Hybrid Anatomically Based Modeling of Animals
UCSC-CRL-98-05

Philip J. Schneider and Jane Wilhelms*
University of California, Santa Cruz

April 14, 1998

Abstract

We describe a new hybrid approach to modeling and animating animals. A pre–defined skin is modeled as a triangle mesh, such as may be purchased from a digital model vendor, or generated with a typical modeling program. This skin is then attached to the underlying bone, muscle, and tissue model. The rest shape of the skin is exactly as given in the surface model of the animal. Internal components – bones, muscles, and general tissue – are directly modeled with triangle meshes or ellipsoids. Changes in joint angle result in changes to the position of bones and generalized tissue, and in changes to the shapes of muscles. The attached skin vertices move with their underlying components, resulting in natural-looking deformations to the animal modeled. This alternative approach to modeling skin allows for greater realism or detail without requiring more accurate internals, and it allows pre–existing animal models to be realistically deformed using an anatomically based method.

---

* Computer Science Dept, Univ. of California, Santa Cruz, CA 95064, USA. E-mail: {`pjs, wilhelms`}`@cse.ucsc.edu`, Telephone: 408-459-2320, Fax: 408-459-4829

1 Introduction

Obtaining models of objects for computer animation has benefited substantially from the tremendous amount of research and development, both academic and industrial, in the general areas of digital modeling and computer-aided geometric design. A vast array of models of objects from chairs to cars to buildings are readily available, as are numerous sophisticated modeling software packages capable of creating complex models. As a result, one can with relative ease create complex scenes of arbitrary design.

Animation of such objects has received quite a lot of attention, as well. Many techniques have been developed, ranging from simple kinematics to sophisticated physics simulations. Very realistic (or intentionally unrealistic) motion of objects is possible, with a high degree of control. The results can be seen quite impressively in recent commercials and films, in which animated models are seamlessly integrated into live-action scenes.

However, the same cannot be said concerning humans and animals, at least not in degree. This may be partially attributed, particularly in the case of the modeling, to the "head start" inanimate objects have received; for example, the automobile and aerospace industries started working on modeling paradigms and techniques many decades ago. Humans and other animals have highly complex shapes, and are quite difficult to model, accounting for the remainder of the difficulty.

Motion control has been the subject of much research [27, 14, 18, 49, 17, 15, 33, 9, 3], but producing a desired motion is still quite difficult. If the motion is intended to be realistic (rather than cartoon-ish), then the problem is exacerbated further by the fact that humans are exceedingly sensitive to the most subtle errors in movement or surface deformation.

This state of affairs is most unfortunate, as humans and other animals are important, and becoming more so, in computer animation. In addition to uses in the entertainment industry, applications of animation of living beings are becoming quite important in the fields of medicine, surgical procedures, biomechanics, ergonomics, and the like.

Recent research in human modeling and animation has turned towards anatomically based methods, using underlying tissues to generate the surface shape [36, 37, 48, 50]. Another class of approaches have attempted to deform an a-priori-defined surface model of the being, using ad-hoc techniques to relate changes in joint angles segment positions to changes in the "skin" [26, 29, 16, 30]. We present a hybrid modeling and animation approach that combines the advantages of each of these previously unrelated paradigms. Like previous anatomically based approaches, the underlying model consists of individual muscles, bones, and generalized tissues, which mimic actual components of the animal body. Previous approaches automatically generated skin from these underlying components. This new method instead takes a pre–existing skin defined by a polygonal mesh and attaches it to the underlying components. In this way, we are able to take advantage of the strengths of both the anatomically based approach and the more traditional approaches.

Our modeling approach involves the following steps: (1) obtain a surface model of the desired animal, or create one using a modeling package such as *SGI Alias/Wavefront* [1], (2) triangulate the surface model, if not already triangles, (3) specify a body hierarchy and rest position that fits within the surface model, (2) design individual muscles, bones, and generalized tissues to approximately fit within the skin, and (3) attach skin surface vertices to the nearest underlying tissue.

Once the animal model has been defined and the skin attached, animation is accomplished as described in [48, 50]. Examples from our work on anatomically based modeling can be found on our web site: `www.cse.ucsc.edu/~wilhelms/fauna`.

2 Background and Related Work

Recent television commercials and feature films [10, 39] have featured animated humans and animals of impressive complexity. Through a highly laborious process, skilled modelers digitize physical models, which are then painstakingly animated by highly trained and talented animators; both human effort and CPU time are considerable.

From the early days of robotics and animation research, robots, humans, animals and other articulated bodies have been represented using a tree-structured hierarchy of rigid segments connected by flexible joints. These hierarchies are generally constructed in the shape of the underlying skeleton [4, 6, 31]. Historically, most representational and modeling schemes for humans and animals have been based on such hierarchies; beyond that, they diverge greatly in how (and if) the skin and internal components are modeled and deformed.

There are many ways of classifying research related to ours; one way is to distinguish between methods that attempt to deform a *pre–existing* skin surface model, and those that *generate* the skin surface. Our work brings together two previously disparate classes of methods for modeling animals for animation; accordingly, in the next sections we outline and give examples of each class.

2.1 Deforming a Pre-Existing Skin

Given the ubiquitous tree-structured hierarchy of rigid segments used to model the skeletons of animals, and an existing polygonal model of the human or animal's surface (sometimes the skin, but often skin, clothing and/or hair), the problem becomes how to modify the positions of the polygons' vertices (or smooth surface control points) in response to changes in the hierarchy. For example, if the angle between the segment representing the upper arm and that representing the lower arm changes (i.e., the elbow bends), how do we modify the nearby surface vertices so the "skin" deforms in the desired fashion?

It appears in the earliest days, polygon vertices were simply projected down onto the lines representing the segments, and were transformed along with the point of projection. Clearly, for any significant joint angle changes, the polygons around the joints would deform unpleasantly.

Magnanent-Thalmann *et al.* [29, 30], describe a method in which there is associated with each joint a specific transformation, which they call a *joint-dependent local deformation* or JLD. They utilize a conventional segment hierarchy surrounded by a polygonally modeled hand; vertices around joints are transformed by an ad-hoc method that linearly interpolates positions based on their proximity to the joint and the joint angle. Gourret *et al.* [20] used a enhanced this approach with a (skin surface) finite element method [42, 30, 12, 43] to model skin deformations of a hand during grasping.

Komatsu [26] presents a technique in which a hierarchy is covered with piecewise biquartic Bezier patches and Gregory patches. Control points surround the segments. In general, the control points are transformed along with their underlying segment, but the transformation is modified according to the angles of nearby joints; this keeps the surface smooth and non-interpenetrating around the joints, and simulates muscle bulging due to flexion.

Gascuel extracted a spline surface around a skeleton [16]. Sometimes the surface is geometrically adjusted during motion to mimic deformation [21]. Turner *et al.* used a deformable, elastic skin for character animation [45].

A variant of this class of methods have attempted to use a pre–defined skin (usually a polygonal mesh or spline surface of some type), which is embedded in some space–filling function whose purpose is to deform the skin surface in response to movement of the hierarchy. Moccozet and Magnenat-Thalmann [32] animated a polygonal hand placed over a hierarchy using Dirichelet free-form deformations to model the wrinkling of the palm and undersides of fingers due to joint flexion. Mark Henne's layered approach [23], used which *implicit fields* to simulate body tissue. Chadwick *et al.* layered approach [11] used *free-form deformations* [38]. Singh *et al.* also used implicit functions to simulate skin behavior [40].

2.2 Generating a Deformable Skin

Various researchers have utilized implicit functions to represent the volume of a body, where the skin is defined to be an isosurface. In some cases, transformations or modifications of the implicit functions are used to model deformations of the skin.

Badler *et al.* mimicked deformable material in early work by covering the body with many spheres [5], while Herbison-Evans used ellipsoids to represent each segment [24]. Blinn's seminal work on implicit surface modeling included a "blobby man" made by extracting a surface from around an articulated skeleton [7]. Bloomenthal [8] describes how a hand may be modelled using polygons and lines as primitives, which are convolved to form another type of implicit surface.

Generally, implicit surfaces are rendered either directly [7] or by grid–sampling (*voxelizing*) the implicit function and producing a polygonal surface from those samples (e.g., the “marching cubes” algorithm of [28]).

The work presented in this paper is directly related to that found in [48, 50], which shows that an anatomically based modeling approach can achieve a good balance between realism and the efficiency needed for interactivity. Underlying bones, muscles and generalized tissues are used to generate a skin surface, using the “marching cubes” algorithm. Anatomically based modeling is also described by Scheepers *et al.* [36, 37]. Their emphasis is on accurate modeling of bones and muscles, but do show preliminary results of generating a skin using the same technique. The research described here differs from and extends that work in that we show that an anatomically based approach for the internals of an animal may be used to deform a pre–existing skin.

Natural-appearing skin, fur, and hair are complex but important components of realistic models. Hanrahan and Krueger demonstrated realistic skin tones [22]. Noteworthy fur [25, 35, 46] and hair [2] have been produced. Reaction-diffusion approaches [44, 51] and “spot noise” [47] can simulate textures resembling animal fur. Like [48, 50], we can model hair or fur atop our pre–existing skin using the technique described in [46].

3 The Basic Underlying Model

Most of the basic model of the horse is created using the techniques described in [48, 50]; we describe it here briefly for context and completeness.

The horse model has 81 segments, including all segments connected by major moving joints in a vertebrate body: skull, jaw, vertebrae, pelvis, arms, legs, wrists, ankles, fingers, and toes. All joints are capable of three revolute degrees of freedom, but their range can be limited by a maximum and minimum angle.

The skeleton and generalized tissues are modeled as triangle meshes or ellipsoids. These components do not change shape during motion, but are each attached to a particular segment in the hierarchy, and thus move relative to one another.

The horse skeleton consists of 45 individual triangle-mesh bones based on a human skeleton model from *Viewpoint DataLabs* [13] and altered using *SGI Alias/Wavefront* software [1] to be more horse-like. There are 42 ellipsoidal bones for the tail, feet, ankles, lower front legs, rear legs, sacral vertebrae, skull, and jaw. The unadorned skeleton in rest state can be seen in Figure 1. Generalized tissue is represented by 17 ellipsoids, shown in purple in Figure 4.

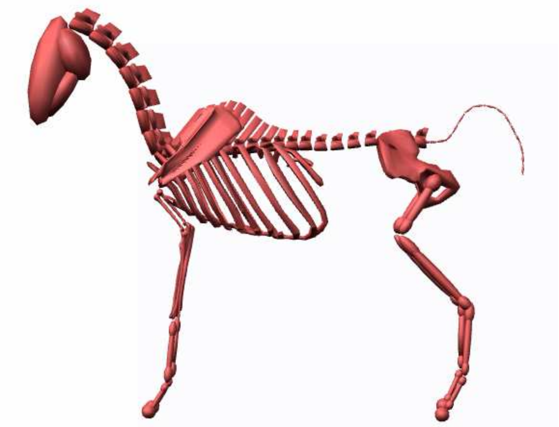

Figure 1: The foundation of the internal components - the skeleton. It contains both ellipsoidal and polygonal mesh bones. Notice the simplification and approximation relative to a true horse skeleton.

3.1 Muscles

Muscles are an elastic tissue, capable of contracting. Contraction of a muscle causes the bones to which they are attached to pull toward each other, causing joint motion (flexion or extension). Contracted and shortened muscles bulge, and relaxed and stretched muscles become thinner; overlying skin changes shape in response. In our modeled animal, the reverse happens – muscle shape changes *because of* joint motion, resulting in realistic skin deformations during animation.

Our muscle model is that introduced in [50]. Briefly, muscles are positioned on the bones using two *origins* and two *insertions* on parametric locations on the bones. Between these sets of origins and insertions, a default discretized, deformable cylinder is generated automatically by the system (Figure 2). There are no explicit tendons.

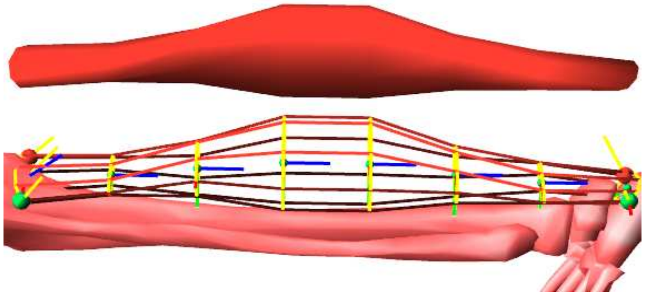

Figure 2: Typical default *deformed-cylinder* muscle. The muscle is defined by two *origins* (red and green spheres at left) and two *insertions* (same at right). The wireframe view shows eight yellow cross-sectional slices, connected by red edges to form a polygon mesh. The short blue and green lines are slice coordinate frame $Z-$ and $Y-$axes. The shaded polygon mesh is shown above.

The muscle cylinder's (Z) axis is a curve that runs from a point midway between the two origins to point midway between the two insertions. Generally, the cylinder is discretized into 7 longitudinal *muscle sections* demarcated by 8 elliptical cross-sectional *slices*, as shown in Figures 2 and 6. The result is a polygonal mesh model of the muscle. A parametric trilinear function (a *tri–affine function*) is defined by adjacent slices, and is used to locate and move skin vertices. Tri–affine functions are defined on adjacent cubes so that they map each shared corner of their cubes to the same point, ensuring that they combine to make a $C0$ continuous transformation. Related ideas are found in free-form deformations of computer modeling [38].

Default rules modify the X and Y dimensions of the muscle on successive slices from origin to insertion; the vertex locations for the intermediate slices are scaled in X and Y to produce a fusiform shape, larger in the middle slices than in the end slices, and larger across (in X) than in thickness (in Y) The polyhedral vertices of the muscle's surface lie in the XY plane of each slice, arranged symmetrically around the slice coordinate frame origin, discretizing the ellipse. The number of vertices in each slice is under user control. Figure 2 connects the muscle vertices within each slice with yellow lines, and muscle vertices between slices by reddish lines.

The user can interactively alter the size and cross–sectional aspect ratio of a muscle, the orientation and location of slice coordinate frames, and the locations of origins and insertions. Figure 3 shows the non-default muscles shapes for the front leg, which illustrate the topics of this section.

Muscles are not directly animated; rather, a joint angle change in the hierarchy cause one or more bones associated with the affected segments to move. When this happens, the relative positions of the origins and insertions of muscles on these bones change, and the system automatically recalculates the shape based on these new positions. A new default shape is found, and automatically adjusted using the user-specified non-default muscle parameters described in the previous section.

Muscle width and thickness are then scaled to maintain approximately constant muscle volume as the joint angle changes. Volume is preserved exactly in regions between parallel slices, and is changed as a second order effect in regions between two nonparallel scaled slices. Regions involving end slices vary in volume, as end slices (which are connected to the bones) do not change shape. Exact muscle

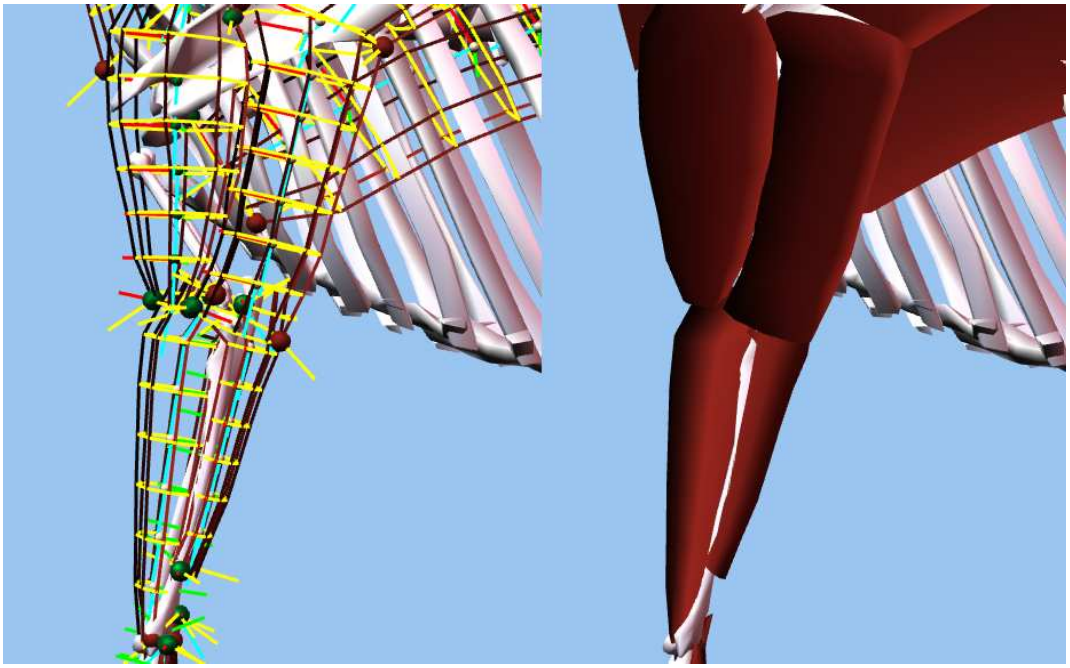

Figure 3: The front left leg muscles of the horse, shown in outline at left, with origins and insertions as spheres, and shaded at right. These muscles are all non-default shapes.

volume preservation, however, is not necessary for realistic deformation, and exact volume preservation of muscles is not biologically justified (consider isometric contraction). This process is repeated with every change in joint angle.

Figure 4 shows all of the muscles used in the horse model. The muscles are extremely simplified relative to a real horse.

4 Skin

In our system, the skin is a pre–existing triangle mesh, obtained from *Viewpoint DataLabs* [13], as shown in Figure 5. There are several things to note: the relative coarseness of the model – there are 2,027 vertices and 4,050 polygons, and that the horse is somewhat stylized, but fairly realistic.

The skin is an elastic triangle–mesh surface that is attached to underlying components but can move relative to them. The novel contribution of this paper is the demonstration that the methodology for skin deformation in response to deformation of the underlying tissue, introduced and used in [48, 50], can be applied to pre–existing skin, allowing anatomically realistic deformations to be applied to standard boundary–rep digital animal models.

4.1 Anchoring Skin

Once the bones, muscles, and stuffing tissue have been defined, in a second stage called *anchoring*, each vertex in the pre–existing triangle-mesh skin is associated with the closest underlying body component.

The *anchor* of a particular skin vertex is the nearest point on its underlying component. More important for animation is the *virtual anchor*, which is the initial position of a skin vertex relative to its underlying component. The anchors and virtual anchors are stored parameterized in the local space of the component. If shape changes occur in the underlying component, they are transmitted through the anchors and virtual anchors, to affect the skin vertices correspondingly. Each skin vertex is considered

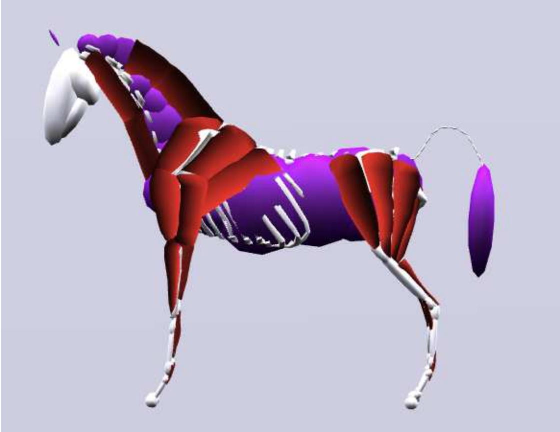

Figure 4: The bones, muscles, and stuffing. Notice the simplification of the musculature.

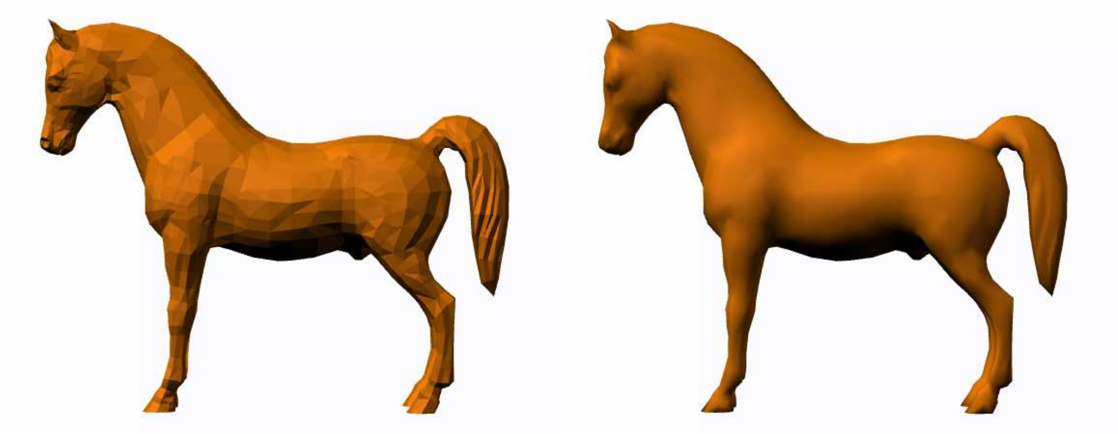

Figure 5: The original polygonal mesh skin model from Viewpoint. The flat-shaded image on the left shows the relative coarseness of the model.

to be connected to its virtual anchor by a spring of rest length zero, and a specified spring stiffness. (See Section 4.2.)

In the *anchoring* process, we find the nearest underlying component of each skin vertex, converting that skin vertex to a parameterized local location relative to the component, and storing this local position of the skin vertex as its virtual anchor.

We have three distinct types of geometric primitives to which we may attach skin vertices: ellipsoids representing the stuffing and some of the bones, polygonal meshes representing the more complex bones, and the discretized cylinders representing the muscles.

In the first case, the problem is to find a point on the surface of the ellipsoid, nearest the skin point. As the solution of the equation for an ellipsoid with a given point only gives the distance from the, a Newton–Raphson method [48] is used, iterating until a user–specified tolerance or iteration count limit is hit; the location at termination of the iteration becomes a candidate anchor point.

Anchoring skin to triangle–mesh bones is a straightforward process of transforming skin vertices into the coordinate system of the bone and scaling by the size of the bone's bounding box in each dimension. The resulting point on the bone becomes a candidate anchor.

Anchoring points to deformed–cylinder muscles is a more difficult and interesting problem. One cannot simply apply a method like that used for the bones and map the skin vertex into the frame of a singe slice, because one would get abrupt changes in skin shape as the muscle changed shape (due to the change in mapping from one slice frame to the next). The tri–affine transformation mentioned earlier is defined the space between the planes of the two slices in rest position; this transformation is inverted, and the skin point is then transformed with it to get a parametric representation in the coordinate frame of the segment between two slices. This parameterized position becomes the (candidate) virtual anchor for the skin point. In Figure 6, the lighter skin vertex lies nearest to the muscle segment between slices 4 and 5, and will be mapped to a parametric location in this region. Details of this process can be found in [50].

A skin vertex may be near several underlying components of different types. By computing, as just described, the nearest point on each underlying component, we simply choose the closest one and use that as the anchor. This expensive process is done just once, at the initial anchoring time.

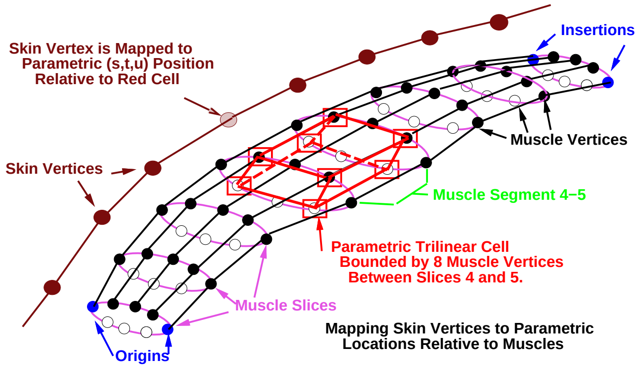

Figure 6: Illustration of mapping skin vertices to parametric trilinear functions over muscle segments. The lightest skin vertex lies between muscle slices 4 and 5, and is mapped into an (s, t, u) coordinate system defined by the eight muscle vertices shown in red.

4.2 Deforming the skin

When joints in the hierarchy move, the skin, muscles, and stuffing move (with the muscles automatically re–shaping as described earlier). In order to model the deformation of the skin due to this motion, the skin vertices must be transformed as well.

Each skin vertex is connected to its virtual anchor by a spring (with zero rest length and a user–modifiable default stiffness), and to each of its neighboring vertices as well (with rest length equal to the initial distance and a user–modifiable default stiffness). These are used, as described below, to give the animal an elastic skin.

For the ellipsoidal stuffing and polygonal–mesh bones, the (initial) positions of the skin vertices are found by transforming the previous position (in the local coordinate frame of the underlying closest component) to world space, using the new relationship between local and world space. The process is a bit more complex for the muscles: when the body is moved, new world space positions are calculated for the slices (see Section 3.1). Then for each adjacent pair of slices a *new* parametric trilinear transformation is defined. Virtual anchor points associated with this segment are mapped from parametric to world space, using the new tri–affine transformation, in the *forward* direction. Each virtual anchor provides an initial skin position for its corresponding skin vertex, for this body configuration.

Together with other forces and constraints in the system, these springs are brought into equilibrium by means of a series of relaxation operations. The initial skin positions, from which relaxation commences, are provided by the positions of the virtual anchor, as described in Section 4.1. Relaxation operations continue iteratively until a user-defined convergence tolerance is reached, or a user-defined maximum number of iterations has occurred. Figure 7 shows the effect of this relaxation process. The smooth redistribution seen in the right image is important to achieve a natural appearance for fur [46] or skin with markings.

Figure 7: This figure illustrates the concepts of anchors, virtual anchors, and elastic relaxations where the skin is anchored on two neighboring segments. Skin vertices are connected by brown edges to form a triangle mesh. In the left image, skin vertices coincide with their virtual anchors, as no elastic relaxation has been done. In the right image, after elastic relaxation, the virtual anchor positions are unchanged, while the skin vertices have been redistributed more uniformly. In both images yellow lines connect skin vertices to their muscle anchors. In the right image red lines connect the skin vertices to their virtual anchors, showing the displacement necessary to equalize spring forces.

User–controllable parameters can be applied to the skin to make it *pull* from a stretched position toward its rest state more strongly than it *pushes* back when it is compressed, and to make the skin appear more smooth by adjusting the model so that the skin is slightly stretched in its extracted configuration.

Figure 8: All components of the horse, shown with a wireframe rendering of the original skin model. Again, note the rather coarse nature of the skin model.

5 How to Build a Horse

Given a pre–existing skin model (such as the horse we used for this paper) and the software for designing hierarchies, bones, muscles, and stuffing, how does one end up with a complete, animatable horse? Here's what we did:

1. Acquired a polygonal model of the desired animal. These can be purchased from digital model vendors such as *Viewpoint DataLabs* [13] (as we did), or created using modeling software such as *SGI Alias/Wavefront* [1].
2. Examined the skin, and decided on a tree structure and number of segments. We modified the monkey hierarchy used in [48], in particular removing the extraneous fingers and toes.

3. Fit the hierarchy within the skin, taking into account the location of the joints. We created images of the original skin model, and used them as background texture maps in the design software; the images were then used as templates to define the initial layout of the hierarchy (much as billboard painters project a slide of the desired image on the billboard in order to sketch in the outlines), helped along by consulting books displaying horse skeletons [19, 41].
4. Modified the design software to read in the horse skin. Read in the skin, triangulating all polygons with more than three vertices.
5. Transformed the hierarchy to approximately place the now–defined skeleton inside the skin, taking advantage of our ability to display the skin translucently.
6. Added bones, again consulting [19, 41]. Some bones were taken from the monkey model used in [48, 50], which had been modified from polygonal bone models from *Viewpoint DataLabs* [13]; others were ellipsoidal bones, sized to fit the location in question.
7. Interactively added stuffing ellipsoids to the model.
8. Added relatively few default muscles where it seemed muscle-defined motion was essential, and interactively shaped them to fit the skin.
9. Fine–tuned the dimensions of the hierarchy, bones, stuffing, and muscles to more closely fit inside the skin.
10. Attached the skin as described earlier and in [48, 50].

At this point, we had a complete horse model with pre–defined skin attached, which would now transform according to changes in the underlying bones, muscles, and stuffing. This sequence actually contains a loop: After attaching the skin and animating the horse, we would detect problem areas, and return to steps 5 to 9 as needed. For example, we found small muscles in the lower legs to be essential to maintain shape during motion.

6 Results and Discussion

In creating a model for animation, the user would be working from an image (on paper or only in his mind's eye) or a sculpture, or the like; in any case, the most important thing, in terms of the ultimate use, is that the outward shape of the skin conform to the designer's idea of what it should look like. The "internals" are likely to be of no direct interest at all to the animator.

Approaches that attempt to generate skin atop a hierarchy (e.g., [26]) tend to produce excessively smooth, unrealistic skins; such an approach may be quite useful, however, for some types of animals (e.g., a snake or starfish).

Approaches using blended implicit primitives (e.g., [7]) also tend to produce excessively smooth surfaces, and it appears that the generation of any sort of realistic surface would be exceedingly difficult. Convolution surfaces (e.g., [8]) have done a passable job at modeling, for example, a hand, but again it would likely be quite difficult to effectively model an entire animal.

While these various approaches may have some elegance and/or technically interesting aspects, they are in a sense "backward" from how an artist, modeler, animator, or technical director likely approaches the whole problem. One would have to attempt to "reverse engineer" the desired skin shape by either crafting the underlying primitives (e.g., density fields, ellipsoids, etc.) properly; this "backward" approach would seem awkward at best, and it appears that it would be nearly impossible to use any of these techniques to create an arbitrary skin surface shape.

Attempts to model the deformation of skin by applying various techniques to transform a surface-only model based on joint angles and segment positions have met with limited success. Methods such as [29, 30] do work to some extent, but are ad-hoc, must be tuned for the joints, most likely fail for any sort of twisting, and may be difficult if not impossible for complex joints such as hips or shoulders.

The methods in which an a-priori surface is embedded in a function (e.g., [32, 23, 11, 40]) seem to be a step in the right direction, in that they also use a pre-formed resting skin surface representation. While gross deformations such as muscle bulging due to flexion have been demonstrated, these approaches are again ad-hoc, and it seems that it would be exceedingly difficult to craft or modify functions that would produce realistic deformations in any case.

Given these shortcomings, and considering the tremendous increases in CPU speed and availability of memory and disk space, researchers have turned to anatomically based approaches, with some success [48, 50, 36, 37]. Their approaches create reasonably realistic skin surfaces, which behave in a relatively predictable and realistic fashion when the joints are moved. However, if the goal of modeling is to create a model whose shape conforms to some a-priori criteria (an artist's conception, a clay figure, an animal digital model, etc.), then the user has the considerable task of creating bones and tissues (muscle, organs, etc.) whose effect together gives the skin the required shape.

This "shortcoming" is not so daunting as in the other approaches. If one is attempting to create a life-like model of a specific type of animal, it may be possible to obtain digitally scanned (e.g., CyberWare) models of the bones to use in a skeleton, and to use CAT or MRI scans to help model the muscles and other tissue. This would, one expects, be a very time-consuming task, but possible. On the other hand, at the present time there is little in the way of digital bone models, and almost no data for other tissues, so very few animals could be modeled this way.

Even given that one could get the skin's shape anatomically accurate by careful attention to detail, the problem still exists that an animator may be requiring a *particular* animal (not just a generic example of the species), or a more stylized version than real-life. Further, if the animal to be modeled is merely a figment of someone's imagination, then anatomically accurate models of bones and muscles simply fail to exist.

The technique described in this paper attempts to remove these shortcomings. By using a pre-defined polygonal skin, we can create an animatable model for arbitrary skin shape, which deforms with good predictability and anatomical correctness.

For creating a model with anatomically believable skin and skin deformations, this approach doesn't require that the bones, muscles, and other tissue be modeled with a great deal of fidelity (as, for example, Sheepers [36, 37] would). Indeed, consider the high fidelity to natural horse motion as seen in Figure 9 and Figure 10, in spite of the very approximate skeleton (Figure 1) on which are placed some fairly quick-and-dirty muscles and stuffing (Figure 4). Figure 10 shows the horse rendered with a fur-generating algorithm [46].

On the other hand, the method described here *requires* a pre–defined skin, while other anatomically based approaches automatically generate the skin. Creating a complex and detailed polygonal skin for our approach is a non–trivial task, but so is the creation of bones, muscles, and tissues that generate an animal with the desired shape. One defining difference, then, between these methods is where they put the most difficult work; current state–of–the–art in modeling software gives the creation of polygonal skin a slight edge.

As noted earlier, the polygonal skin is quite coarse, a characteristic that tends to work quite strongly against high fidelity of deformations. Consider that the monkey modeled in [50] had a generated skin with approximately 150,000 triangles, while the horse skin used in this work had 4,050 triangles. Our experience suggests that a more finely-tesselated skin would show even more impressive results.

Further evidence of the efficacy of our approach can be seen in the accompanying video, which shows the horse model in action. The trot was (very inexpertly) animated by the first author, using as keyframes the images from Eadweard Muybridge's [34] animal motion sequence "Daisy with Rider". Some frames from the video are shown in Figure 11.

7 Future Work

Creating the joint-and-segment hierarchy and creating and modifying the muscles to fit inside the pre-existing skin is entirely manual, and is quite laborious. Several modifications could improve this process significantly:

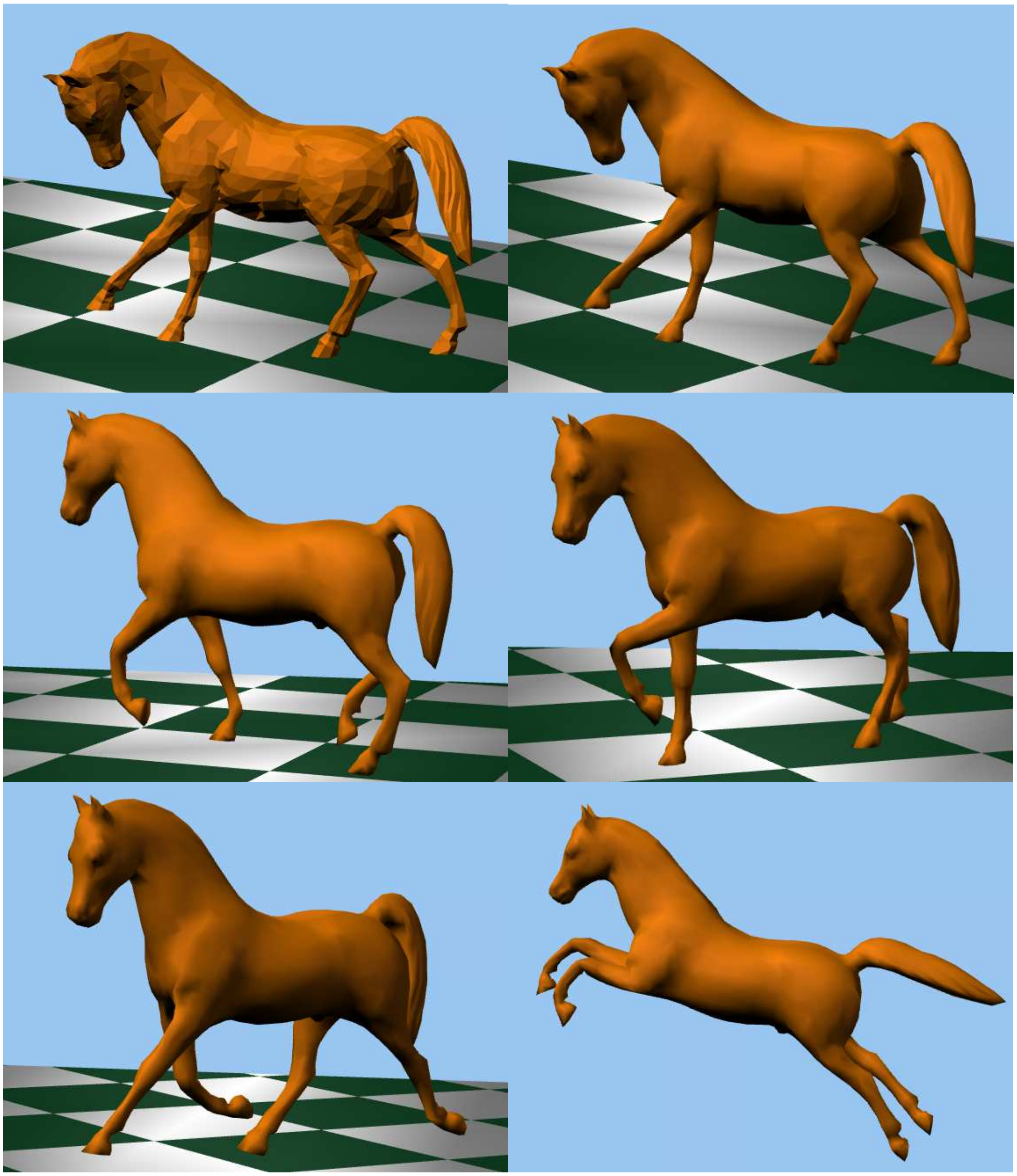

Figure 9: A selection of horse images. Notice the effect of individual muscles, and the ability of the model to simulate both stretching and folds. The flat-shaded image shows more clearly the natural changes in the shape of the skin.

- Introduce simple inverse kinematics to allow the user to pull on a joint or segment, and have a constrained portion of the hierarchy above it move along with it. For example, one could adjust the position of the leg sub-hierarchy by repositioning the foot.
- Enhance the process of modifying an existing hierarchy by automatizing the changes. For example, one could modify an existing spine by allowing the user to interactively specify the positions of, say, the first lumbar and first cervical vertebrae, and having the system apply linear interpolation

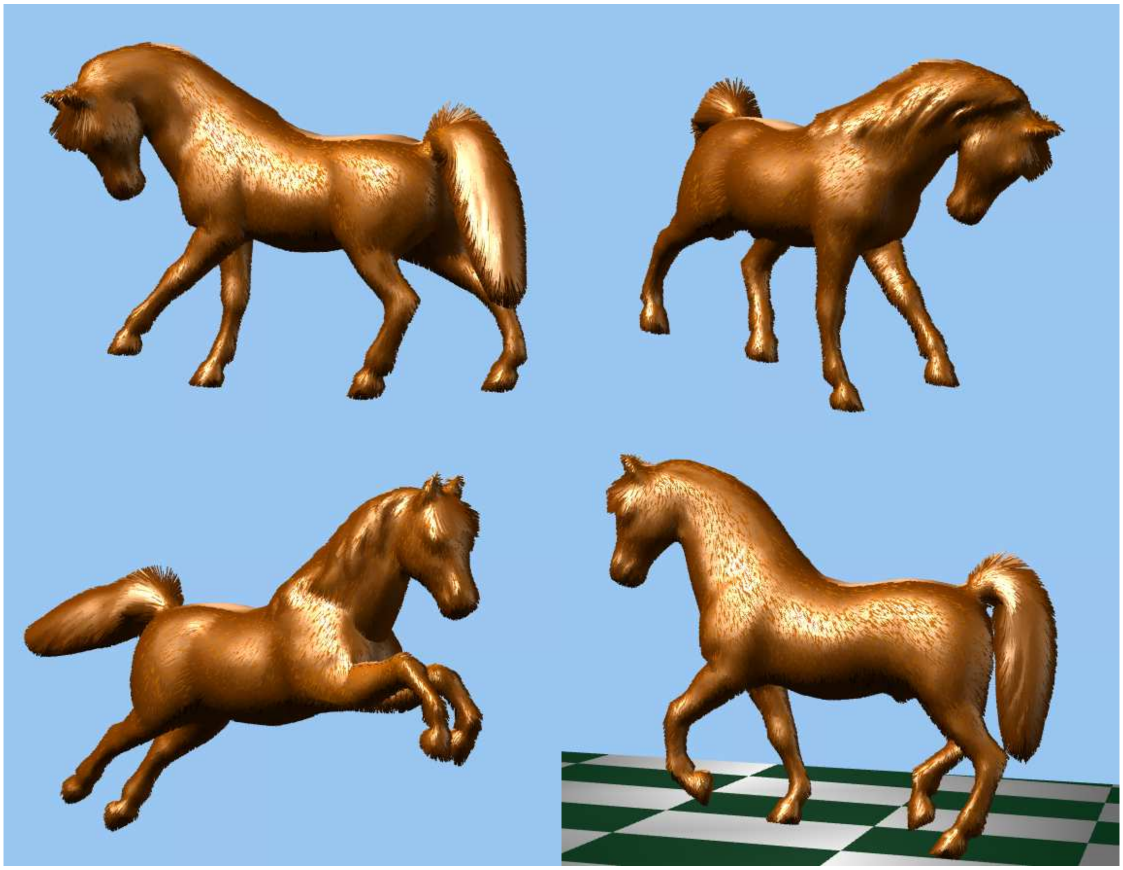

Figure 10: More horse images, showing again the realistic shape deformations due to joint angle changes, and showing the fur-rendering algorithm used.

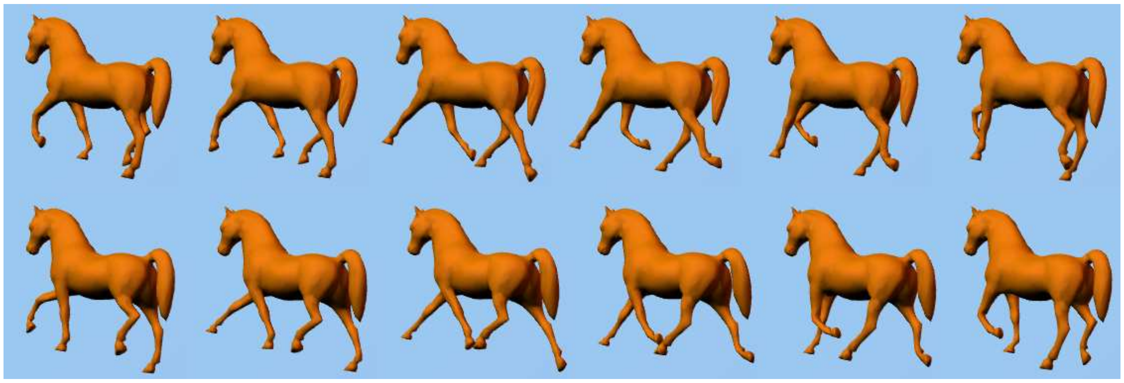

Figure 11: Some frames from the animation of our horse trotting.

to automatically position and scale the vertebrae in between.

- For best effect, muscles and stuffing should be positioned relatively close to the skin, and have shapes whose outward surface “follows” the shape of the skin it underlies. This process is entirely manual, and exceedingly tedious. It should be possible to have the system automatically “inflate” a muscle in the direction of the overlying skin.

- For areas of skin that overlie bone directly, or have only small or thin muscles intervening, it may be possible to have the system automatically (or semi-automatically) modify the shape or position of the underlying bone to better match the skin's shape.

8 Conclusions

This paper describes a hybrid modeling and animation approach for humans and other animals, utilizing an anatomically based representation of internal components, but which allows the use of separately-created surface models of the skin. The result is that we can relatively efficiently and consistently, with little artistic talent, produce believable, anatomically reasonable, and pleasing whole–body deformations of a polygonally modeled animal or human.

We believe this is approach benefits from the advantages of both anatomically based techniques and more traditional ad–hoc surface–deformation techniques, and avoids significant difficulties and shortcomings of these two previously disparate methods.

Acknowledgments

List processing software by Yumi Tsuji was used in this software package. Marlon Veal helped with programming. Allen Van Gelder helped with the animation. Research supported by a gift from Research and Development Laboratories, and by NSF Grant CDA-9115268.

References

[1] Alias/Wavefront. Silicon Graphics Inc. Mountain View, Ca.

[2] Ken-ichi Anjyo, Yoshiakai Usami, and Tsuneya Kurihara. A simple method for extracting the natural beauty of hair. *Computer Graphics (ACM SIGGRAPH Proceedings)*, 26(2):111–120, July 1992.

[3] Kiyoshi Arai. *Models and Techniques in Computer Animation*, chapter Keyframe Animation of Articulated Figures Using Partial Dynamics, pages 243–257. Springer-Verlag, Tokyo, 1993.

[4] Norman Badler and David Zeltzer, editors. *Making Them Move*. Morgan Kaufmann Publishers, Inc., San Mateo, CA., 1991.

[5] Norman I. Badler, Joseph O'Rourke, and Hasida Toltzis. A spherical representation of a human body for visualizing movement. *Proceedings of the IEEE*, 67(10):1397–1403, October 1979.

[6] Norman I. Badler, Cary B. Phillips, and Bonnie Lynn Webber. *Simulating humans : computer graphics animation and control*. Oxford University Press, New York, 1993.

[7] James F. Blinn. A Generalization of Algebraic Surface Drawing. *ACM Transactions on Graphics*, 1(3):235–256, July 1982.

[8] Jules Bloomenthal. *Hand Crafted*. Association for Computing Machinery, July 1993. SIGGRAPH '93 Course 25 Notes: Modeling, Visualizing, and Animating Implicit Surfaces.

[9] Ronan Boulic and Daniel Thalmann. Combined direct and inverse kinematic control for articulated figure motion editing. *Computer Graphics Forum*, 21:189–202, 1992.

[10] Phil Carpenter. Commercial spot cola bears. *Cinefex Magazine*, December 1994.

[11] John E. Chadwick, David R. Haumann, and Richard E. Parent. Layered Construction for Deformable Animated Characters. In *Computer Graphics (SIGGRAPH 89 Conference Proceedings)*, volume 23 of *Annual Conference Series*, pages 242–252. Addison Wesley, August 1989.

[12] David T. Chen and David Zeltzer. Pump It Up: Computer Animation Based Model of Muscle Using the Finite Element Method. In *Computer Graphics (SIGGRAPH 92 Conference Proceedings)*, volume 26, pages 89–98. Addison Wesley, July 1992.

[13] Viewpoint DataLabs. Orem, Utah.

[14] R. Featherstone. The calculation of robot dynamics using articulated-body inertias. *International Journal of Robotics Research*, 2(1):13–30, Spring, 1983.

[15] David Forsey and Jane Wilhelms. Manikin: Dynamic analysis for articulated body manipulation. *Graphics Interface '88*, June, 1988. also UCSC Tech. Report UCSC-CRL-87-2 to appear.

[16] Marie-Paule Gascuel. Welding and pinching spline surfaces: New methods for interactive creation of complex objects and automatic fleshing of skeletons. In *Graphics Interface '89 Proceedings*, pages 20–27, June 1989.

[17] Michael Girard. Interactive design of 3d computer-animated legged animal motion. *IEEE Computer Graphics and Applications*, 7(6):39–51, June 1987.

[18] Michael Girard and Antony A. Maciejewski. Computational modeling for the computer animation of legged figures. *SIGGRAPH '85 Conference Proceedings*, 19:263–270, July, 1985.

[19] Peter C. Goody. *Horse Anatomy: A Pictorial Approach to Equine Structure*. J. A. Allen, London, 1976.

[20] Jean-Paul Gourret, Nadia Magnenat-Thalmann, and Daniel Thalmann. Simulation of Object and Human Skin Deformations in a Grasping Task. In *Computer Graphics (SIGGRAPH 89 Conference Proceedings)*, volume 23, pages 21–30. Addison Wesley, July 1989.

[21] M. Hahas, H. Huitric, and M. Saintourens. Animation of a b-spline figure. *The Visual Computer*, 3(4), 1987.

[22] Pat Hanrahan and Wolfgang Krueger. Reflection from layered surfaces due to subsurface scattering. *Computer Graphics (ACM SIGGRAPH Proceedings)*, 27:165–174, August 1993.

[23] Mark Henne. A Constraint-Based Skin Model for Human Figure Animation. Master's thesis, University of California, Santa Cruz, Santa Cruz, CA 95064, June 1990.

[24] Don Herbison-Evans. Nudes 2: A numeric utility displaying ellipsoid solids. *SIGGRAPH '78 Conference Proceedings*, pages 354–356, August, 1978.

[25] James T. Kajiya. Rendering fur with three dimensional textures. *Computer Graphics (ACM SIGGRAPH Proceedings)*, 23(3):271–280, July 1989.

[26] K. Komatsu. Human Skin Model Capable of Natural Shape Variation. *The Visual Computer*, 4(3):265–271, 1988.

[27] James U. Korein and Norman I. Badler. Techniques for generating the goal-directed motion of articulated structures. *IEEE Computer Graphics and Applications*, 2(9):71–81, November, 1982.

[28] William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. *Computer Graphics (ACM Siggraph Proceedings)*, 21(4):163–169, July 1987.

[29] N. Magnenat-Thalmann, R. Laperriere, and Da. Thalmann. Joint-dependent local deformations for hand animation and object graphics. In *Proceedings of Graphics Interface '88*, pages 26–33, 1988.

[30] Nadia Magnenat-Thalmann and Daniel Thalmann. Human Body Deformations Using Joint-Dependent Local Operators and Finite Element Theory. In N. Badler, B. Barsky, and D. Zeltzer, editors, *Making Them Move*, pages 243–262. Morgan Kaufmann Publishers, Inc., San Mateo, CA, 1991.

[31] Nadia Magnenat-Thalmann and Daniel Thalmann, editors. *Models and Techniques in Computer Animation*. Springer-Verlag, New York, 1993.

[32] Laurent Moccozet and Nadia Magnenat Thalmann. Dirichelet free-form deformations and their application to hand simulation. In *Proceedings of Computer Animation '97*, pages 93–102. IEEE, 1997.

[33] Matthew Moore and Jane Wilhelms. Collision detection and response for computer animation. *SIGGRAPH '88 Conference Proceedings*, 22(4):289–298, August, 1988.

[34] Eadweard Muybridge. *Animals in Motion*. Dover Publications, Inc., 1957. Excerpted from Muybridge's 1887 *Animal Locomotion*.

[35] Rhythm and Hues. Aurora borealis, skate, and christmas twins, 1993-1994. (Coca Cola Polar Bears).

[36] Coenraad F. Scheepers. *Anatomy-based Surface Generation for Articulated Models of Human Figures*. PhD thesis, The Ohio State University, 1996.

[37] Ferdi Scheepers, Richard E. Parent, Wayne E. Carlson, and Stephen F. May. Anatomy-based modeling of the human musculature. In *SIGGRAPH 97 Conference Proceedings*, Annual Conference Series. ACM SIGGRAPH, Addison Wesley, August 1997.

[38] Thomas W. Sederberg and Scott R. Parry. Free-form deformations of solid geometric objects. In *Computer Graphics (SIGGRAPH 92 Conference Proceedings)*, volume 20, pages 151–160. Addison Wesley, August 1986.

[39] Don Shay and Jody Duncan. *The Making of Jurassic Park*. Ballantine Books, New York, 1993.

[40] Karansher Singh, Jun Ohya, and Richard Parent. Human figure synthesis and animation for virtual space teleconferencing. In *Proceedings of the Virtual Reality Annual International Symposium '95*, pages 118–126, Research Triangle Park, N.C., March 1995. IEEE Computer Society Press.

[41] George Stubbs. *The Anatomy of the Horse*. Dover Publications, Inc., New York, 1766. Updated Dover edition, 1976.

[42] Demetri Terzopoulos and Kurt Fleischer. Modeling inelastic deformation: Viscoelasticity, plasticity, fracture. *SIGGRAPH '88 Conference Proceedings*, 22(4):269–278, August, 1988.

[43] Xiaoyuan Tu and Demetri Terzopoulos. Artificial fishes: Physics, locomotion, perception, behavior. *Computer Graphics (ACM Siggraph Proceedings)*, pages 43–50, July 1994.

[44] Greg Turk. Generating textures on arbitrary surfaces using reaction-diffusion. *Computer Graphics (ACM SIGGRAPH Proceedings)*, 25(4):289–298, July 1991.

[45] R. Turner and D. Thalmann. The Elastic Surface Layer Model for Animated Character Construction. In N. M. Thalmann and D. Thalmann, editors, *Proceedings of Computer Graphics International '93*, pages 399–412, Lausanne, Switzerland, June 1993. Springer-Verlag.

[46] Allen Van Gelder and Jane Wilhelms. An Interactive Fur Modeling Technique. In *Proceedings of Graphics Interface*, May 1997.

[47] Jarke J. van Wijk. Spot noise—texture synthesis for data visualization. *Computer Graphics (ACM Siggraph Proceedings)*, 25(4):309–318, 1991.

[48] Jane Wilhelms. Animals with Anatomy. *IEEE Computer Graphics and Applications*, 17(3):22–30, May 1997.

[49] Jane Wilhelms and Brian A. Barsky. Using dynamic analysis for the animation of articulated bodies such as humans and robots. *Proceedings of Graphics Interface '85*, pages 97–104, May 1985.

[50] Jane Wilhelms and Allen Van Gelder. Anatomically based modeling. In *Computer Graphics*, pages 173–180, Los Angeles, Ca., August 1997. ACM Siggraph Conference Proceedings.

[51] Andrew Witkin and Michael Kass. Reaction-diffusion textures. *Computer Graphics (ACM SIGGRAPH Proceedings)*, 25(4):299–308, July 1991.