Character Animation using Hierarchical B-splines

David R. Forsey

The challenge of building and animating computer generated human and animal figures is complicated by the need to smoothly and realistically control the deformation of the surface around the articulations of the underlying skeleton. This paper reviews approaches to surface modelling for character animation and describes a geometric (as apposed to physically-based) approach to character modelling using an extension to the *hierarchical B-spline*. This techinique provides differential attachment of the surface to the skeleton and allows multi-resolution control of surface deformation during animation. The attachment mechanism is simple, easy to use, inexpensive, extensible and can drastically reduce the effort required to animate a surface. The techniques introduced are illustrated using examples of human and animal forms.

1. Introduction

The animation of computer generated characters is a complex and difficult task that demands both an effective way to control the motion of the figure, and an effective represention of the figure itself. An antagonistic struggle occurs between *detail* and *effort* during the design and creation of an animated figure, Effort is a nebulous concept, but in this paper is considered as proportional to the number of values required to specify all the degrees of freedom during an animation sequence. A DOF that is constant requires no effort. A DOF that requires an animator to specify 30 key values for one second of animation requires more effort than the same animation specified with the 4 control points of cubic B-spline curve. This definition does not attempt to address the question of how those values are determined nor how difficult it was to determine them.

Detail corresponds to the number of degrees of freedom (DOF) in the figure. Detail is increased through the use of digitized images for surface texture, or by increasing the number or type of geometric primitives in the figure to better represent a desired surface appearance or shape. Adding detail to a figure can increase its versimillitude, but it can also vastly increase the effort to animate the entire figure because of the possible need to control every extra degree of freedom, for every frame of the animation, 30 times per second, 1800 times a minute.

Current research in animation, whether it be on dynamics, kinematics, free-form deformations or motion control, is rooted in the need to handle this ‘‘DOF problem’’ [12].

Computer representations for surfaces are varied, but typically are based upon polygons, splines, or implicit surface formulations. As animators strive for more realism, the demands placed upon the surface representation rapidly increase, particularly in those regions around the face and surrounding the joints. Whatever the representation, the creation of an animated surface for an articulated figure involves the DOF problem and the animator must weigh the cost of increasing the complexity of the surface against the cost of animating the added degrees of freedom.

This paper addresses the DOF problem in the specification and animation of surfaces for articulated figures. The following section outlines the problems inherent in the representation of integument in general, and current approaches to surfaces for character animation.

The Integument

The integument (skin or hide) of an animal is a wonderously complex organ – an extremely elaborate system with intricate behaviour arising from the physical properties [34] of the material itself (visco-elastic, non-homogeneous and anisotropic) and also from its relationship to the underlying fascia of muscle, bone and connective tissue comprising the skeleto-muscular system. The degree on interconnection can range from almost nothing where the skin just slides over the facia as around parts of the knee, to regions where the skin is tightly connected to the underlying fascia, such as the glabrous skin of the hand. The nature of the integument varies from species to species and between individuals within a species according to age and/or genetic factors.

The mechanical behaviour of the integument spans the scale from very small wrinkles to the broad-scale folds and creases and, The former arising directly from the composition and properties of the integument itself while the latter arises indirectly from the arrangment of the underlying skeleto-muscular system and its attachments. To further complicate the situation, an animator often does *not* want to copy the behaviour of a physical system, but requires a charicature, exaggeration or embellishment of real life.

To adequately serve as a general model of the integument for character animation, the surface representation must be able to capture the essence of whatever the behaviour the animator is striving for. It is most likely that no single representation will be able to serve in all situations.

Surface Representations

Polygonal models are still the most common form of surface representation. However, polygonal models of the integument attempt to model a basically continuous surface with linear approximations requiring many polygons to obtain a visually smooth surface. (Of course the number of polygons needed to maintain this illusion changes depending upon the viewpoint). Increasing the number of vertices in the model increases the number of points that have to be animated around the joints.

Implicit surfaces [56] offer an interesting approach white approximates a smooth surface and provides a mechanism for building a surface around a joint between two segments. However, implicit surfaces are still very computationally intensive and it is not yet known how to specify or control the behaviour of the joint deformations during animation.

Tensor-product splines offer smooth continuous surfaces, but have limitations arising from their rectilinear nature. (Some of these limitations are addressed in [78]). Triangular surface patches while having the potential to construct surfaces of arbitrary topology, still have problems as a modelling tool because of an unreasonable distribution of curvature over each patch and the lack of reasonable subdivision algorithms would seem to limit the usefulness of b-patches [9 , and S-patches10] for character animation. Nevertheless, spline surfaces are currently the most flexible representation with the best potential for use in character animation because of the amount of control provided in the definition of a smooth surface.

Character Animation

For computer character animation a wide variety of techniques have been used. Perhaps the cheapest and most widespread is the interpolation method where a number of separate surface models are created, each representing a different pose (or facial expression) of the figure. The surface is animated by interpolating between a pair of models. [11].

The sheer number of points and poses involved make surface animation through shape interpolation a tedious process, encouraging the development of methods that automatically modify the vertices as the joint changes angle. Thalmann [1213] uses a procedural model called a *joint-dependent local deformation* (JLD) whereby the polygons around the joints of the hand are modified to maintain a realistic appearance. Each JLD is specific to a particular type of joint and is non-trivial to construct.

Chadwick [14] combines a multi-vertex editing technique called *free-form deformations* (FFD) [15] with a procedural model in the **Critter** system where the user interacts with the control vertices of a deformation lattice instead of the polygonal model to change the general shape of the polygons as the joint angle changes. Chadwick employs a procedural model, similar to the one used by Komatsu [16] for a Bezier-based body model, where particular nodes in the deformation lattice are associated with the bisection of the joint angle and where inter-penetration of the lattice is prevented. The fine details of the surface are represented in the polygonal surface model (a continuous spline surface could just as easily be used) while the broad-scale changes are handled by the deformation lattice. FFD's have found a welcome home in commerical animation system precisely because they allow control of a complex shape through the manipulation of a relatively small number of points. However their use in character animation is not adequately demonstrated by Chadwick who does not provide a good example of a realistic joint deformation.

FFD's are very useful for changing the overall shape of an object, but because they simply distort space, they do not use any knowledge of surface geometry. The FFD's strength, and its weakness, lies in its indifference to the geometry of the imbedded figure. Deformations can be made regardless of the complexity of the surface and with relative ease, but only because the space encompassing the figure is deformed without regard to any intrisic geometric relationships that the figure might have. For example, Figure ## shows a simple surface deformed using an FFD. While the resulting shape may be a useful effect, it is rarely what is required. Multiple FFD's might be used to control the stretching on the right side of the figure, but non-trivial coordination between overlapping FFD's would be required. Indeed, Pacific Data Images uses a system of interconnected deformation lattices in their animation system to compensate for this behaviour of FFD's in their animation **Locomotion**. [17].

Nahas [18] uses a cubic B-spline as the surface representation for face and body. A fine mesh of patches provides the surface definition while broad-scale changes are accomplished by embedding the control vertices of the model in a coarser spline mesh. This broad-scale spline is created by selecting a relatively few *characteristic points* from the original B-spline model of the surface. When the characteristic points are moved, it displaces the vertices of the model itself. This mechanism is used to control facial expression, and although mention was made of a similarly defined body model, no details about the joints were provided.

Another approach to the complexity problem is to have the animator specify some ‘‘high-level’’ action while a program deals with the extra degrees of freedom. These *procedural models* exist for facial muscles [1920521] and physically-based simulations [22133] for the particular body regions. Physically-based simulations tend to be complex to configure, computationally intensive, and prone to numerical instabilities. Though the usefulness of simulation is clear, it may not be appropriate for the majority of animations which require the rapid construction and animation of a variety of characters.

Digitized motion has been used to drive computer models for a number of years. The motion can be recovered from video images or the model can be driven directly by ‘‘waldos’’ that record body motion directly.2324]. An intriguing extension of this technique uses images (motion) to drive the shape of the animated figure

This paper describes an extension to the *hierarchical B-spline* [8] where a hierarchical surface is attached to an underlying skeleton in such a way that the animator has control over the location and scope of the surface deformation. This is performed without the use of external data-structures (such as an FFD) and realistic joint deformation are achieved without resorting to joint-specific procedural deformations. This surface representation retains the editing capabilities and compact representation of the hierarchical spline and allows both small and large-scale shape deformations controlled by the change in the joint angles of the underlying skeleton. These features combine to make animating surfaces for character animation faster and easier to do.

2. Character Modelling with traditional spline surfaces

This section reviews and extends the concept of a hierarchical spline surface. [8] A familiarity with tensor-product spline surfaces [25] is assumed.

Two complications, relating to knot insertion, appear when using tensor-product spline surfaces (such a B-splines) for free-form surface design. As the designer builds the surface, the need arises to increase the number of degrees of freedom (i.e. the number of patches) to achieve the desired shape. In splines, this is accomplished by inserting knots into the formulation increasing the number of patches. In a traditional spline surface, this has two effects upon the design process:

a) Because of the tensor-product nature of the formulation, knot insertion acts on the entire surface - either an entire row or an entire column of patches is created. The designer cannot add patches locally, and patches distant from the area of interest will be split where additional patches are niether desired or useful. (So-called ''local'' refinement is possible in composite surfaces constructed from a set of Bezier patches, but this approach has its own drawbacks and limitations. See [26]) For example, the relatively simple surface in Figure Worst requires 1024 control vertices.

b) Because of the *local support* property of the basis functions, knot insertion affects how the designer is able to change the shape of the surface in the region where the knot was added. Figure **a shows the region of a bicubic B-spline affected by the manipulation the indicated control vertex. Figure **b shows the reduced region of the surface affected after knot insertion. Thus if the designer wishes to once again manipulate the extended region of Figure **a, additional methods (such as FFD's or multi-vertex editing techniques) must be employed. Local support is, in general, a very useful property. However, it makes it difficult for the interactive surface designer to make broad-scale changes in surface shape, and because knot insertion is non-local, the patch structure may be imposed in direct conflict with the designers goals.

These two characteristics complicate the use of splines for the design of static surfaces, but have an even greater effect in surfaces used in animation where each additional degree of freedom may need to specified for each frame of the animation, and where uncontrolled addition of patches may interfere with previously defined procedural shape definitions. In contrast, hierarchical B-splines exhibit *local knot insertion*, and allow *multi-resolution manipulation* over surface shape.

Hierarchical Surfaces

The general form for a hierarchical tensor-product spline surface $\mathbf{S}(u, v)$ is defined by a series of *overlays* L^o, each with a set of *control vertices* $\mathbf{V}^o_{i,j}$ and *basis functions* $B^o_{i,k}(u)$, $B^o_{j,el}(v)$ of some polynomial order k and el, respectively,

$$\mathbf{S}(u, v) = \sum_{o} \sum_{i} \sum_{j} \mathbf{V}^{o}_{i,j}\, B^{o}_{i,k}(u)\, B^{o}_{j,l}(v) \tag{1}$$

Some or all of of the control vertices may be absent at any particular level of overlay. The underlying basis functions can be of any order, but must exhibit both local support and have a refinement algorithm. Note that the hierarchical approach is not limited to spline surfaces; it may also be applied to polygonal mesh models, or extended to tensor-product volumes in three parametric variables [268].

In a hierarchical surface, control vertices are added only in the regions where they are required and not across the entire surface, thus patches that would normally be split by the refinement process, remain whole. A hierarchical spline surface (such as Figure Worst where the patches are coloured according the level of refinement) develops as a patchwork of various sized patches. This addresses one of the two complications with refinement described above, namely non-local knot insertion. The other complication: the loss of broad-scale shape control after insertion, is addressed by *offset referencing*.

Offset Referencing

Equation (1) is a representation of a *static* surface in terms of a sum of overlay surfaces. Although useful, this form does not address the construction, modification or animation of that surface. Offset referencing allows the modification of any overlay within the hierarchical surface without affecting the continutity.

The control vertices within each overlay $\mathbf{L}^o$ are represented not as absolute positions, but as vectors defined in a particular frame of reference. In this "reference-plus-offset" notation, each control vertex $\mathbf{W}_{i,j}$ of any part of the surface, whether root-level parent surface or overlay at any level of refinement, is written in the form

$$\mathbf{W}^{o}_{i,j} = \mathbf{R}^{o}_{i,j} \oplus \mathbf{O}^{o}_{i,j} \tag{2}$$

The point $\mathbf{R}^{o}_{i,j}$ is called the *reference position* of the vertex and is derived from the parent overlay through refinement. The $\vec{\mathbf{O}}^{o}_{i,j}$ is the *offset vector* at level $\mathbf{L}^o$ and corresponds to the $\mathbf{V}^{o}_{i,j}$ of Equation 1. The final world-space postion of the control vertex $\mathbf{W}^{o}_{i,j}$ is obtained by combining the reference position with the offset. The $\oplus$ function is arbitrary and can be as simple as vector addition.

An offset technique that has proven to be very useful is the *tangent plane offset* where the $\mathbf{R}^{o}_{i,j}$ determines not only a position in space, but also a "frame of reference" defined by the position of the control vertex and tangent plane and the normal to the surface at the point on the surface maximally influenced by that control vertex. The offset vector is *relative* to this coordinate system and as the coordinate frame changes (due to changes in the shape of one of the parent overlays), the final position of the control vertex also changes. Editing changes to level $\mathbf{L}^o$ of a surface revise the $\mathbf{R}^{o+1}$'s, and through these recursively alter the final position of the $\mathbf{W}^{o+1}$'s. Editing changes to level $\mathbf{L}^o$ are recorded entirely as changes to the $\mathbf{O}^o$'s. In the case of the root (base-level) surface, the offsets are defined in some world-space coordinate system.

Thus as one particular level of overlay changes, the "details" upon that surface (lower level overlays) follow the change in the surface shape. Figure 2a shows a bi-cubic hierarchical spline that has been altered from its original planar configuration at several levels of overlay. The colour of the surface is lighter in regions with finer overlays. A single control vertex moved near the root of overlay structure results in Figure 2b illustrating how the details of the surface follow the change of the underlying surface. Another editing operation, this time with a finer scope, produces the surface in Figure 2c. In Figure 2d, the surface has been edited to produce changes at the finest levels of detail. In Figures 2e and 2f the previous broad-scale edits were undone. An important feature of the offset method is that the shape deformations occur at the chosen level of detail while retaining the results of more detailed editing and, as illustrated in Figure 2, are *order independent*. Local support extends between levels of overlay in that alterations at one level of detail will not alter the the positional relationships between control vertices located some distance (determined by the order of the basis) down in the hierarchy.

This operation extends the behaviour beyond that easily attainable with the deformation techniques of Cobb [27], Barr [28], or Sederberg [15], or other multi-vertex editing techniques involving spatial transformations of the control vertices in a single coordinate frame. For example, Figure 9 illustrates how a 2D free-form deformation might operate on a shape similar to that in Figure 2. The *reference*⊕*offset* operation is not a simple function of 3D position in a bounded volume.

3. Spline Surfaces over Skeletons

Typically, to create an articulated computer animated figure, the separate parts of the figure are each designed in their own local coordinate frame and the frames are connected hierarchically together to form the complete object. Thus when the transformations connecting the local coordinate systems together are changed, the figure changes its position and pose [29]. Because the limbs are defined separately, if nothing if done to modify the shape, the separate surfaces will simply inter-penetrate. This behaviour is satisfactory for many applications, but not for realistic human figures where the surface must seamlessly span the joints.

If the surface spans adjacent limbs, the surface definition must also span the coordinates spaces defining those limbs. Thus it becomes more complicated to use a surface defined in a local coordinate frame because some of the surface definition resides elsewhere and that information must be either transformed into the local frame, or the local information transformed into some common coordinate frame before the shape of the surface can be determined. This is not a particularly difficult problem, it is simply unfortunate because the hardware support (and the software libraries that exploit that hardware) in most graphics workstations today is tailored specifically to deal quickly with a hierarchy of separately defined objects.

Simply spanning the articulations with a surface does not solve the joint problem - as the joint angle becomes more acute, the surface collapses upon itself (See Figure ***). The problem lies in controlling how the model spans the joint. As described in Section # above, numerous approaches to this problem have been proposed and utilized in the literature. The next section shows how a simple extension to the hierarchical B-spline surface formulation can be used to control joint deformations.

Articulated Hierarchical Surfaces.

The hierarchical B-spline formulation described above has been used as the basis of a powerful and flexible design environment for free-form surfaces. As part of the process of extending this editor to deal with character animation, a facility to define and manipulate a hierarchically arranged ‘‘skeleton’’ was added allowing the user to interactively create, connect and manipulate simple polygonal objects that represent the coordinate frames of each limb in the underlying skeleton of an animated figure. The skeleton can be created before any surface, added to an existing surface, or incrementally modified at any time. Newly created surfaces are initialized in the world coordinate frame which corresponds to a ‘‘root’’ skeletal segment.

To aid in the creation of surfaces, the editor is based upon direct manipulation of points on the surface [83031]. When a point on the surface is selected, the region affected is high-lighted. Moving up or down in the hierarchy affects a larger or smaller region of the surface, and the region around the selected point(s) can be refined to add more patches to the region. The selected point (or points) can be moved to change the shape of the surface. The control vertices that determine the position of the selected point(s) are modified automatically by the editor, but are not visible to the user.

The basic mechanism for all surface animation is termed *attachment*, and operates on the control vertices implicitly specified during the selection process. Once the user has selected a region (by picking points and selecting the appropriate level in the hierarchy - the same action used for editing the surface) the user interactively picks a skeletal segment ($\mathbf{F}_n$) and chooses the ‘‘Attach’’ menu item. Visually, nothing happens (unless the user has chosen to colour the surface representation according to the attached frame of reference). Internally, the surface data is restructured to alter the offset information of the affected control vertices so that after attachment, when the selected skeletal segment moves, so do its attached control vertices.

In the basic hierarchical form, the offsets in Equation (2) are represented in terms of a coordinate frame determined by the parent overlay. In an *attached* vertex the frame of reference ($\mathbf{R}_{i,j}$) becomes that of the skeleton segment, and the offset ($\mathbf{O}_{i,j}$) is recalculated such that the final position $\mathbf{W}_{i,j}$ remains unchanged in world space.

When the underlying skeleton is animated, the coordinate system of each skeletal segment changes position and orientation. For each attached vertex, a new final position ($\mathbf{W}_{i,j}$) is calculated using the skeletal segment's reference frame and the offset information. This, in turn, affects overlays defined at lower levels in the hierarchy which must be recalculated through the standard refinement and offset referencing technique described in Section 2. This process is both fast and numerically stable.

Attachment provides the user with the ability to control the nature and extent of the deformation by choosing the level in the surface hierarchy where attachment is performed. If broad-scale control is required, attachment is made nearer the root level in the hierarchy. If a finer-scale effect is needed, attachments are made at a fine-scale level. The more attachments made in a particular region, the more tightly that region will follow the movement of the attached skeletal segment. Any region can be made to move rigidly with a particular segment simply by attaching a sufficient number of control vertices in that region.

We return to the collapsing tube to show how the attachment mechanism is used to control the deformation around a joint (an elbow in this case). In Figure Elbow1, 24 surface points were selected (using an area select function) and attached to the lower segment of the arm. When the elbow bends, the surface collapses as expected. In Figure Elbow2, the region around the elbow has been locally subdivided and five vertices (the three visible verticies are marked) at this overlay level attached to the upper arm segment. Attachment over-rides the influence of the parent overlay and alters the shape of the deformation. Note that in regions affected by unattached vertices, the collapse of the parent overlay still influences the shape creating the dimple region near the vertex marked "A" in Figure Elbow2. This dimple is removed by attaching the vertex to the upper arm and altering the position of all the attached vertices to improve the shape of the arm when bent (Figure Elbow3). Finally, the overall shape is edited to more closely resemble a human elbow (Figure Elbow4).

TODO Example of a tube.

-Perhaps the best way to convey the modelling paridigm is to show how
-a simple spline cylinder behaves when bent.
- low level attach over-ride broad-scale changes.
- show example of same refinement pattern but all attachments at a lowest level.
- note that as more cv's in a region are attached, the more rigid the movement.

Multiresolution attachment speeds up the animation process because the user does not have to position an FFD around the surface, nor decide what lattice size to use.

The hierarchical nature of the attachment mechanism aids the animation process by providing control over the surface at the appropriate level of detail. Consider a more complicated example, the jaw and cheek regions of the dragon's head in Figure Dragon1, that was animated by attaching the lower jaw of the dragon head to the mandible of the skeleton with about six user actions, i.e. A single point at the tip of the lower jaw was selected, and the overlay level chosen such that the influence of the affected control vertex covered the entire jaw and cheeks almost to the eyes. Thus the entire cheek region will change shape when the jaw moves (Figure Dragon2). Regions that have not been attached, but are still affected by the jaw, deform according to any previously applied offset method. Thus the dewlap of the jaw (the serrated fringe running along the ventral side of the head), does **not** have to be animated separately, the tangent plane offset ensures that those details follow the broad-scale changes in surface shape. Certain details, notably the lower teeth (which turned inwards) and the tip of the jaw (which didn't appear firm enough) deformed when they should have moved solidly along with the mandible. This was corrected by attaching a few more control vertices to the jaw segment − one for each tooth and a few at the tip of the jaw, to effectively lock down these particular regions.

The amount of effort required to animate a surface will, of course, vary considerably with the situation. The entire dragonhead figure - including animated nose, eyebrows, nostrils, earflaps and neck, uses only 239 attachments. Since the editor enforces bilateral symmetry when requested, and many of these attachments were made on selected regions (rather than individual points), the dragon head, composed of over 3000 patches, was animated using less than 120 user actions.

Multi-segment Joints

Several of the approaches dealing with surfaces around joints [121416] exploit the bisection of the joint angle to aid in coaxing the surface around the joint in a reasonable manner. Rather than use a separate procedural mechanism to affect the control vertices, the hierarchical attachment mechanism can be used directly if we define each joint as having several *sub-segments* that subtend the angle. Each sub-segment is a full-fledged reference frame to which control vertices are attached. The number and length of the sub-segments is selectable during the construction of the skeleton. Each sub-segment subtends an equal proportion of the total joint angle, thus all joints with an odd-number of sub-segments will allow attachment of control vertices which follow the bisection of the joint angle. The multi-segment joint allows the designer to proportionally deform the surface around the joint using the same attachment mechanism used for the rest of the surface without recourse to special purpose code.

4. Extensions: Procedural Offset Methods

Hierarchical attachment does not preclude the use of other techniques to control the shape of the surface. Indeed the system that created the images in this paper provides a number of additional techniques: shape interpolation between offset positions, and a parametric animation capability that maps a parameter value (which can relate to joint angle) to keyframes consisting of a set of either offset positions or lengths of offset vector. The animating parameter can also be a vertex position, so that one vertex can ‘‘follow’’ another like a rudimentary cellular automaton. None of these techniques conflict with the attachment mechanism. This section describes an example of how a procedural method may be combined with attachment to add secondary action to kinematically specified motion.

Secondary Action using Simple Dynamic Simulation

Simulation of physical properties is increasingly used to augment the description and behaviour of surfaces in computer animation, but a comprehensive modelling of the dynamics of the system may be difficult or impossible to derive or compute efficiently.

Simple dymanic systems, such as Chadwick’s [14] add dynamic behaviour to surface models through the deformation lattice by assigning point masses as the control vertices of the lattice and interconnecting everything with springs. When the system is accelerated the point masses react and the lattice itself is altered thereby changing the shape of the body model.

One problem with this approach, or any approach that treats the integument as a simple set of sprung masses hanging in space, is that the perturbed system acts like a blob of jello. One solution, of course, is to increase the complexity of the simulation, using stiffer springs with better damping within a more highly interconnected lattice [22] interacting with a model of the underlying bone and musculature. This is perhaps overkill for most character animation, though with the recent advances in both processor speed and algorithms for elastic deformations, this approach will most certainly grow in use.

Pacific Data Images’ approach to the problem was to simplify it. In their *Goop* system [23] a point mass with spring/damper is attached to every polygon vertex in the model. As the system is subjected to acceleration, the mass moves away from its rest position. This new position is used as the polygon vertex for during rendering. Since the point masses are not interconnected, the new positions can be calculated at very little cost. However, since the points are unconstrained except for the attachment point, the model will still exhibit the ‘‘jello’’ effect.

Rather than using a mass/spring/damper model to affect the positions of the control vertices, an equally simple model simulates the behaviour of a mass on the end of a stiff (though flexible) rod (Figure 8). Such a system is equivalent to a forced planar mass/spring/damper combination with a spring constant $\mathbf{k}$ given by $\mathbf{k} = \frac{3El}{l^3}$, where El is the *flexural rigidity* and l is the rod length.

To determine l and the orientation of the rod (and thus the plane in which the forces act on the mass), the overlay and offset referencing structure of a hierarchical spline is once again used directly. For a particular vertex $\mathbf{W}_{i,j} = R_{i,j} + O_{i,j}$, the rod is embedded at the reference point and directed along the offset vector with the mass residing at $\mathbf{W}_{i,j}$. When the system is accelerated the mass changes position and alters the final position of the control vertex. This operation can also be applied at any level of the hierarchy to control the extent of the resulting deformation. Figures Swing1 and Swing2 shows secondary action of the earflaps of the dragon during kinematic animation of the head itself. Dynamics have been added to features defined either through tangent offsets, or attached offsets.

Examples

Attachments at a broad-scale affect the broad-scale behaviour of the surface in such regions as the cheeks, neck of the dragon (Figure 4), about the thigh of the knee (Figure 6) and around the knuckle (Figure 7). Fine-level attachments constrain these broad-scale effects by anchoring down the surface at particular point. These were used for the teeth within the dragon's mouth, to control where folds occurred in the dorsal region of the knee and on the ventral surface the knuckle joint.

Control vertices that are unattached to any particular segment react indirectly to the changes brought about by skeletal movement through offset referencing. Thus the dewlap of the dragon follows the broad changes in the shape of the neck and the folds on the top of the knuckle flatten as the finger bends – even while the folds of the interphalangal crease become more pronounced.

5. Future work.

In the current system, attachment forms the basis of all animation, but allows the definition of coordinate frames that are not part of the basic skeleton, i.e. in the face to move surface features (smiling dragon).

Finally, just as Chadwick equates the deformation lattice with a finite element model, future research will explore the possibility of using multi-resolution finite element models (which also exhibit a hierarchical structure) to control the dynamic behaviour the of surface for such features as skin folds and creases.

attaching to frenet frames of space curves.

more constraints (distance to skeleton, vertex/vertex separation, bisection plane testing.

Since the motion of the tip of the rod traces an arc, large displacements will at some point begin to warp the surface unrealistically. Because the hierarchy provides at every level of overlay a frame of reference at every point on the surface (depending on the order and nature of the basis functions used), more complicated constraints may be added to enhance the behaviour. (We note that such constraints are still simple when compared to those of a deformable model.) For example, the displacement of the mass can be forced to follow the reference surface with differential damping more forcefully opposing motion into the surface than motion along it.

6. Conclusion

The hierarchical spline is a space efficient data structure that provides a flexible editing interface to spline surfaces where the scope of the editing operation is controllable and does not deform the inter-relationship between small details within the scope of the deformation. This same structure can be directly applied to the problem of covering articulated figures with a smooth continuous surface (including folds and creases) and integrates well with procedural animation and a primitive dynamic simulation. The attachment mechanism is integrated into the hierarchical form, requires no external data structure and does not add significantly to the storage requirement of the surface. The surface is attached to the skeleton by a simple user interface requiring no programming. Skill is needed in the sculptor's hands (where it should be) through choice of skeletal segment, vertex, and level of detail.

References

1. David Zeltzer, ''Motor Control Techniques for Figure Animation,'' *IEEE Computer Graphics and Applications*, vol. 2, no. 9, pp. 53-60, November, 1982.
2. D. Zeltzer, ''Toward an Integrated View of 3-D Computer Animation,'' *The Visual Computer*, vol. 1, no. 4, pp. 249-259, April 1985.
3. Steve Pieper, ''Physically-Based Animation of Facial Tissue for Surgical Simulation,'' State of the Art in Facial Animation, SIGGRAPH'89 Tutorial Notes, ACM, 1989.
4. V. Wright, ''Elasticity and Deformation of Skin,'' in *Biophysical Properties of Skin*, ed. H.R. Elden, p. Ch.12, Wiley-Interscience, New York, 1977.
5. David R. Hill, Andrew Pearce, and Brian Wyvill, ''Animating speech: and automated approach using speech synthesis by rules,'' *The Visual Computer*, vol. 3, pp. 277-289, 1988.
6. Jules Bloomenthal and Ken Shoemake, ''Convolution Surfaces,'' *Computer Graphics*, vol. 25, no. 4, ACM, Proceedings of SIGGRAPH '91, July 1991.
7. Marie-Paule Gascuel, ''Welding and Pinching Spline Surface: New Methods for Interactive Creation of Complex Objects and Automatic Fleshing of Skeletons,'' *Proceedings Graphics Interface'89*, pp. 20-27, Edmonton, June 1989.
8. D. R. Forsey and R. H. Bartels, ''Hierarchical B-Spline Refinement,'' *Computer Graphics*, vol. 22, no. 3, pp. 205-212, Atlanta Georgia, August 1988.
9. H.-P. Seidel, ''Alogrithms for B-patches,'' *Graphics Interface '91*, June 1991.
10. Charles Loop and Tony DeRose, ''Generalized B-spline Surfaces of Arbitrary Topology,'' *Computer Graphics*, vol. 24, no. 4, ACM, Proceedings of SIGGRAPH '90, August 1990.

11. P. Bergeron and P. Lachapelle, “Controlling Facial Expression and Body Movements in the Computer Generated Short ’Tony de Peltrie’,” SIGGRAPH ’85 Tutorial Notes, 1985.

12. N. Magnenat-Thalmann, R. Laperriere, and D. Thalmann, “Joint-Dependent Local Deformations for Hand Animation and Object Grasping,” *Proceedings Graphics Interface ’88*, pp. pp. 26-33, Edmonton, June 1988.

13. Jean-Paul Gourret, Nadia Magnenat-Thalmann, and Daniel Thalmann, “Simulation of Objects and Human Skin Deformations in a Grasping Task,” *Computer Graphics*, vol. 23, no. 4, pp. 21-30, July, 1989.

14. John E. Chadwick, David R. Haumann, and Richard E. Parent, “Layered Construction for Deformable Animated Characters,” *Computer Graphics*, vol. 23, no. 4, pp. 243-252, Boston, July 1989.

15. T. Sederberg and S. Parry, “Free Form Deformations of Solid Geometric Models,” *Computer Graphics*, vol. 20, no. 4, 1986.

16. Koji Komatsu, “Human skin model capable of natural shape variation,” *The Visual Computer*, no. 3, pp. 265-271, 1988.

17. Graham Walters, *Personal Communication*.

18. M. Nahas, H. Huitric, and M. Saintourens, “Animation of a B-Spline Figure,” *The Visual Computer*, vol. 3, no. 4, 1987.

19. Keith Waters, “A Muscle Model for Animating Three-Dimensional Facial Expression,” *Computer Graphics*, vol. 21, no. 4, July, 1987.

20. P. Bergeron, “Techniques for Animating Characters,” SIGGRAPH ’85 Tutorial Notes, ACM, 1985.

21. F. I. Parke, “Parameterized Models for Facial Animation,” *IEEE Computer Graphics and Applications*, vol. 2, no. 9, pp. 61-68, November 1982.

22. Keith Waters, “A Dynamic Model of Facial Tissue,” State of the Art in Facial Animation, SIGGRAPH’89 Tutorial Notes, 1989.

23. Graham Walters, “The Story of Waldo C. Graphic,” 3D Character Animation by Computer, SIGGRAPH’89 Tutorial Notes, 1989.

24. Barbara Robertson, “Mike, the Talking Head,” *Computer Graphics World*, July 1988.

25. Richard Bartels, John Beatty, and Brian Barsky, *An Introduction to Splines for Use in Computer Graphics and Geometric Modeling,* Morgan Kaufmann Publishers, Palo Alto, California, 1987.

26. David R. Forsey, “Motion Control and Surface Modeling of Articulated Figures in Computer Animation,” University of Waterloo Technical Report, CS-90-28 (PhD. Thesis), Waterloo, Ontario, 1990.

27. Elizabeth Cobb, ‘‘Design of Sculptured Surfaces Using the B-Spline Representation,’’ University of Utah PhD Thesis, Salt Lake City, Utah, 1984.

28. Alan Barr, ‘‘Global and Local Deformations of Solid Primitives,’’ *Computer Graphics*, vol. 21, no. 4, pp. 21-30, Minneapolis, Minnesota, July 23-27, 1984.

29. James Foley, Andries van Dam, Steven Feiner, and John Hughes, *Computer Graphics Principles and Practice,* Addison Wesley, 1990.

30. Richard H. Bartels and John C. Beatty, ‘‘A Technique for the Direct Manipulation of Spline Curves,’’ *Proceedings, Graphics Interface ’89*, CIPS, 1989.

31. Barry M. Fowler, ‘‘Geometric Techniques for Interactive Curve Design,’’ Master’s Thesis, Department of Computer Science, University of Waterloo, Waterloo Ontario, 1990.