Eurographics/ ACM SIGGRAPH Symposium on Computer Animation (2006)
M.-P. Cani, J. O'Brien (Editors)

Sketching Articulation and Pose for Facial Animation

Edwin Chang[1] † and Odest Chadwicke Jenkins[1] ‡

[1]Department of Computer Science, Brown University

Abstract

We present a method for articulating and posing meshes, in particular facial meshes, through a 2D sketching interface. Our method establishes an interface between 3D meshes and 2D sketching with the inference of reference and target curves. Reference curves allow for user selection of features on a mesh and their manipulation to match a target curve. Our articulation system uses these curves to specify the deformations of a character rig, forming a coordinate space of mesh poses. Given such a coordinate space, our posing system uses reference and target curves to find the optimal pose of the mesh with respect to the sketch input. We present results demonstrating the efficacy of our method for mesh articulation, mesh posing with articulations generated in both Maya and our sketch-based system, and mesh animation using human features from video. Through our method, we aim to both provide novice-accessible articulation and posing mesh interfaces and rapid prototyping of complex deformations for more experienced users.

Categories and Subject Descriptors (according to ACM CCS): I.3.6 [Computer Graphics]: Methodology and Techniques-Interaction Techniques I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling-Geometric Transformations

1. Introduction

Articulating and posing are both tasks inherent to the animation of 3D meshes. Defining the **articulation** (or rigging) of a mesh traditionally involves specification of several deformation variables over the range of desired motion. To achieve satisfactory results, a user may need to manually specify deformation settings for hundreds of vertices. Furthermore, an infinite number of plausible deformations can exist for a given mesh that range from the realistic flexing and extending of underlying muscle to cartoon squash and stretch motion. Consequently, articulation is often a tedious and complex process requring substantial technical as well as artistic skill. This problem is compounded when defining the articulation of a facial mesh, where motion is quickly discernable as natural or unnatural to a human viewer.

Once articulation is performed, an animator creates animations by specifying **poses** of the mesh in the articulation space. To specify a pose efficiently, an animator is often provided with a control rig comprised of widgets and sliders that provide a puppet-like control of the mesh deformation. Unfortunately, users face a considerable learning curve to understand and utilize such control rigs, often requiring as much time as creating the control rig itself.

To address the lack of accessiblity in current rigging systems, we aim to leverage the familiarity of 2D sketching as an interface for 3D mesh animation. While current articulation and posing interfaces provide detailed control, such interfaces lack intuition and accessiblity for novices and traditional animators trained with pencil and paper. A sketching interface, however, provides a familiar interface while still providing a high level of control to users. It can be particularly helpful to a novice who lacks a strong understanding of facial movement but is comfortable working with simple line drawings of a face. For traditional animators, sketching provides a direct correlation between hand drawn and 3D animation.

In this paper, we present a 2D sketching interface to facilitate procedures for articulating a single mesh and posing an articulated mesh. Our method focuses on the inference of

† edwin_chang@alumni.brown.edu
‡ cjenkins@cs.brown.edu

Figure 1: *A reference curve (green) and target curve (blue) are sketched to pose the lower lip of an articulated mesh in our posing system.*

reference and target curves on the mesh from user sketch input. In our posing procedure, the user first draws a sketch to place a reference curve on the mesh. The user then draws a sketch to identify a target curve, which specifies the desired manipulation of the reference curve. We then use the downhill simplex method to optimize mesh pose in the articulation space such that the distance between these two curves is minimized. The user can additionally introduce additional constraints to pin parts of the mesh in place. In our articulation procedure, reference curves are generated from the sketching of regions of interests on the mesh that are then manipulated by sketched target curves.

Both our articulation and posing methods can work in tandem or independent such that one can be replaced by alternative mechanisms. For example, we show results from use our posing method with a mesh articuated by blend-shapes modeled in Alias Maya. While our methods are suited to facial meshes, these same procedures are applicable to other types of meshes as well.

2. Related Work

There exists much previous work in mesh articulation and deformation, as well as the related field of mesh editing. One typical approach for facial articulation is to create several meshes with the same topology and blend between them, i.e. a blend shape approach. While robust and granting users a high amount of control, this approach often requires users to create a large number of blend shapes. The blend shape process has also been combined with a skeletal approach to provide the flexibility of a skeletal system with the expressiveness of a blend shape system [LCF00]. Shapes have also been used as examples for a combination of shape and transform blending [SRC01]. We seek to maintain that level of expressiveness in our method without requiring users to create additional shapes. We do use a simple blend shape method of 15 shapes, however, to test the ability of our posing process to work with several types of articulation systems. Our articulation system requires only one shape, which users articulate.

Free-Form Deformation (FFD) [SP86] is one method that provides a wide range of possible deformation without requiring multiple shapes. Our work parallels the use of FFDs, in particular a curve-based FFD method that warps the mesh [SF98, CJTF98]. This type of FFD provides a method of smooth deformation that facilitates the use of curves sketched by users. Sketches have also been used to specify FFDs based on scalar field manipulation [HQ03] and as input to a gesture based FFD interface [DE03]. Outside of FFDs, sketches have also been used as skeletal strokes [HL94] to bend and twist 2-dimensional images.

Recent work has also focused on drawing directly onto the image plane in order to specify deformation. This poses challenges in interpreting the intent of users as well as providing a coherent translation from 2D to 3D space. This problem has also been encountered in 3D modeling using 2D sketches. One modeling method interprets 2D sketches as silhouettes to infer and construct 3D polygonal shapes [IMT99]. A similar approach uses implicit surfaces, allowing for surface modification by silhouette oversketching [KHR02]. Another solution used for mesh editing is to treat the process as an inverse NPR process [NSACO05], where the mesh is transformed to match user-drawn contours and silhouette contours. This method of interpretation of user-drawn curves is very natural to users, and while our approach differs we aim to replicate its functionality. Silhouette sketches additionally also been used to stylize previously animated motion by specifying the silhouette contours of the desired pose [LGXS03]. Sketches have also been used to specify curves in a free-form skeleton method [KG05], but the approach was limited to deformation in appendage-like parts of a mesh, e.g. tails, legs, or arms. We extend this approach, in particular its use of reference and target curves, to work with enclosed areas of a mesh, which is necessary for facial articulation.

Often one limitation of drawing on the image plane is that deformations remain parallel to the image plane. We approach this by constraining vertices to follow a surface similar to the method used in manipulating clothing [IH02], where cloth can be positioned by moving it across surface of a mesh.

Posing an articulated mesh involves its own unique challenges separate from those encountered in articulation. Control widgets are often added that allow users to interact directly with the articulation parameters. Sketching has been applied to manipulate multiple control points in order to pose the mesh [SD04], but these control points must have been previously defined by a user. Sketches have also been used to describe the motion of a figure across time rather than through individual poses [TBvdP04]. Other work has

treated the posing problem as an optimization problem, attempting to determine the pose of a human figure that best matches hand-drawn sketches [DAC*03]. Our work in posing also views the problem as an optimization problem but focuses on posing articulated facial meshes. Posing a facial mesh has been approached previously using a blend shape method [CB02], but required users to build a set of key blend shapes instead of using a previously created set. Other work has applied inverse kinematics to sets of existing blend shapes [SZGP05], allowing users to interactively pose between the blend shapes. Our posing method also works with blend shape approaches, but is versatile enough to work with many types of articulation methods (including our own).

One of the methods of evaluation we use for our posing process involves the use of curves generated from the tracking of facial features in video. We use the eigen-points approach [CB96] in order to determine these curves. This approach uses an eigen-feature based method in order to place control points onto unmarked images, which we then use to define curves for posing. Other work has also used video with facial models, creating high resolution models of a moving face that can be used to pose new expressions by interactively dragging surface points [ZSCS04].

3. Approach

While our posing and articulation methods are independent, they share a simple interaction scheme based on sketching reference and target curves. Reference and target curves are 3D features on the mesh inferred from a 2D sketch. These curve features are used in a slightly different manner for the articulation and posing processes. Our articulation method uses the additional selection of regions of interests to compute reference curves. The posing procedure incorporates user-selected constraints to keep parts of the mesh in place. In the remainder of this section, we explain further these differences in the computation of feature curves from sketching.

3.1. Sketch-based Mesh Posing

Given an articulated mesh, posing that mesh presents challenges in estimating the appropriate articulation parameters. Our approach casts pose parameter estimation as an optimization problem. We apply our optimization engine to an articulated mesh based on blend shape interpolation as well as one from our sketch-based articulation method.

The user first draws a reference curve C_r, an ordered set of points $\{\mathbf{r}_1,...,\mathbf{r}_n\}$. Each of these points is projected from the camera onto the closest visible face of the mesh (Figure 2) and stored as weights of the vertices of that face. As the mesh is deformed, we recalculate new 3D positions for these points based on the vertex weights so that the curve follows the surface of the mesh during deformation. The user then draws a target curve C_t, an ordered set of points $\{\mathbf{t}_1,...,\mathbf{t}_n\}$,

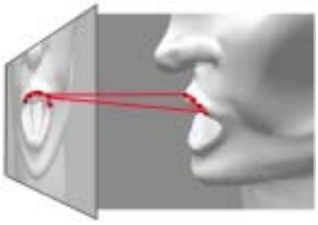


Figure 2: *A reference curve drawn on the image plane is projected onto the mesh*

which is projected onto the mesh. We reverse the order of the points of C_t if the target curve's endpoint, $\mathbf{t}_n$, is closer to the reference curve's start point, $\mathbf{r}_1$, than the target curve's own startpoint, $\mathbf{t}_1$ (i.e. reverse C_t if $\|\mathbf{t}_n - \mathbf{r}_1\| < \|\mathbf{t}_1 - \mathbf{r}_1\|$. We then reparameterize the target curve to match n, the number of points in the reference curve. We reparameterize the curve by choosing the new points by distance along the original line, where $\mathbf{r}'_i$ is the i-th of n points along the reparameterized curve:

$$\mathbf{r}'_i = C_r\left(\frac{i-1}{n}\right) \tag{1}$$

The target curve is then projected into 3D space using the distances from the camera along the reference curve C_r. Our system then searches the articulation space M^d of d deformers to find an optimal pose P given by the optimal articulation parameters $\mathbf{x}$ that minimizes the distance between the reference curve, which maintains its position on the mesh, and the target curve (Figure 1). The distance term for the optimization is given by the following, where $\mathbf{r}_i$ and $\mathbf{t}_i$ are corresponding points on the reference and target curves for a given pose P in an articulation space of d dimensions.

$$E(P) = \sum_{i=1}^{n} \|\mathbf{r}_i - \mathbf{t}_i\| \tag{2}$$

In order to solve this optimization problem, we use the downhill simplex method [PFTV92], which gives us the ability to perform optimization without the use of derivatives. Since this is the case, the optimization process does not need knowledge of the underlying articulation system and can work with any type of articulation. The downhill simplex method searches a d-dimensional space using a simplex shape of $d+1$ points that searches the space by reflecting and contracting itself until it reaches its goal (Figure 3). The optimization works best with non-hierarchical articulation (like faces, rather than arms), however, and is only efficient for a limited number of variables ($d < 20$). We propose methods to deal with this limitation in our discussion section.

Using the downhill simplex method, we determine we

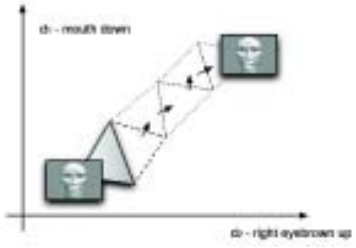


Figure 3: *Searching in a two-dimensional articulation space using the downhill simplex method*

have reached an acceptable solution when the vector distance travelled in one iteration is less than a fractional tolerance of 0.05. After we have found this solution, we perform a cleanup stage. Since several of the articulation parameters may have had no effect on the region of interest, these parameters may have become unneccessarily changed through searching the articulation space in the optimization process. We evaluate a pose P_i for each articulation variable x_i, where x_i is set to its original value and all other variables are set to those from $\mathbf{x}_o$, the set of articulation variables derived from optimization. If the difference between $E(\mathbf{P}_i)$ and $E(P_o)$ (where P_o is the pose set by $\mathbf{x}_o$) is minimal, we return x_i to its original value.

Our system also provides a method for the user to set constraints on the mesh with additional curves in order to keep parts of the mesh in place. Each constraint curve K_j, a set of ordered points $\{k_1,...,k_n\}$ is projected onto the mesh. When a pose is evaluated using equation 2, the following term is also added for each constraint, where k_i' is the position of k_i in the new pose P.

$$E_j(P) = \sum_{i=1}^{n} \|\mathbf{k}_i' - \mathbf{k}_i\| \tag{3}$$

Constraint curves are useful as a deformer on a mesh may have a small effect on the region the reference curve lies on even though it mainly deforms a separate area. For example, a cheek deformer could slightly affect the vertices around the lips on a mesh. When the user attempts to pose the lips the cheeks could then be undesirably affected. These constraints are drawn on the mesh in the same manner the reference curve is. Previously used reference curves can also be used as constraint curves in order to keep previously specified deformation in place.

3.2. Sketch-based Mesh Articulation

Users specify one articulated deformation at a time in our system in a 4-step process. Users first select a region of general interest, then a region of specific interest. Users can then draw reference and target curves to specify the deformation. Each of these deformations becomes one dimension in the articulation space. The 4 steps are pictured in Figure 4.

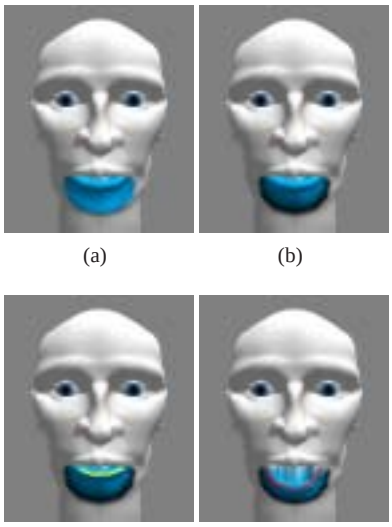


Figure 4: *(a) A region of general interest is selected and then (b) a region of specific interest to specify articulation weights. (c) A reference curve and (d) target curve are then drawn to specify the deformation.*

3.2.1. Regions of Interest

In the first step, the user must pick a general region on the mesh where the deformation is desired, a set of vertices V_g. To do so, the user first draws a curve C_g to encircle the region on the image plane, selecting a set of vertices V_a. The desired set of vertices will be a subset of the set of all selected vertices ($V_g \subseteq V_a$), but often $V_g \neq V_a$ as some vertices in V_a may be behind the desired region or occluded by other parts of the mesh. In order to avoid selecting these vertices, V_g is chosen to be the set of connected vertices containing the vertex closest to the camera and of sufficient size ($|V_k| > 10$, where $V_k \subseteq V_a$ and V_k is connected). Each vertex in this set ($\mathbf{v}_i \in V_g$) is then projected to the image plane in order to determine its 2D distance to the drawn curve C_g, which is then stored. We will call this distance g_i for every vertex $\mathbf{v}_i$ in V_g.

The user then encircles a region of specific interest with a new curve C_s to specify a set of vertices V_s, where $V_s \subseteq V_g$. Each vertex $\mathbf{v}_i$ in V_g is then assigned articulation weights by the following equation, where w_i is the articulation weight and c_i is the distance to the curve C_s on the image plane:

$$w_i = \begin{cases} 1.0 & \text{if } \mathbf{v}_i \in V_s, \\ \dfrac{g_i}{g_i + c_i} & \text{otherwise.} \end{cases} \tag{4}$$

In this manner, articulation weights smoothly blend off from 1.0 to 0.0 from the region of specific interest to the borders of the region of general interest. Our system displays the articulation weights to users by coloring vertices white if unselected, and blue to black from 1.0 to 0.0. The camera is restricted from movement in this step as g_i and c_i are 2D distances calculated on the image plane. Using different camera views when selecting the two regions may result in an undesirable blend of articulation weights. This camera restriction only exists at this step.

3.2.2. Estimating a Reference Curve

Our system then generates a reference curve C_r, an ordered set of points $\{\mathbf{r}_1,...,\mathbf{r}_n\}$, that estimates a skeletal curve in 3D space for the region of specific interest. The curve C_r is determined by ordering the vertices in V_s by their x values when projected to the image plane, where x and y refer to the horizontal vertical axes on the image plane. If the difference in the minimum and maximum y values of the vertices when projected to the image plane in V_s is larger than the minimum and maximum x values, the y value of each vertex is used to order C_r instead. This 3D curve is then smoothed through convolution with a triangle filter reference $f(j)$ across a kernel size v that is $1/3$ of the number of vertices in the curve ($v = |C_r|/3$):

$$\mathbf{r}'_i = \sum_{j=-v/2}^{v/2} f(j)\mathbf{r}_{i+j} \tag{5}$$

In some cases this estimated curve may not be satisfactory to the user, especially when the region of interest does not have a distinct curve-based feature, like the cheek of a face. If desired, the user can redraw the curve C_r on the image plane, which is then projected onto the mesh to form the reference curve. The reference curve, either estimated or not, is then slightly smoothed to account for noise (convolved with a triangle filter reference) and reparameterized to have a regular spacing.

With the reference curve C_r smoothed and reparameterized in 3D space, the user can choose to move the camera and view the mesh at different angles. In order to facilitate this, the system does not depth test when rendering curves, instead overlaying them over the entire image.

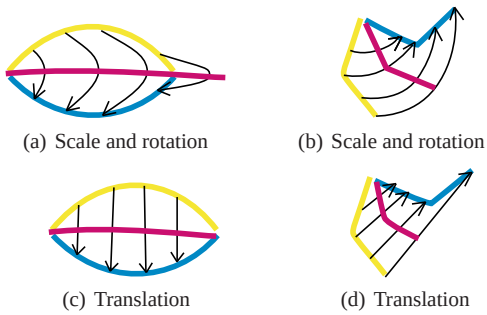
In the final step, the user draws a target curve C_t, an ordered set of points $\{\mathbf{t}_1,...,\mathbf{t}_n\}$, indicating how the mesh should be deformed so that the reference curve meets the target curve. The order of the points of the curve is reversed if the target curve's endpoint, $\mathbf{t}_n$, is closer to the reference curve's start point, $\mathbf{r}_1$, than the target curve's own startpoint, $\mathbf{t}_1$ (i.e. reverse C_t if $\|\mathbf{t}_n - \mathbf{r}_1\| < \|\mathbf{t}_1 - \mathbf{r}_1\|$. The target curve is then reparameterized to match the number of points in the reference curve, n. The points of the target curve are then projected into 3D space by using the distances to the camera of the corresponding points on the reference curve, d_1 to d_n.

3.2.3. Curve Interpolation

Since the target and reference curves now share the same number of points, we can determine rotations between the matching line segments on the two curves by finding the cross product of the two segments and the angle between them. We will call these rotations ϕ_j for each segment j. The system stores these rotations as relative for each segment, such that each rotation assumes all rotations previous to a line segment have been applied. By keeping rotations relative, we can determine partial rotations between points on the curves when we perform the mesh deformation. The system also calculates a scale change s_i between the two segments, as the two curves may be different lengths:

$$s_i = \frac{\|\mathbf{t}_{i+1} - \mathbf{t}_i\|}{\|\mathbf{r}_{i+1} - \mathbf{r}_i\|} \tag{6}$$

With the rotations and scales for each segement of the lines, the system can then interpolate between the curves by applying a partial transformation α (where $0 \leq \alpha \leq 1$) of $\alpha\phi_j$ and αs_i to the line segments of the reference curve.



(a) Scale and rotation (b) Scale and rotation

(c) Translation (d) Translation

Figure 5: *Different methods of curve interpolation, where the initial curve is yellow, the in-between curve is red, and the final curve is blue*

In certain situations, however, applying scale and rotation to interpolate is inappropriate. The curves in Figure 5(a) and 5(b) are interpolated using the rotation-scale method. In Figure 5(b), this works well, especially if these lines pose an appendage like a leg. In Figure 5(a), however, the curve becomes undesirably extended, which would be inappropriate if these curves were posing a blinking eyelid. For this case, we instead linearly interpolate between corresponding points on the two curves, translating the points without regard to scale or rotation of the line segments (Figure 5(c). Our system automatically chooses this method if the endpoints of the reference and target curves are within 10 pixels of each other, but also allows the user to specify otherwise.

3.2.4. Mesh Deformation

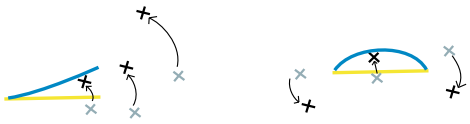
Once the system has an appropriate method of interpolation between the reference and target curves, it can deform the vertices of the mesh according to those curves. Each vertex $\mathbf{v}_i$ in V_g is projected onto the reference curve to find the

closest point on that curve, which is then stored as a proportional distance along the length of the entire curve, l_i, where $0 \leq l_i \leq 1$. This projection is done on the image plane in 2D space so that vertices farther from camera than other vertices still project to an appropriate reference point $\mathbf{r}_i$. The system then determines the corresponding point on the target curve, which we will call the target point $\mathbf{t}_i$, by the distance along the target curve according to l_i. We then apply the translation from the reference point to the target point $(\mathbf{t}_i - \mathbf{r}_i)$ to the vertex. We must also apply a rotation transformation to the vertex centered around the target point. Since this point does not likely lie on the end of a line segment on the curve, we must calculate the rotation.

Our system first combines all the line segment rotations previous to the target point, ϕ_j from 1 to $k-1$, where the target point lies on segment k. We then apply a partial rotation of the last line segment's rotation, ϕ_k, according to the length along on that segment the target point lies, a value from 0 to 1 we will call u. We express this in the following equation, where the final rotation is ϕ_t. The rotations are centered about the target point.

$$\phi_t = \left(\prod_{j=1}^{k-1} \phi_j \right) (\phi_k u) \tag{7}$$

In order to pose between the reference and target curves, the system applies the same operations, but instead uses an interpolated curve, determined using the method described in Section 3.2.3, instead of the target curve. For similar reasons discussed concerning curve interpolation, rotations are not always desired in the mesh deformation. In Figure 6, a mesh deformation with rotations on three vertices is depicted in two examples. Rotations are appropriate for Figure 6(a), but less so for Figure 6(b), especially if this were deforming an eyelid. Vertices past the endpoints of the curves can move greater distances than expected due to rotation. For this reason, when curve interpolation does not use rotations, we do not apply them in mesh deformation as well.



(a) Deformation with rotation

(b) Deformation with undesirable rotation

Figure 6: *Different examples of mesh deformation with rotation*

Since deformations are specified using curves in the image plane, it can be difficult to specify deformation outside of one plane of movement. We approach this problem by allowing the user to set the deformation to follow the surface of a sphere. In Figure 7, the deformation is constrained to maintain the vertices' distance from the eyeball sphere. Otherwise, the vertices move in only one direction and the eyelid intersects the eyeball. While we use only a sphere in our approach, other surfaces could be used by constraining the vertices to maintain their original distance from the surface by projecting outward from the normal of the closest point on that surface.

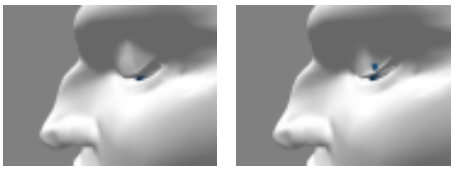


Figure 7: *An eyelid deformation constrained and not constrained to follow the eyeball surface*

Once all the vertices have been repositioned according to the target curve, they are returned back to their original positions according to the value of the articulation weights determined previously. The transformation is calculated as a linear translation for each vertex, where vertices with weight 0 return completely to their original position and vertices with weight 1 stay in their new position. In this manner we can ensure smooth deformations even when the region is enclosed by other parts of the mesh.

4. Application: Animation from Video Features

We additionally used our posing process with curves generated from tracking of facial features in video on our sketch-based articulated mesh. These curves were determined through the eigen-points method [CB96] and follow the eyebrows, eyelids, and lips of the subject in the video. These tracked curves, while slightly noisy, remain unfiltered in our testing. Since the facial features of the subject do not match those in the 3d mesh, relative changes in the tracked curves are applied to user-drawn reference curves to create new target curves. For one curve from video C_{vf} in frame f, relative changes, $\{\mathbf{c_1}, ..., \mathbf{c_n}\}$, from frame to frame for each point were determined. These relative changes were then applied to a user-drawn curve C_u reparameterized to have n points, $\{\mathbf{d_1}, ..., \mathbf{d_n}\}$. For each frame of video a new curve C'_u was determined by applying $\mathbf{c_i}$ to every point $\mathbf{d_i}$. The change $\mathbf{c_i}$ was also scaled up in order to account for difference in length between C_{v0} and C_u:

$$\mathbf{d_i}' = \mathbf{d_i} + \mathbf{c_i} \frac{|C_u|}{|C_{v0}|} \tag{8}$$

5. Results

We begin with the mesh of a face and articulate it using our system to specify a deformation for each eyelid, eyebrow,

Figure 8: *Posing using curves from tracking of facial features in video*

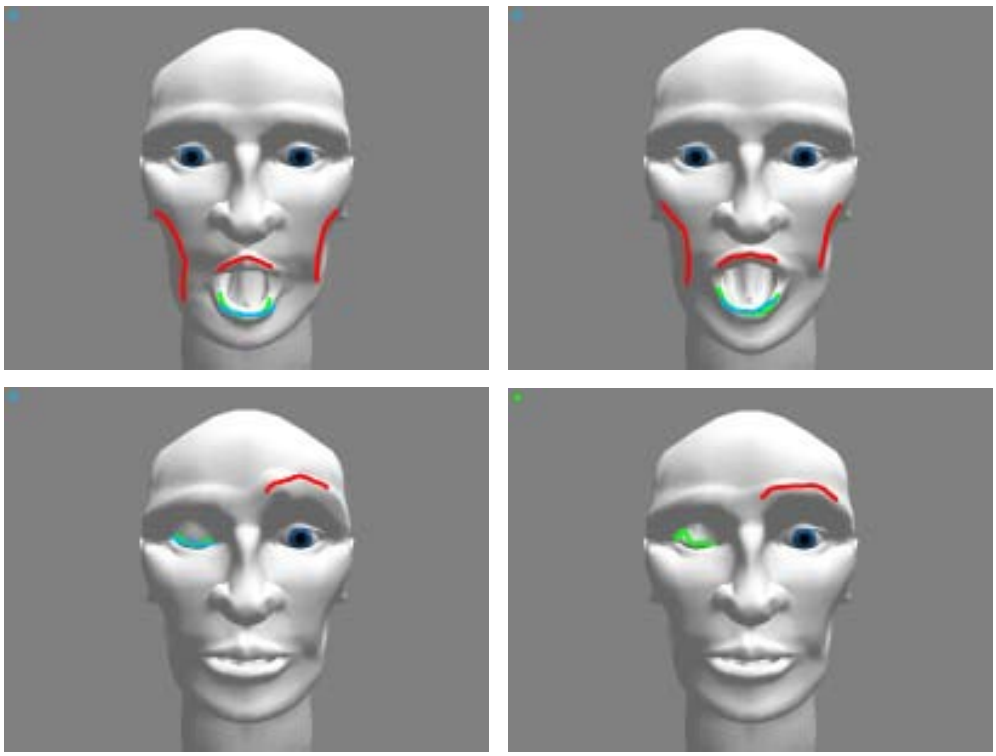


Figure 9: *Posing using similar curves on a sketch-based articulation (left) and a blend shape articulation (right)*

Figure 10: *An articulated mesh colored according to deformer and articulation weights*

cheek, jaw, and various movements of the lips (Figure 13) for a total of 15 deformers. Figure 10 depicts these deformers as separately colored regions that fade to black according to articulation weights. Figure 11 shows some of the poses that can be acheived using this articulation. Each of these deformations was created quickly, in under 2 minutes for each. By comparison, specifying similar deformations in a blend shape approach required 10-20 minutes per shape. For eyelid deformations, we specified the deformation to follow the surface of a sphere centered at the eye. Our system also works for non-facial meshes, like the trunk of an elephant (Figure 12).

Figure 11: *A sampling of poses in our articulation space*

Bringing this articulated mesh into our posing system, we can pose the face using reference and target carves. We also test our posing system with a mesh articulated by a blend shape method using shapes created in Alias Maya (Figure 9) and achieve similar results in both. In the top examples, we pose the mouth in two iterations, one for the upper lip and one for the lower lip. In total, with the cheek constraints, we drew 6 curves to pose the face (2 constraint curves, and 2 pairs of reference and target curves). In the lower examples we posed the right eyelid and left eyebrow using 4 curves (2 pairs of reference and target curves).

When posing the face from video features, the limited articulation of the mesh does not fully match the range of expression in the human face. Even so, the posing process works well at capturing the motion of the face across frames (Figure 8).

The optimization process for posing requires many iterations before convergence and results in a pause in the posing system after drawing the target curve. On a AMD XP 2000+ processor, this pause is under 5 seconds for the blend shape method and under 10 seconds for our articulation method. The optimization takes longer for our articulation method because it takes slightly longer to pose than the blend shape method. From pose to pose this time is small (the mesh can be posed at ($\sim$50 fps), but is still longer than the blend shape method ($\sim$90 fps).

6. Discussion

Our implementation allows users to quickly sketch out a wide range of articulations for a mesh. Several additions could be added to the system to allow for more complex movements, such as combining separately defined deformations, but our work focuses on establishing a simple system suited to novices or prototyping deformations for more complex systems.

We also maintain a level of control in our deformations comparable to blend shape approaches (Figure 9). Furthermore, we do not face the limitations blend shapes have, such as the issues linear blending between shapes can cause. For example, it is difficult to have rotational movement with blend shapes, like an eye blink or jaw movement. Our method can recreate these motions. With our method, we are able to apply the strengths of free form deformation to enclosed areas of the mesh while maintaining smooth deformation.

Our posing process is likewise easy to use and requires little to no training or knowledge of the articulation system. Through sketching curves to pose the mesh, the user has intuitive control over the articulation space while unaware of the actual articulation parameters. Usability testing is needed to further substantiate the intuitive nature of this approach.

The optimization required for posing is not instantaneous and does not work at interactive rates (>15 Hz), though it is fast enough for user interaction (typically within 5 seconds per pose). Further limitations involve the limit of number of variables the optimization process can deal with. The downhill simplex method is only effective to under 20 variables and a large number of variables will further slow down the optimization. As many professional facial animation systems often have several hundred controls, this method may be impractical. We can reduce the problem, however, by limiting the articulation search space only to those articulation variables that affect the reference curve. If the search space still remains overly large, our method can be used in stages, first posing articulation variables that have a large effect and then smaller variables in greater detail. Another possible approach would be to create low-dimensional subspaces [SHP04] or probabilistic priors [GMHP04] from previously generated poses.

7. Conclusion

In this paper, we presented a sketch-based method of preparing a mesh for animation in two processes - articulation and posing. Our system focused on inference of reference and target curves for facial animation, but was adept at animating motion for meshes with different kinematic structures. This system has a simple and intuitive interface that allows for a wide span of deformation to be specified. Our approach to posing is flexible for usage with typical articulation and rigging systems, including blend shape interpolation.

References

[CB96] COVELL M., BREGLER C.: Eigen-points. In *IEEE International Conference on Image Processing Proceedings* (1996), pp. 471–474.

[CB02] CHUANG E., BREGLER C.: *Performance Driven Facial Animation using Blendshape Interpolation*. Tech. rep., Stanford University Computer Science, 2002.

[CJTF98] CORRÊA W. T., JENSEN R., THAYER C., FINKELSTEIN A.: Texture mapping for cel animation. In *SIGGRAPH 98 Proceedings* (1998), pp. 435–446.

[DAC*03] DAVIS J., AGRAWALA M., CHUANG E., POPOVIĆ Z., SALESIN D.: A sketching interface for articulated figure animation. In *SIGGRAPH/Eurographics Symposium on Computer Animation Proceedings* (2003), pp. 320–328.

[DE03] DRAPER G. M., EGBERT P. K.: A gestural interface to free-form deformation. In *Graphics Interface 2003 Proceedings* (2003), pp. 113–120.

[GMHP04] GROCHOW K., MARTIN S. L., HERTZMANN A., POPOVIĆ: Style-based inverse kinematics. In *SIGGRAPH 04 Proceedings* (2004), pp. 522–531.

[HL94] HSU S. C., LEE I. H. H.: Drawing and animation using skeletal strokes. In *SIGGRAPH 94 Proceedings* (1994), pp. 109–118.

[HQ03] HUA J., QIN H.: Free-form deformations via sketching and manipulating scalar fields. In *ACM Symposium on Solid modeling and Applications Proceedings* (2003), pp. 328–333.

[IH02] IGARASHI T., HUGHES J. F.: Clothing manipulation. In *ACM Symposium on User Interface Software and Technology Proceedings* (2002), pp. 91–100.

[IMT99] IGARASHI T., MATSUOKA S., TANAKA H.: Teddy: a sketching interface for 3d freeform design. In *Graphics Interface 2003 Proceedings* (1999), pp. 113–120.

[KG05] KHO Y., GARLAND M.: Sketching mesh deformations. In *Symposium on Interactive 3D Graphics and Games Proceedings* (2005), pp. 1142–1147.

[KHR02] KARPENKO O., HUGHES J., RASKAR R.: Free-form sketching with variational implicit surfaces. In *Computer Graphics Forum* (2002), pp. 585–594.

[LCF00] LEWIS J., CORDNER M., FONG N.: Pose space deformation: a unified approach to shape interpolation and skeleton-driven deformation. In *Conference on Computer Graphics and Interactive Techniques Proceedings* (2000), pp. 165–172.

[LGXS03] LI Y., GLEICHER M., XU Y.-Q., SHUM H.-Y.: Stylizing motion with drawings. In *ACM SIGGRAPH/Eurographics Symposium on Computer animation* (2003), pp. 309–319.

[NSACO05] NEALEN A., SORKINE O., ALEXA M., COHEN-OR D.: A sketch-based interface for detail-preserving mesh editing. In *SIGGRAPH 05 Proceedings* (2005), pp. 1142–1147.

[PFTV92] PRESS W. H., FLANNERY B. P., TEUKOLSKY S. A., VETTERLING W. T.: *Numerical Recipes in C - The Art of Scientific Programming*. Cambridge University Press, 1992, pp. 408–412.

[SD04] SWAIN M., DUNCAN B.: Sketchpose: Artist-friendly posing tool. SIGGRAPH 2004 Sketch, 2004.

[SF98] SINGH K., FIUME E.: Wires: a geometric deformation technique. In *Conference on Computer Graphics and Interactive Techniques Proceedings* (1998), pp. 405–414.

[SHP04] SAFANOVA A., HODGINS J. K., POLLARD N. S.: Synthesizing physically realistic human motion in low-dimensional, behavior-specific spaces. In *SIGGRAPH 04 Proceedings* (2004), pp. 514–521.

[SP86] SEDERBERG T. W., PARRY S. R.: Free-form deformation of solid geometric models. In *Conference on Computer Graphics and Interactive Techniques Proceedings* (1986), pp. 151–160.

[SRC01] SLOAN P.-P. J., ROSE C. F., COHEN M. F.: Shape by example. In *Symposium on Interactive 3D Graphics Proceedings* (2001), pp. 135–143.

[SZGP05] SUMNER R. W., ZWICKER M., GOTSMAN C., POPOVIĆ J.: Mesh-based inverse kinematics. In *SIGGRAPH 05 Proceedings* (2005), pp. 488–495.

[TBvdP04] THORNE M., BURKE D., VAN DE PANNE M.: Identifying and sketching the future: Motion doodles: an interface for sketching character motion. In *SIGGRAPH 04 Proceedings* (2004), pp. 424–431.

[ZSCS04] ZHANG L., SNAVELY N., CURLESS B., SEITZ S. M.: Spacetime faces: high resolution capture for modeling and animation. In *SIGGRAPH 04 Proceedings* (2004), pp. 548–558.

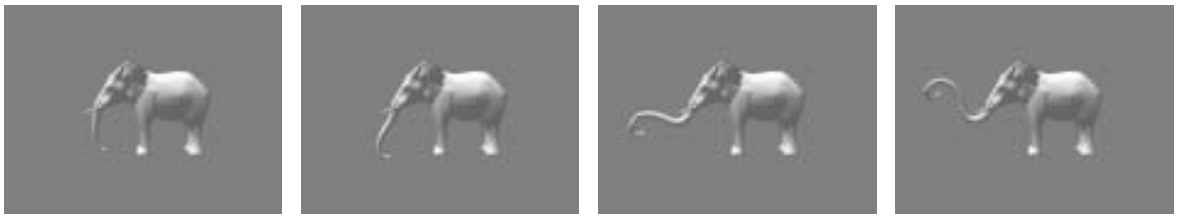


Figure 12: *Deformation of an elephant's trunk using the articulation system*

Figure 13: *A few of the deformations created in the sketch-based articulation system*