Skeleton Driven Laplacian Volumetric Deformation

C. Budd¹, A. Hilton²

¹ University of Surrey, c.budd@surrey.ac.uk ² University of Surrey, a.hilton@surrey.ac.uk

Abstract

This paper proposes a novel mesh animation technique which combines the flexible interactive control of skeleton based animation rigs with volumetric mesh deformation to avoid mesh collapse and self-intersection under folding and twisting motion. Our solution combines the industry standard Linear Skin Blending with a mesh based volumetric deformation approach. Linear Skin Blending is used to attach and efficiently animate a small number of points with a skeletal control rig. These points provide constraints for a Laplacian mesh deformation scheme which solves for the mesh which satisfies the constraints and gives minimum volume deformation of a tetrahedralization of the mesh vertices. This approach allows rigging and animation of a mesh driven animation for meshes of several thousand vertices without the known drawbacks of Linear Skin Blending, mesh collapse around joints and the 'candy wrapper effect'.

Keywords: Volumetric Deformation, Skeleton Driven Animation

1 Introduction

The ability to interactively repose a character is invaluable in both the film and game industries. Skeleton based control rigs are widely used to control the surface animation with mesh deformation based on the skeletal motion. Computationally efficient techniques such as vertex weighting or Linear Skin Blending are widely used to achieve interactive mesh deformation [14]. These techniques can result in visual artifacts due to mesh collapse around joints and axial twisting of the limbs. Recently Laplacian mesh deformation techniques [23, 1, 28] have been introduced which allow direct manipulation of mesh vertices to control the surface and interpolate example bases producing natural mesh deformation. However, these techniques lack the intuitive skeletal control of conventional animation rigs. In this paper we propose a method to combine skeletal animation rigs with example based volumetric mesh deformation to achieve interactive animation without the mesh collapse which occurs with conventional skeletal animation techniques.

Skeleton based approaches including the standard linear blend skinning method provide a number of advantages. Firstly, smaller dynamic scene representation without the need to store independent position data for every vertex at every frame. Secondly, animators use a large range of existing tools for editing and rendering skeleton based animation, giving them both familiarity and flexibility working with this style of animation. Thirdly, simplicity, the animator need only reposition the joints and does not need a detailed understanding of the underlying deformation algorithm.

Here we propose a means to achieve the visually pleasing results of mesh based deformation techniques whilst maintaining the advantages of skeleton based approaches. Our method uses the industry standard linear skin blending technique to connect a small number of points to a skeleton. These points then provide the small number of constraint points on which a tetrahedral deformation technique can act. This gives an animator intuitive control over the mesh based deformation and the ability to create animation with the traditional key-framing of joint positions and orientations or retargetting of motion capture data.

The technique we propose fits well with current animation pipelines. A full implementation of our skinning methodology has been implemented as a plug-in for Autodesk’s Maya. We demonstrate our approach to animate high-resolution capture meshes from 3D surface scans and multiple view reconstruction. Skeleton driven volumetric deformation allows captured high-resolution surfaces to be transformed into a representation which can be manipulated using standard animation tools.

2 Related Work

The most widespread and commonly implemented approach to mesh animation is Linear Skin blending [14]. Linear skin blending computes vertex location as a weighted combination of joint orientations. This method is employed by many commercial animation packages including Autodesk’s Maya and 3D Studio Max. Linear skin blending provides interactive animation of a character with the simple manipulation of joint positions. Although simple and fast, linear skin blending often fails to produce physically plausible character deformation resulting in collapse around joints due to folding and twisting motion.

Many extensions to linear skin blending attempt to overcome its downfalls. Much work has been done into reducing the affect of collapse under large angle changes and twisting [10, 16]. Most recently Kavan et al [9] present an approach using dual quaternion interpolation which does not present
the artifacts linear skin blending does under twisting. An efficient GPU implementation also presents comparable performance. A number of more complex skeleton based animation techniques have been developed with the aim of achieving more accurate character deformation. Chadwick et al. present a multilayer model [6] which consists of not only the skeleton and skin layers but a number of intermediate layers. These layers simulate the physical behavior of muscles, tendons and other anatomical structures. Thalmann et al. introduce metaballs [25] as an intermediate layer representing muscle and other organic structures. A more detailed approach is the hybrid model [26] in which each anatomical layer is modeled according to the level of detail required for accurate representation. The most detailed models represent the anatomy of a human accurately [27, 17]. Although these approaches produce highly accurate and realistic deformation they are computationally expensive and the models themselves are complex to construct.

Other work focuses on schemes for directly editing the mesh without skeletal information. Earlier work in this area focuses on subdivision and multiresolution mesh representations. Here geometry is encoded as a base mesh and several levels of refinement. Typically a low-frequency base mesh provides basic shape with detail refinements described locally. The multiresolution hierarchy can be constructed for both connectivity and geometry [31, 8] or simply the geometry [11, 13, 3]. In meshes with high levels of detail many levels of the multi-resolution hierarchy may be required to adequately preserve this detail.

More recent work focuses on a single resolution representation using Laplacian (differential) coordinates [1] or the so called pyramid coordinates [18]. These schemes allow an animator to manually place a small number of vertices and compute the remainder. These positional constraints are factored into the system of equations which reconstructs the Cartesian coordinates of the mesh from the differential coordinates. The primary difficulty with Laplacian deformation is the handling of rotations within the local frame. Both [15] and [29] attempt to deal with this by explicit assignment of these rotations.

Sumner et al. [22] introduce the concept of deformation gradients. A deformation gradient describes the individual affine transformations of each triangle with respect to a reference pose. Their MeshIK system [23] uses this differential representation for the purpose of shape interpolation. They allow a mesh to be reposed as a non-linear combination of a number of example poses. The concept of shape interpolation using a differential representation is common to a number of mesh deformation frameworks [2, 28]. The main disadvantage of this style of deformation is the requirement for a number of example pose meshes with correspondence.

Botsch et al [4] present a approach to mesh editing based upon prisms. Their approach (PriMo) embeds the surface mesh in a layer of prisms connected by non-linear elastic energies. These prisms behave like thin shells and plates resisting bending and stretching. The rigidity of these prisms prevents degeneration of the mesh. More recent work has moved towards volumetric deformation rather than surface based techniques. Botsch et al [5] represent a mesh using voxels to form a number of rigid cells again connected with non-linear elastic energies. Deformation is first carried out on these cells and then transferred to the original mesh.

Stoll [21] introduced a volumetric approach to mesh deformation using Laplacian editing which has the advantage of preserving the internal mesh volume. This approach first performs a constrained tetrahedralization of the mesh vertices to obtain a volumetric representation. Laplacian differential coordinates of the mesh are then used to represent the shape and internal volume. Given the target displacement of a subset of vertices the approach solves a Laplacian system for a mesh which satisfies the target vertex constraints and minimizes the change in volume of the tetrahedra. This approach allows volumetric mesh deformation by manipulation of a subset of vertices. In this paper we integrate the volume deformation scheme into a conventional skeleton based character animation pipeline. This allows skeleton driven animation of highly detailed captures meshes from 3D surface scans without problems of mesh collapse associated with widely used surface based animation techniques.

3 Approach to Skinning

In this paper we combine volumetric mesh deformation [21] with widely used Linear Subspace Deformation [14] to enable volumetric skeleton driven character animation. Volumetric mesh deformation requires that a number of vertex locations on the desired pose are known. These are used to solve for the remaining unknown vertices. We propose a simple means to provide these constrained vertex locations with the inclusion of a skeleton in the character to be animated. At each of the joints a number of vertices are selected to be attached to that joint. The motion of these vertices is controlled by linear skin blending.

Deformation requires that first the mesh is tetrahedralized. This is achieved via constrained Delaunay tetrahedralization where all tetrahedra are inside the mesh. The attachment of a small subset of mesh vertices at each joint can then be made using standard mesh skinning tools. Reposing the skeleton provides new target locations for each of the attached vertices. The deformation process iterates a linear Laplacian deformation step, the extraction of rotational components and update of the differential coordinate representation of the mesh in order to approximate non-linear transformation components.

3.1 Skeleton and Vertex Attachment

The tetrahedral character mesh is initially fitted with a standard inverse-kinematic (IK) skeleton in its neural pose. This character mesh can be produced by either manual modeling or from 3D surface reconstruction as with the character in Figure 1.

The tetrahedral deformation step of our algorithm requires a
number of desired vertices locations to be known. In order to derive this from the motion of the skeleton a number of vertex are selected to be directly connected to the skeleton using linear skin blending. Figure 1, image C shows an example of these constraints for a human character and image D shows those points in a there reposed positions.

3.2 Linear Skin Blending

Linear Skin Blending (LSB) or sub-space deformation has been established as the industry standard approach to skinning [14]. Due to its simple linear nature linear skin blending provides very fast real-time deformation even for large scale meshes with a vast number of vertices. This technique is implemented in a number of animation packages.

The basis of this method revolves around computing the transformation of a vertex as a weighted combination of the transformations of the related joints. Each of the vertices is first assigned a weighted set of joints \( \{1, \ldots, n\} \) and then the vertex transformation is computed as a weighted combination of the transformations \( T_j \) of influencing joints \( j \). Each vertex is initially transformed rigidly by each of its influencing joint transformations and then a weighted linear combination of these transformations is used to produce the final deformed vertex position, \( S(v_0) \).

\[
S(v_0) = \left( \sum_{j=1}^{n} w_j T_j \right) . v_0
\]  

(1)

This technique is used to specify the motion of a small number of user selected vertices. These vertices provide the constraints for a volumetric approach to character deformation. This procedure gives the animator the ability to control the volumetric deformation by manipulation of the skeletal rig either directly by changing joint angles or with standard IK solver by manipulating end-effector positions. The use of LSB to animate a small number of vertex positions is highly efficient and provides constraints for the volumetric deformation.

3.3 Volumetric Deformation

The volumetric deformation technique [21] is a Laplacian editing technique which operates on a tetrahedral mesh rather than the traditional triangular mesh [23, 30]. Such a volumetric technique offers the advantage of minimizing the change in volume of the tetrahedral elements preventing mesh collapse. This overcomes limitations of triangular mesh editing and LSB animation methods where the mesh collapses due to bending and twisting (the candy wrapper effect). This style of deformation allows meaningful extrapolation of new poses without the need for a large number of example poses as in [23].

We produce the tetrahedral mesh from a traditional triangular representation by constructing a constrained Delaunay tetrahedralization [19]. Tetrahedra are constrained to lie inside the volume of the original surface mesh. The tetrahedralization is constructed such that the minimum number of additional vertices are added maintaining low computational complexity.

Laplacian deformation involves fixing a number of vertex locations and solving for the others by fitting the Laplacian (differential coordinates) of the new geometry to the differential coordinates of the original mesh:

\[
Lx = \delta
\]  

(2)

where \( L \) is the Laplacian operator matrix, \( x \) is a vector of the mesh’s vertices stacked \((x_1, \ldots, x_n, y_1, \ldots, y_n, z_1, \ldots, z_n)\) and \( \delta \) is the differential coordinates of mesh.

The shape functions of a tetrahedron defined in Laplacian (local) coordinates are given by equations 3 through 6...
where $P_i$ are the four vertices of the tetrahedron. The gradient operators $G_i$ can be combined to form one large $9n \times 3m$ sparse matrix for a mesh with $n$ tetrahedra and $m$ vertices. The Laplacian operator $L$ is defined from the connectivity of the mesh and is given by:

$$L = G^T DG$$

(8)

where $D$ is the diagonal degree matrix, in this case a diagonal matrix of tetrahedral volumes. The differential coordinates of the geometry can be calculated from equation 2. Computing the set of unknown deformed vertex locations, $x_u$, involves solving equation 2 factored according to a number of constrained known vertex locations, $x_k$. The rows and columns of $L$ corresponding to the known vertex locations are removed and the differential coordinates of the known locations factored into the right hand side yielding:

$$x_u = \arg \min_{x_u} \| Lx_u - (\delta + c) \|$$

(9)

where $\delta$ is the differential coordinates of the original mesh and the vector $c = Lx_k$ is the result of multiplying the Laplacian operator $L$ by the vector of known constrained vertex locations $x_k$. Each element of $x_k$ is defined by equation 10 considering a mesh $M$ with vertices $(v_1, ..., v_n)$. Solving for $x_u$ gives the location of the deformed set of vertices.

$$x_k[i] = \begin{cases} v_i & \text{if } v_i \text{ is known} \\ 0 & \text{otherwise} \end{cases}$$

(10)

### 3.4 Rotation Interpolation

It is well known that linear interpolation of large rotations causes unnatural deformations. To overcome this an iterative approach is taken in which equation 9 is solved repeatedly whilst updating the differential coordinates on each iteration. After each iteration the transformation $T_i$ for each tetrahedron $t_i$ with original vertices $V = (v_1, ..., v_4)$ and transformed vertices $V' = (v'_1, ..., v'_4)$ is computed as:

$$T_i = \begin{bmatrix} v_1 - v_4 & v_2 - v_4 & v_3 - v_4 \\ v'_1 - v'_4 & v'_2 - v'_4 & v'_3 - v'_4 \end{bmatrix}^{-1}$$

(11)

These transformations are factorised into rotation $R_i$ and scale/shear $S_i$ components using the polar decomposition [20].

$$T_i = R_iS_i$$

(12)

At each iteration these rotations are applied independently to their corresponding tetrahedra from the original mesh. An updated set of differential coordinates is computed by application of equation 2 and equation 9 solved accordingly. These steps of linear Laplacian deformation and differential coordinate update are repeated until convergence.

### 3.5 Efficiency

To achieve interactive rates equation 9 must be solved several times within a fraction of a second. To achieve this we use a highly optimised sparse LU solver as implemented in UMFPACK [7]. UMFPACK makes use of BLAS and requires a highly optimised BLAS implementation. For this purpose we make use of the go-toBLAS [24] which provides a vendor non-specific implementation of BLAS combining high optimisation with portability of code.

Once the constrained vertices have been selected and the Laplacian operator re-factorised accordingly then the LU factorisation can be precomputed and reused at each iteration. Deformation occurring from subsequent motion of the skeleton only requires re-computation of the LU factorisation if the animator alters the vertex constraints.

### 4 Results

We have implemented our skinning algorithm as a plug-in for Autodesk’s Maya. This allows a model to be set up and skinned using the standard Maya interface and animation tools. This implementation allows an animator to manipulate a mesh with all the functionality of a common animation package.
permitting our algorithm to work seamlessly with an existing animation pipeline. The Maya interface allows the user to set up a skeleton and skin the character according to our approach. Motion of the skeleton within Maya can be controlled by direct user manipulation, IK or predefined animation curves to promote volumetric deformation of the character mesh.

Figure 2 demonstrates the two well know problems that occur with linear skin blending and shows how our approach responds in the same situation. Under a full 360 degree axial twist the linear skin blending approach almost entirely collapses whereas minimal collapse occurs with our tetrahedral approach. With a 90 degree bend linear skin blending again shows a large amount of collapse. The tetrahedral method eliminates the mesh collapse.

Figure 3 further demonstrates this affect where the shoulder is seen to collapse under linear skin blending but not with our tetrahedral deformation technique. Figure 3 also highlights another key advantage to our approach. Linear skin blending requires weighting of each of the vertices appropriately, which although greatly simplified with tools such as Maya’s weight painting interface, is still a time consuming process. The linear skin blending result shown in Figure 3 could be improved with manipulation of the weights. The same automated weight generation is employed for the constrained points in our algorithm. Figure 3 visually identifies the advantages of our approach under these conditions. With no manual manipulation of the weights our tetrahedral deformation produces far more visually pleasing results. With our approach automatically generated weighting is sufficient to produce aesthetic deformation results.

Figure 4 shows an example animation produced with the skeleton driven volumetric deformation approach. Motion capture data of a person dancing is re-targeted onto the Stanford Armadillo character with 96,987 vertices [12]. The character has been skinned according to our tetrahedral algorithm. As the skeleton moves according to the re-targeted animation curves our deformation techniques deforms the character accordingly. Animations can also be produced from keyframing joint positions and orientations. Figure 5 shows a character reconstructed from multiple view video capture of a person. The character has been interactively reposed into a number of different poses. These are produced by repositioning the underlying skeleton. These images demonstrate the versatility of our approach when reposing a character. Figures 4 and 5 present a wide range of character poses in which there is no sign of collapse even with large alterations to the reference T-pose. Here we have applied our technique to the animation of characters from different production sources (3D video capture and laser scan data). Our technique is extremely flexible and functions well on characters produced using any approach.

Table 1 and the corresponding graph in Figure 6 show the time taken for deformation with respect to the number of vertices and tetrahedra. The graph demonstrates quadratic complexity in relation to the number of vertices. The timings were produced on a Intel Q6600 CPU with 4Gb of RAM. The deformation time includes the time to run the entire iterative deformation procedure until convergence. The timings illustrate our ability to produce interactive deformation with up to a couple of thousand vertices.

5 Conclusion
We have presented an approach which combines the advantages of volumetric mesh based deformation with
<table>
<thead>
<tr>
<th>Vertices</th>
<th>Tetrahedra</th>
<th>Time (secs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>182</td>
<td>1648</td>
<td>0.03</td>
</tr>
<tr>
<td>682</td>
<td>6784</td>
<td>0.26</td>
</tr>
<tr>
<td>1502</td>
<td>14576</td>
<td>0.44</td>
</tr>
<tr>
<td>2642</td>
<td>26434</td>
<td>1.06</td>
</tr>
<tr>
<td>4102</td>
<td>40304</td>
<td>1.35</td>
</tr>
<tr>
<td>13002</td>
<td>128816</td>
<td>6.63</td>
</tr>
<tr>
<td>25467</td>
<td>352352</td>
<td>38.0</td>
</tr>
</tbody>
</table>

Table 1: Deformation time for meshes with the specified number of vertices and tetrahedra

The methods interactive computation time and use of a standard Inverse Kinematic skeleton give simplicity to the animator and provide easy integration with existing tools. Implementation as a plug-in for Autodesk’s Maya gives the user the ability to interact easily with the algorithm using standard rigging and animation tools. This interface allows the user to select any subset of vertices to provide constraints for the tetrahedral deformation algorithm. The plug-in works with a traditionally skinned character using an inverse kinematic skeleton and Linear Skin blending. This allows integration of high-resolution captured 3D models into the standard animation pipeline. From a traditionally skinned character a user need only select the subset of vertices to be moved by linear skin blending for volumetric deformation of the mesh. Unlike linear skin blending automated weight generation for the linear skin blending is generally good enough to produce visually very pleasing results.

Currently we are limited to a interactive deformation with only a couple of thousand vertices using a single thread implementation of the algorithm. Multi-threaded CPU or GPU implementation of the approach is likely to achieve interactive rates for meshes an order of magnitude larger.

The most interesting area of future work revolves around an extension along the lines of the work of Sumner et al. [23]. Extending this tetrahedral deformation technique to use multiple example meshes to represent the space of meaningful deformation could lead to increased artistic control whilst maintaining the advantages of skeletal animation and volumetric mesh deformation.

References


Figure 4: Dancing animation produced from re-targetting motion capture data onto the Stanford Armadillo character skinned according to our technique. The first image shows the original T-pose of the character.

Figure 5: A character mesh deformed into a number of different poses using our tetrahedral approach to skeleton based animation.