Augmented Skinning for Finger Simulation

Duo Li

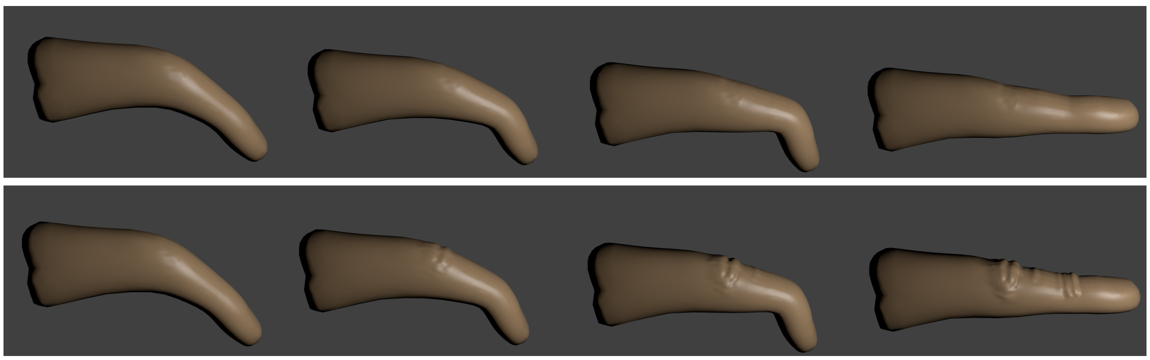

Figure 1: Automatic wrinkling on a finger mesh

1 Algorithm Overview

Our system consists of two components: dual quaternion blending [1] and automatic wrinkling [2]. We first deform the vertex with the skeleton by dual quaternion blending, and then automatically generate wrinkles upon the deformed mesh. We will discuss how we modify and implement these algorithms in the following two chapters.

1.1 Dual Quaternion Blending

Geometric skinning binds each vertex with a serial bone, which contribute the transformation of the vertex with different weights.

$$v = \sum_i w_i C_i v' \tag{1}$$

v is the vertex position and v' means the original vertex position. w_i denotes the amount of influence of the ith bone and C_i represents the rigid transformation. For linear blending, C_i is just the configuration matrix.

Rigid transformation can be decomposed into the rotation part and the translation part. Dual quaternion blending encodes these two components together with dual quaternion. Therefore, with just a linear combination and normalization, we can get a valid interpolated rigid transformation.

We also exploit the same mathematic trick to avoid extra matrix conversion from dual quaternion to configuration matrix as in [1]. The final procedure is as following:

1. Convert configuration matrices C_i into dual quaternions $\hat{q}_i$
2. Linearly interpolate $\hat{q}_i$ into $\hat{b} = \sum_i w_i \hat{q}_i$
3. Normalize b_0 (non-dual part of $\hat{b}$) as $c_0 = b_0/||b_0||$
   Normalize b_d (dual part of $\hat{b}$) as $c_d = b_d/||b_d||$
4. Construct the translation and rotation components directly with c_0 and c_d as a configuration matrix M
5. modify vertex v with $v' = Mv$.

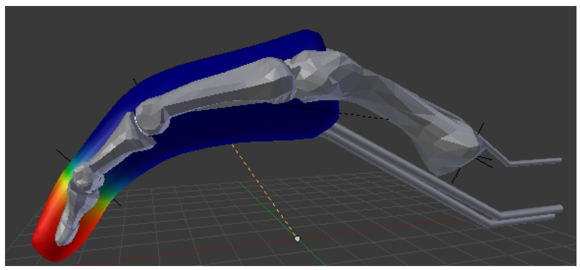

Figure 2: Automatic generated weights

For the tedious influence weights tuning, we use the approach [3]. This approach automatically generates the weights for an input skeleton with heat equilibrium. Figure 2 shows the weights of the finger mesh with respect to the distal bone.

1.2 Automatic Wrinkling

We use [2] to automatically generate wrinkles upon on the blended skinning. The general procedure consists of 4 steps:

1. Generate a physical simulation mesh sequence with a coarse mesh
2. Build a stretchiness tensor field from the physical simulation
3. Integrate wrinkle curves with previous and current simulated frames
4. Subdivide the mesh and project the vertices along the wrinkle curves with 'implicit primitive'

To fit this work into our pipeline for skinning, we modified some details.

1. We replace the physical simulation with the standard geometric skinning because simulating the highly constrained skin itself is an unsolved problem.

2. Since the wrinkle propagation direction is persistent on the finger, we do not need the curve integration step. Instead, we place the wrinkle curves in advance. We use Blender to design the curves and export this information into the simulator.

3. Following the circular arc model, we can get the radius and arc offsets of each vertex along the wrinkle curves. We use the kd-tree to determine the distance between the vertex pairs. For the vertices on the 'skeleton wrinkle curves', we radiate its contribution to its neighbor vertices

according to the distance. Then, instead of using the time consuming 'implicit primitive', we use the fast displacement based approach [4], which smooth the contribution of each vertex as:

$$P_i = \frac{\sum_j ||\Delta P_{ji}||^m P_j}{||\Delta P_{ji}||^m} \tag{2}$$

Here P_i is the vertex position, j is the adjacent vertices of vertex i, ΔP_{ji} is the distance between vertex j and i, and m gives the control of the smoothing level.

Eventually, we can generate the wrinkles in real time.

2 Results

We test our pipeline with a coarse finger mesh (Figure 1). The upper sub figures is the results with dual quaternion blending, without wrinkling augmentation. The bottom sub figures shows how the mesh be deformed with wrinkles with respect to the stretchiness tensors. The wrinkles show and disappear accordingly along with the finger movements.

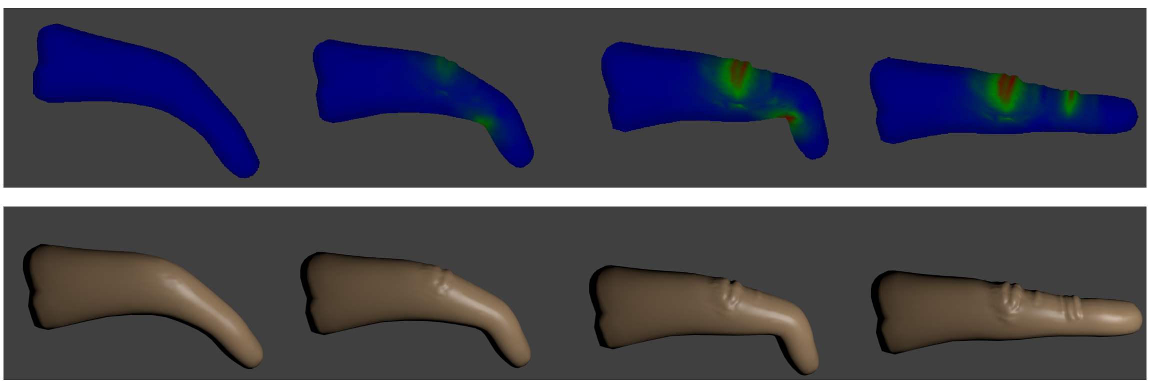

Figure 3: Stretchiness tensor map

Figure 3 shows the stretchiness tensor map. The color represents the stretchiness magnitude on this vertex. Red means highly compressed while pure blue means no compression at all.

Of course we can tune the parameter R_{min} to get different level of wrinkles just as [2].

3 Limitations and Future Works

In this project, we implemented the most advanced geometric skinning approach dual quaternion blending, and augmented this skinning with an automatic wrinkling algorithm.

We found an interesting problem for the geometric skinning approach. Not surprisingly, the geometric skinning approach lacks of proper physical property of the skin. Geometric skinning all relies on the weights for determining the region and magnitude of influence of each rigid body or joint. Usually this type of approach assumes that the skin is nearly rigid which only deforms near the joints. This limited influence region leads to a plausible mesh shape but also a very unbalanced

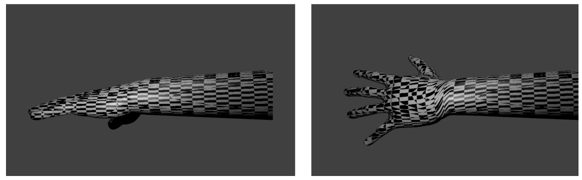

Figure 4: Distortion of the texture on the wrist

strain distribution. However, for the real skin, the strain tends to be uniformly distributed due to skin's elastic property. The finger demo is a good example (Figure 1). With dual quaternion blending, the finger mesh only compresses around the joints while there is no compression at all on the area between two joints. This result looks acceptable no physical augmentation approach is applied upon it. However, when we detect and augment the mesh with wrinkles with respect to the compression rates, or in other words the strains, little wrinkle can be generated for the intermediate areas.

This is even more noticeable with texture mapping. Just as the wrist demo (Figure 4), with the dual quaternion blending, we get a highly distorted texture around the wrist when the hand rotates along the forearm.

All these examples demonstrate we do need the sliding property of the skinning. We are working on this slidable skinning currently. We plan to tackle this problem with Eulerian simulation, which allows the material coordinates instead of the vertices themselves to slide along the geometrically blended skin.

References

[1] Ladislav Kavan, Steven Collins, Jiri Zara, Carol O'Sullivan *Geometric Skinning with Approximate Dual Quaternion Blending* ACM Trans. Graph. Volume 27, Number 4, Page 105, 2008

[2] Rohmer Damien, Popa Tiberiu, Cani Marie-Paule, Hahmann Stefanie, Sheffer Alla *Animation Wrinkling: Augmenting Coarse Cloth Simulations with Realistic-Looking Wrinkles*, ACM SIGGRAPH Asia, 2011

[3] Ilya Baran and Jovan Popovi??. *Automatic Rigging and Animation of 3D Characters.* ACM SIGGRAPH 2007

[4] Singh, K. and Fiume, E. *Wires: a geometric deformation technique.* ACM SIGGRAPH 1998