker at the center of the field and smoothly narrower at the edges. For fast value query each segment's bounding box is also inserted in a KD-tree.

3.4 Projection

After obtaining vertices positions from the last step of implicit skinning and wrinkle field $v_{wrinkles}$, we combine the HRBF field and wrinkles field. Thus, the value v_{comb} at an arbitrary point p_w is:



(a) Original field's wrinkles

(b) Topology-biased field's wrinkles

Figure 3: Wrinkles generated from the fields in Figure 2

$$v_{comb}(p_w) = \max(v_{HRBF}(p_w) - 0.5,\ \ v_{wrinkles}(p_w))$$

It can be seen that here we use simple a set-theoretic operation, yet more complex blend operators can be used as well. We project vertices to this combined field by using Newton iterations (Figure 3). During projection we do not calculate the collisions using gradient discontinuities. Instead, to avoid evident mesh interpenetrations, we reuse the smoothing coefficients generated in the main implicit skinning step to exclude from the displacement the vertices in the colliding folding region.

Unfortunately we cannot use the equivalency between the decomposed stretch tensor's eigenvalues and the angles magnitude to activate the wrinkles because already in rest-pose the angle values are not zero and during animation we experienced noisy fluctuations in the angle field. Therefore, in order to have a smooth appearance of the wrinkles, the depth of the curve is modulated by the offset between the current and the rest-pose's joint activation angles, in a fashion similar to how a pose-based deformation works. The gradual appearance is achieved by placing the curve vertices initially under the surface at a distance R corresponding to the radius of the segment and smoothly moving them upwards in the normal direction.

For fast preview purposes also a simplified procedure can be applied. Instead of projecting the vertices to the wrinkles field, a simple displacement of the vertices corresponding to the curve points can be executed. However in this case the wrinkles radius is not controllable and wrinkles appear to be too sharp and uniform.

3.5 Parameters

Our system is procedural and allows the user to control the wrinkles appearance by tweaking basic parameters (see Figure 4), some of which are described below.

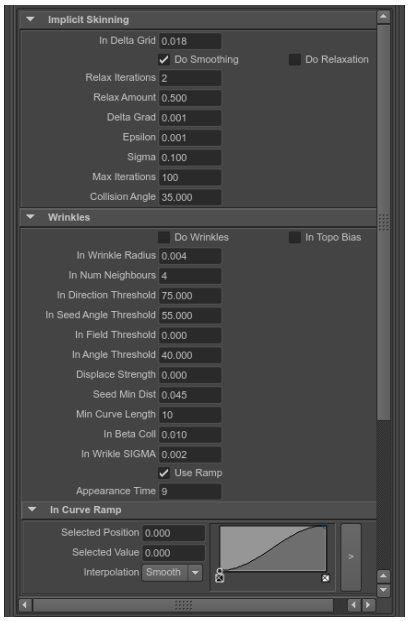


Figure 4: Plugin Parameters

Wrinkle Radius: the base radius for the convolution surfaces
Direction Threshold: the minimum angle that the gradients at two consecutive points of curve must form. This parameter helps to achieve a more organic look and longer wrinkles as it allows the curves to be more or less straight.
Seed Angle Threshold: the minimum angle that a seed point must have to be selected. This allows wrinkles to start forming also at the edges.
Field Threshold: minimum isovalue that a seed vertex must have to be selected
Angle Threshold: the minimum angle that any curve point must have; this controls the termination of the curve growth process
Seed Distance: keeps the seeds at a minimum Euclidean distance
Displace Strength: controls how deep the convolution surfaces stay under the mesh so to modulate bigger or smaller displacements
Ramp Control: a smooth spline that controls how fast and linear is the wrinkle appearance
Appearance Time: activation time, expressed in frames or joint angle degrees
Topology Bias: this parameter forces the angle field calculation to examine only the connected vertices.

4. IMPLEMENTATION

We implemented the method as a C++ Maya 2015 plugin. We store both value and gradient of the rest-pose HRBF fields as OpenVDB textures because their evaluation is computationally too expensive as it requires solving a linear system per frame. This means that at each frame we don't recompute the HRBF field but we use the inverse matrix of the joint transformations to read the value in rest-pose.
The angle field computation and the vertex projection are multi-threaded, but could also be implemented on the GPU easily for example using the Fabric Engine framework [1]. Our code could be further optimized, but performance highly depends on the number of iterations in the projection steps which in turn determines the number of accesses to the textures. On a quad-core Intel Xeon X3470 machine. for the thumb mesh with 15K vertices, 15 curves and 4 joints the framerate is ~10fps (including implicit skinning projection and smoothing).

5. RESULTS

We conducted tests of the technique mainly on cylindrical objects such as fingers, arms and legs and achieved believable results. Figure 5 and accompanying video show a detail of the thumb which is particularly interesting due to various wrinkle patterns of different sizes. This example uses constant radius and the field used is the non biased one: note how the curves develop independently of the mesh connectivity because of the higher degree of freedom in terms of possible neighbours directions.
Even though our technique doesn't always produce full wrinkle curves from one edge to the other of the finger (due to discontinuities or noise in the field), it still creates an organic and believable look. Note the use of the smoothing coefficients to avoid wrinkle displacement where vertices collide and the patterns comparable to a real example (Figure 5 (f)). Moreover, close wrinkles are blending in quite naturally because of the convolution surfaces formulation. Figure 6 shows comparative results between biased and non biased fields for the mesh of an arm: the topology-biased field produces more believable wrinkles.
In some cases unwanted bulges can appear: this is due to curves which are too short or with too high curvature in some of their segments. This could be addressed by filtering out these cases from the final used set. In addition, we noticed that the 3D texture resolution can strongly condition the quality of the angle fields due to interpolation approximations. In general we achieved better results using 4 or more ring-neighbours levels, especially for high resolution meshes; even though this improves the accuracy of the vector field, it also consequently slows down the system because more vertices have to be processed.

During animation the technique behaves as expected in terms of temporal coherence, with wrinkle curves that don't pop between frames and slide thanks to the underlying deformation. If the curves were retraced at each frame from the seeds, the result would be unstable and non coherent, mainly because of the differences in two temporally consecutive fields.
Nevertheless, the behaviour of the technique during animation could be further improved. Dynamic appearance is still not convincing enough, in fact the wrinkles appear too suddenly: this could be improved by tweaking the activation nonlinear curve manually or deriving it from experiments. Self collisions between the convolution surfaces (using a bulge-in-contact operator) would also prove beneficial to the overall fleshy look and dynamism of the deformation.

6. LIMITATIONS

Despite its potential, the presented technique has some limitations. For example, being the system time-dependent, finding the right parameters could require various iterations, something that makes it close in spirit to a simulation. Nevertheless, the user can always tweak single frames without rerunning the whole animation; in this

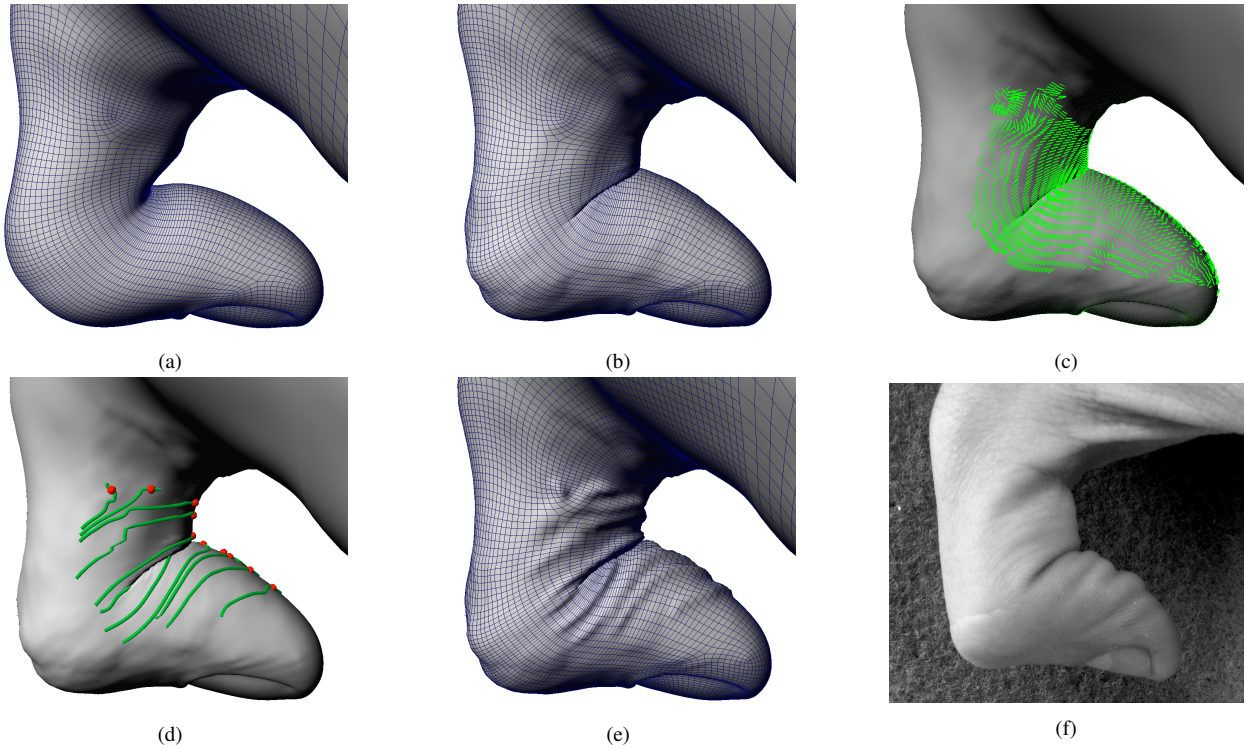


Figure 5: (a) Default Dual Quaternion skinning; (b) Implicit Skinning; (c) Angle gradient field (non biased); (d) Seeds and curves; (e) Wrinkles after projection; (f) Picture of a real thumb

case wrinkles will form following only the field at current frame. In addition, We also noticed that the quality of the fields tends to depend on the input shape, something that might limit the technique to cylindrical limbs and prevent the application to, for instance, face wrinkles generation.
Finally, a better falloff of the radius per curve would be desirable as wrinkles should fade away the further they are from the joint.

7. CONCLUSIONS AND FUTURE WORK

In this paper we presented a technique that allows wrinkles creation within the Implicit Skinning framework. The technique creates convenient and plausible results even comparable with physically accurate stretch-based methods. The results show the potential of the technique for applications within a production pipeline.

As future work we intend to integrate adaptive tessellation using for example Pixar's OpenSubdiv. Even though in a production pipeline is not common to have varying topology, the performance would benefit by having more detail only where needed, with the option to export displacement maps for rendering.
It would also be useful to investigate ways to make the convolution surfaces look less cylindrical and more organic, not restriced only to circle profiles; for example noise could be added, or segments with different radius could be layered.
Finally, another interesting future direction inspired by [4] would be to combine the proceduralism of the method with a trained approach to learn the space of input parameters for a closer match to physical appearance of acquired data.

8. REFERENCES

[1] Fabric Engine , 2015,. http://fabricengine.com/.

[2] WETA digital, 2013, tissue system,. http://www.fxguide.com/fxguidetv/fxguidetv-166-weta-digitals-tissue-system/.

[3] R. Bridson, S. Marino, and R. Fedkiw. Simulation of clothing with folds and wrinkles. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 28–36. Eurographics Association, 2003.

[4] C. Cao, D. Bradley, K. Zhou, and T. Beeler. Real-time high-fidelity facial performance capture. *ACM Trans. Graph.*, 34(4):46:1–46:9, July 2015.

[5] S. Clutterbuck and J. Jacobs. A physically based approach to virtual character deformation. In *ACM SIGGRAPH 2010: Talks*, 2010.

[6] J. Gain and D. Bechmann. A survey of spatial deformation from a user-centered perspective. *ACM Trans. Graph.*, 27(4):107:1–107:21, Nov. 2008.

[7] J. Jimenez, J. I. Echevarria, C. Oat, and D. Gutierrez. *GPU Pro 2*, chapter Practical and Realistic Facial Wrinkles Animation. AK Peters Ltd., 2011.

[8] S. Kimmerle, M. Wacker, and C. Holzer. Multilayered wrinkle textures from strain.

[9] P. Li and P. G. Kry. Multi-layer skin simulation with adaptive constraints. In *Proceedings of the Seventh International Conference on Motion in Games*, MIG '14, pages 171–176, New York, NY, USA, 2014. ACM.

[10] J. McCormack and A. Sherstyuk. Creating and rendering convolution surfaces. In *Computer Graphics Forum*, volume 17, pages 113–120. Wiley Online Library, 1998.

[11] O. Rémillard and P. G. Kry. Embedded thin shells for wrinkle

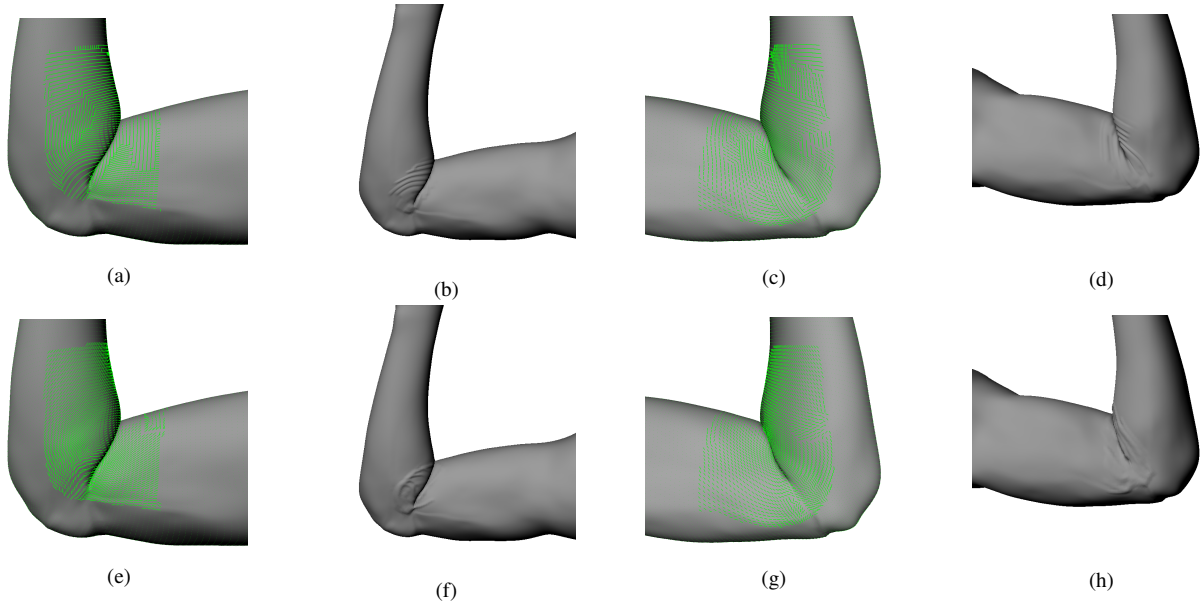


Figure 6: Comparison of the results obtained for an arm. (a)-(d) up and down side using topology biased field; (e)-(h) up and down side using normal, non biased field

simulation. *ACM Trans. Graph.*, 32(4):50:1–50:8, July 2013.

[12] D. Rohmer, T. Popa, M.-P. Cani, S. Hahmann, and A. Sheffer. Animation wrinkling: Augmenting coarse cloth simulations with realistic-looking wrinkles. *ACM Trans. Graph.*, 29(6):157:1–157:8, Dec. 2010.

[13] Y. Talpaert. *Tensor Analysis and Continuum Mechanics*. Springer Netherlands, 2010.

[14] R. Vaillant, L. Barthe, G. Guennebaud, M.-P. Cani, D. Rohmer, B. Wyvill, O. Gourmel, and M. Paulin. Implicit skinning: Real-time skin deformation with contact modeling. *ACM Trans. Graph.*, 32(4):125:1–125:12, July 2013.

[15] R. Vaillant, G. Guennebaud, L. Barthe, B. Wyvill, and M.-P. Cani. Robust iso-surface tracking for interactive character skinning. *ACM Trans. Graph.*, 33(6):189:1–189:11, Nov. 2014.