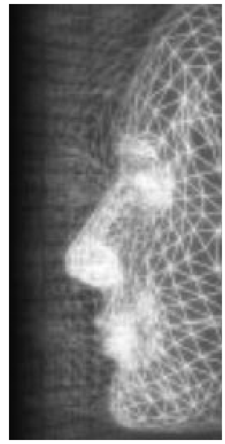
COMPUTER ANIMATION AND VIRTUAL WORLDS
Comp. Anim. Virtual Worlds 2006; **17**: 281–292
Published online in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/cav.132

Curve skeleton skinning for human and creature characters



*By Xiaosong Yang, Arun Somasekharan and Jian J. Zhang**

The skeleton driven skinning technique is still the most popular method for animating deformable human and creature characters. Albeit an industry de facto due to its computational performance and intuitiveness, it suffers from problems like collapsing elbow and candy wrapper joint. To remedy these problems, one needs to formulate the non-linear relationship between the skeleton and the skin shape of a character properly, which however proves mathematically very challenging. Placing additional joints where the skin bends increases the sampling rate and is an ad hoc way of approximating this non-linear relationship. In this paper, we propose a method that is able to accommodate the inherent non-linear relationships between the movement of the skeleton and the skin shape. We use the so-called curve skeletons along with the joint-based skeletons to animate the skin shape. Since the deformation follows the tangent of the curve skeleton and also due to higher sampling rates received from the curve points, collapsing skin and other undesirable skin deformation problems are avoided. The curve skeleton retains the advantages of the current skeleton driven skinning. It is easy to use and allows full control over the animation process. As a further enhancement, it is also fairly simple to build realistic muscle and fat bulge effect. A practical implementation in the form of a Maya plug-in is created to demonstrate the viability of the technique. Copyright © 2006 John Wiley & Sons, Ltd.

Received: 10 April 2006; Revised: 2 May 2006; Accepted: 10 May 2006

KEY WORDS: skeleton; curve-skeleton; deformation

Introduction

A realistic and visually accurate character animation necessitates proper skin deformation of the character models. Skin deformation owes a large part to proper rigging of the characters. The virtual skeleton forms the interface by which the animator can pose or animate the characters. The joint-based skeleton has been very popular in the animation industry for many years and has nearly become a *de facto* standard. Other technologies like inverse kinematics, forward kinematics, motion capture etc. are built on this hierarchical system of joints. It is plain to see why the joint-based skeleton system is thoroughly integrated into the current production pipeline in animation. Where visual fidelity is of the utmost importance, with respect to film quality animation, a combination of techniques including muscle simulation is used to achieve the realistic best in mesh deformation. The attachment of mesh geometry to the underlying skeleton rig is called 'skinning' and this can be understood as a function mapping of the skeleton parameters to a deformation field.[1] One of the common skinning methods in interactive systems is known by the following nomenclatures: *sub-space deformation* (SSD), *smooth skinning*, *linear blend skinning* and *enveloping*. The process followed by this technique is to assign influence joints and blend weights to each vertex of the character. Transforming the vertex by a weighted combination of the joints local coordinate frames completes skin computation. But in spite of computational performance and ease of use, the joint skeleton skinning is not without its share of problems, particularly where skin deformation is concerned. Unusual deformation artifacts appear in the skin while deforming. Some of the commonly seen problems in joint-based skinning during deformation are: *candy*

*Correspondence to: Jian J. Zhang, National Centre for Computer Animation, Bournemouth Media School, Bournemouth University, Poole, Dorset, BH12 5BB, UK.
E-mail: jzhang@bournemouth.ac.uk

Contract/grant sponsor: British Arts and Humanities Research Council; contract/grant number: B/RG/AN5263/APN12727.

wrapper effect during twist deformation and *collapsing joints*, which would create a rubber-tube like effect. There are certain solutions to circumvent these problems (which we will examine later in the paper) but they have their own drawbacks. Nevertheless, the joint-based system is popular owing to its interactivity and use of minimal animation data. More importantly, it is almost an integral part of the current animation workflow and animators are reluctant to abandon their familiar production practice.

The relationship between a skeleton and the skin shape is highly non-linear. The problems of joint-based skeleton skinning mentioned above, in essence arises from under-sampling. The transformations of the two related joints are too far from each other. And with that low-rate sampling they fail to give a good approximation of the deformed skin surface.

In this paper, we introduce a novel method called *curve skeleton skinning*, to overcome the persisting drawbacks of joint skeleton skinning. The basic idea is to represent the relationship between the skeletal movements and the skin deformation in a non-linear continuous fashion. Since a lot of contemporary animation technology is built upon the hierarchical joint-skeleton based system, it is not wise to entirely replace the current practice. What we propose to do is to enhance the current joint skinning system using the curve skeleton skinning and retain the current animation production pattern that the animators are familiar with. While the joint skeleton is a discrete centre line representation of an object, the curve skeleton offers a continuous skeletal representation. Thus a character will have two skeletons: the ordinary joint skeleton and a curve skeleton. The curve skeleton being continuous gives the maximum sampling rate and provides skin deformation transformation without any artifacts. In addition, we will demonstrate how the curve skeleton technique can drive muscle-based systems to achieve realistic muscle deformation during animation. Using our technique, the animator is able to work without digressing from the familiarity of the current joint-based system, but at the same time achieves maximum visual realism in terms of skin deformation.

What needs pointing out is that the term *curve skeleton* has been used for other applications, such as virtual navigation, reduced-model formulation, visualization improvement, surface reconstruction and it was defined as 'a 1D subset of the medial surface of a 3D object.'[2] Despite some similarity, it should not be confused with what we are presenting in this paper. One should neither confuse this with the inverse kinematics (IK) spline handle tool provided by the animation package Maya. Despite their seeming similarity, they are in essence very different techniques.

Related Work

In this Section we briefly review some relevant skinning techniques.

Character Skinning

Mesh deformations due to skeletal joint influence have undergone significant improvements in the recent years. Some of the normal deformation techniques like free form deformations (FFDs) or lattices can be used in skin deformation techniques. Singh and Kokkevis[3] demonstrate this in their paper. They use surface-oriented FFDs for skinning. An interactive deformation technique for complex geometric objects using curves or wires is detailed in Reference [4]. There are basically two main approaches to modeling skin deformations, namely, anatomy-based approach and skin-shape based approach (e.g., example-based skinning).

The anatomical approach derives its name from its implementation using anatomical models of muscles and skeletons and other relevant interior structures. These modules undergo deformation when the body moves and a skin simulation and collision detection algorithm is run which would realistically deform the skin where and whenever it is required. Reference [5] details a technique of efficient muscle shape deformation using the anatomical skin deformation technique. Reference [5] resorts to the creation of a muscle model, which is categorized into two layers: an *action line* and a *surface mesh*. Basically, the action line is the mechanism that drives the deformation. They also implement attractive and repulsive force fields in the form of ellipsoid metaballs to stabilize the action line. Simulation of complex dynamics and performing complex collisions and also providing a visually realistic output form the main strength of the anatomical approach. Incorporating physical properties of anatomy structures can potentially improve realism. Physics can be used either at the muscle level[6] or used to help character rigging.[7] Reference [8] presents another approach to deformation using an elastic surface layer model. It uses a layered structure of anatomical parts from the inside out, *skeleton->bone->fat->skin*. The surface is discretized and finite differencing techniques are used to evolve the deformation through time. The drawback comes in the form of computation expense. The anatomy-based approach is therefore used mainly in high-quality film visual effects where anatomical accuracy is a must for believable computer generated characters.

The example-based approach forms a suitable alternative where computational expenses are to be minimized. This method takes an interpolative approach to deformation. An artist models certain key poses of the characters where a correlation is maintained for the degrees of freedom, in this case, it would be the joint positions or rotational angles. New poses are interpolated from these key poses. A modified least square fitting technique is used to compute the weights of the deformation and the subsequent generalization of skin movement to other animated poses. In Reference [9] the algorithm is trained in a statistical manner so that deformation computation for an arbitrary animated pose can be done. They use a technique called multi-weight enveloping in place of single-weight enveloping for better deformation. Reference [10] also implements an example-based approach to deforming meshes by using radial basis functions to supply the interpolation weights and also for shape interpolation. A variation of example-based approach where key example poses are derived from arbitrary unrelated examples is detailed in Reference [11] where a range scan is used. Thus example-based approaches have the advantage over anatomical approaches by being computationally faster and also due to the fact that creating example poses are much easier compared to creating detailed anatomically correct models. Most of the described techniques are built upon the existing hierarchical skeletal joint system and modify[10] or even create[9,12] new weight calculations to rectify any sort of physical artifacts in the skin deformation. The example-based approach relies on key sample poses to derive a generalization of deformation, and this becomes a major disadvantage, as this in itself is an expensive and time-consuming process. It is not desirable to create many examples and train the system.

Our approach builds upon the existing system using the curve skeleton for a continuous sampling of the skin surface thereby facilitating skin deformations devoid of geometry artifacts. Since our technique falls in between the two approaches, seamless integration with the two is also possible and becomes its strong advantages. A relevant technique to ours is the sweep-based skinning.[13] The body of a character is segmented with a large number of sweep planes which will be transformed by the joint skeleton. These planes are used to guide the transformation of every skin point during animation. With our method, the skin surface does not need to be approximated by sweep surfaces. It will be deformed directly by the underlying curve skeleton, leading to a simpler process.

Our Contribution

Skeleton and skin relationship in the present production pipeline is strictly linear, whereas observation of the various geometry artifacts like *candy wrapper* and *collapsing joints* intuitively point to the fact that linear blending or skeletal space deformation falls short in accurately depicting skin deformations because of their non-linear nature. This non-linear nature is explored in Reference [12] where a spherical blending is proposed. Only the translation factor is most commonly used for the skin vertices and the rotation factor is not considered. It is our knowledge that the problem reduces after weight painting only when the joint influence fall-off follows a curve pattern. Wang and Philips[9] introduce a multi-weight technique to eliminate this problem in a normal joint-based skeleton skinning. However, this requires the generation of a large number of pre-modeled examples in the first place. The solution to the *collapsing joints* problem, which is to place additional joints[1,14] (placing additional joints is basically bringing a curve nature to the joint chain) near the main joint, has the added problems of: (1) creating a new joint in the hierarchy; (2) joint connections have to be done again to connect the new joint in the existing chain; and (3) painting of weights have to be adjusted to accommodate the new joint. With our curve skeleton technique, the curve serves as a duplicated skeleton to the actual underlying joint skeleton. Effectively any point on the curve can be considered as a joint. In other words, the skeleton is equipped with an infinite number of joints, which will influence the skin deformation. The curve nature of the skeleton makes it easier to manipulate it with a great order of flexibility. The idea to use a curve skeleton side by side with the traditional joint skeleton is conceptually simple and functionally efficient giving realistic skin deformations even under extreme mesh duress. Our curve skeleton technique takes full advantage of the non-linearity of the skeleton-skin relationship.

Curve Skeleton

The curve skeleton can be generated in two ways depending on what the animator supplied in the first place. If the animator supplies a skin model and a skeleton model in the traditional manner, the curve skeleton generation is easy. If on the other hand, the animator supplies only a 3D surface model (not a voxelized representation), the generation of the skeleton becomes slightly more complex in that an additional step is required. A temporary copy of the surface model can be created (during runtime) and voxelized. Once voxelized, a curve skeleton is created using the repulsive force field function.[15] Then the temporary mesh can be deleted and the skeleton can be used with the original surface model.

The whole structure of a curve skeleton may involve several curves, which depend on the topology of the original joint skeleton. The three types of topology for skeleton segments used here are:

Linear Linkage (Joint With Two Links)

In a linear linkage the centre of the joint gives the first control point (CP) of the curve. Then one Bone_CP each is inserted on the opposite sides of the Joint_CP (Figure 1a). Both Bone_CPs have floating positions along the two neighbouring bones, its position being constrained by the angle between the two bones. The reason for the floating position is to eliminate the self-intersection of the skin mesh (Figure 1-a1). Before we can predict the exact movement of the Bone_CP, first we should estimate the approximate distance d from the skin surface to the relevant link of the skeleton. The condition for non-self-intersection is to check if the local radius of curvature r at the joint is not less than d. If we analyse the curve function, we can extract the exact expression from the position of Bone_CP, but because the distance d is only an approximate result, it may not fit exactly in the animation. So here the floating position of the Bone_CP is left to the animator to define interactively (Figure 1-a2). By providing the animator with more parameters, which he/she can tweak, we grant flexibility and freedom to adjust the animation. One curve is generated.

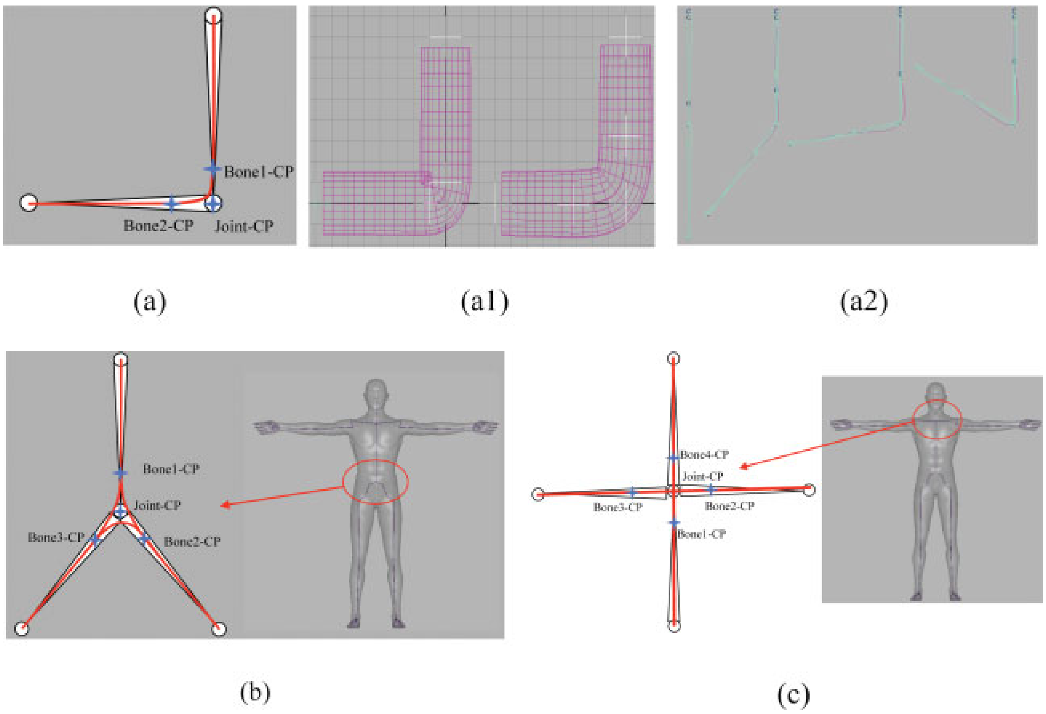


Figure 1. Three types of skeleton topology: (a) linear linkage structure, it shows a schematic diagram of the curve skeleton with the Bone_CPs inserted on the two sides of the Joint_CP; (a1) shows the curve skeleton within the actual skin mesh, the white crosshairs denote the Bone_CP which is placed to avoid mesh self-intersection; (a2) shows the Floating Bone_CP, the pink line denotes the curve skeleton, the joint skeleton is shown in green, the Bone_CPs on either side of the Joint_CP changes position as the joint angle varies. (b) Bifurcation linkage structure, it shows the schematic blow-up of the abdomen area of the skin model. Here we can see that there are three curves for the linkage. (c) Cross linkage structure, it shows the schematic blow-up of the neck area of the skin model. Here we can see that there are two curves for the linkage.

Bifurcation Linkage (Joint With Three Links)

In anatomical areas like the hip (Figure 1b), a fork exists in the joint chain. Hence, three curves are generated, two curves starting from the central link to the two limbs linkage (in the example) and one curve linking the two links.

Cross Linkage (Joint With Four Links)

In the neck area, a cross exists in the joint chain. Although it appears to have four links, we only need to generate two curves for the curve skeleton, as seen in Figure 1(c).

As can be seen from the above classification, for a human character, we will use a maximum of three curves for each joint. In most cases, one curve is sufficient.

Curve Skeleton Motion Synthesis

Representation

In our method, the parameter t on the curve plays an important role in the deformation. We use B-splines to represent the curve skeleton.

Local Frame Definition

Similar to the joint-based skeleton, each point on the curve in a curve skeleton has a local frame (similar to a Frenet frame) defining the space transformation sampled at that point. This local frame is a function of the parameter t of the curve point. The frame on the curve can be defined in two phases:

(i) For the local frame of the point associated with original joints in the joint skeleton. These points normally form the curve segment endings. They can be easily found from the curve definition. At these points, the local frame is defined from the related joint's local coordinate axis: the x-axis x' is the tangent direction at that point on the curve (Figure 2a)

Then the y' and z' can be determined by (Figure 2)

$$\begin{aligned} y' &= z \times x' \\ z' &= x' \times y' \end{aligned} \tag{1}$$

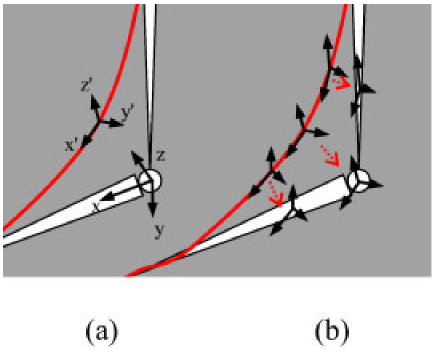


Figure 2. Local frame definition. (a) Shows the local coordinated axis for both the curve skeleton and the joint. (b) Shows the local coordinate axis at each curve point on the curve skeleton transposed onto the joint skeleton.

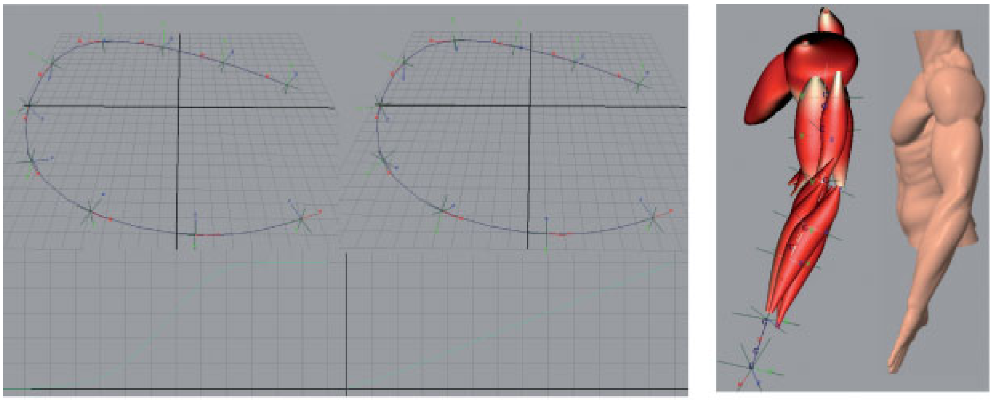
(ii) For the local frame at any point inside the curve segment: This can be calculated by using the frames at the two endings to perform a linear interpolation.

$$\langle x(t), y(t), z(t) \rangle \geq (1-t)\langle x_1, y_1, z_1 \rangle + t\langle x_2, y_2, z_2 \rangle \tag{2}$$

An important feature of the local frame is the centre of the local coordinate system. If the centre is lying on the curve, the deformed skin will move out from underneath the skeleton. The underlying structures like muscles or bones will be exposed. So here the centre of the local frame is translated on to the original skeleton shown in Figure 2b.

Twist

When the bone twists around its local x-axis, it will not have any effect on the associated curve skeleton. This is not acceptable. In order to remedy this problem, here on the curve skeleton we define two extra attributes, *twist angle* and *twist distribution*. The twist angle can be easily queried from the associated joint. The rotation angle is for the curve ending. For each point on the curve, we still need a twist distribution to define how the curve twists along its path. Normally it is not evenly distributed as can be seen from the twist of a forearm. In order to perform even distribution of twisting, we provide the animator with the freedom to control how the curve twists by manipulating the distribution curve. The distribution curve (Figure 3a) is very much like the animation curves in Maya. Here the twist angle is distributed along the distribution curve so that the twisting is smooth and natural.



(a) (b)

Figure 3. Twist operation on curve skeleton: (a) The twist distribution curve is shown in green in the bottom. It helps to evenly distribute the twist along the curve skeleton; (b) Arm twist action on muscles using curve skeleton. Left: The muscle deformation Right: The corresponding skin layer deformation.

Skinning

Binding Skin to the Curve Skeleton

The process of skin binding is to transform each skin surface point $\langle x,y,z \rangle$ at the binding pose to the local frame coordinate system $\langle i, t, \theta, d \rangle$, where i is the index to the specified curve segment, t is the parameter along that curve segment, θ is the rotation angle around the x-axis from the y-axis, d is the distance from the local frame centre. Actually the triple parameter $\langle t, \theta, d \rangle$ may be considered as being expressed in a cylindrical coordinate system. The values $\langle t, \theta, d \rangle$ can be easily computed if we can settle the associated curve segment. Thus the challenging part of the work is to find the associated curve segment, and assign the weighting parameter for each curve segment—skin binding. There is a lot of work[16] associated with the traditional joint-based method, like the containment-binding algorithm, point-to-line mapping, Delaunay tetrahedralization.

The relevant default weight factors w_i of a skin point for the ith curve segment is determined by the distance between the skin point concerned with the relevant curve segments. If a skin point is related with only one curve, which represents the majority of cases, the weight factor is always 1. For those skin points associated with two curve segments, the default weights are proportional to the distances to the relevant curve segments, that is, the further away a skin point is from the curve segment, the smaller the weight is. This is also the case for any skin points associated with three curve segments. In all cases, the summation of the weights are constrained to one, $\sum w_i = 1$. The animator will have freedom to edit the weighting factors in the same way as the smooth skinning.

Given that we have a maximum of only three curve segments for each skin point, weight assignment for a skin point is simpler than the traditional smooth skinning method and the computation for skin deformation is computationally cheaper. Smooth skinning usually involves three weights for each skin point and in many cases there could be as many as five weights. This is worsened if additional joints are placed in order to remedy the unpleasant artefacts. The more the joints, the trickier it is to determine the weight distribution. With our curved skeleton, this problem will almost certainly not arise.

Deforming the Skin With the Curve Skeleton

Once the skin is bound with the curve skeleton, deforming the skin is pretty straightforward. The local coordinates of each skin surface point are transformed with the associated local frame to obtain the new position in the world coordinate system.

So the new point P, is defined by

$$P = \sum_i w_i \mathbf{M}_{(i,t)} P_{L(\theta,d)} \tag{3}$$

where w_i is the weight for the specific curve segment i, **M** (i,t) is the new transformation matrix at the parameter t position along the curve segment i.

As discussed earlier, we use on average a smaller number of weights. This leads to a smaller number of summation terms needed for the calculation of the deformed skin points (see Equation (3)). As a result, our computation speed is at the same order, but is slightly faster than that of the traditional smooth skinning technique.

Muscle Deformation

So far, we have discussed how to realistically skin a character without taking into account the anatomical

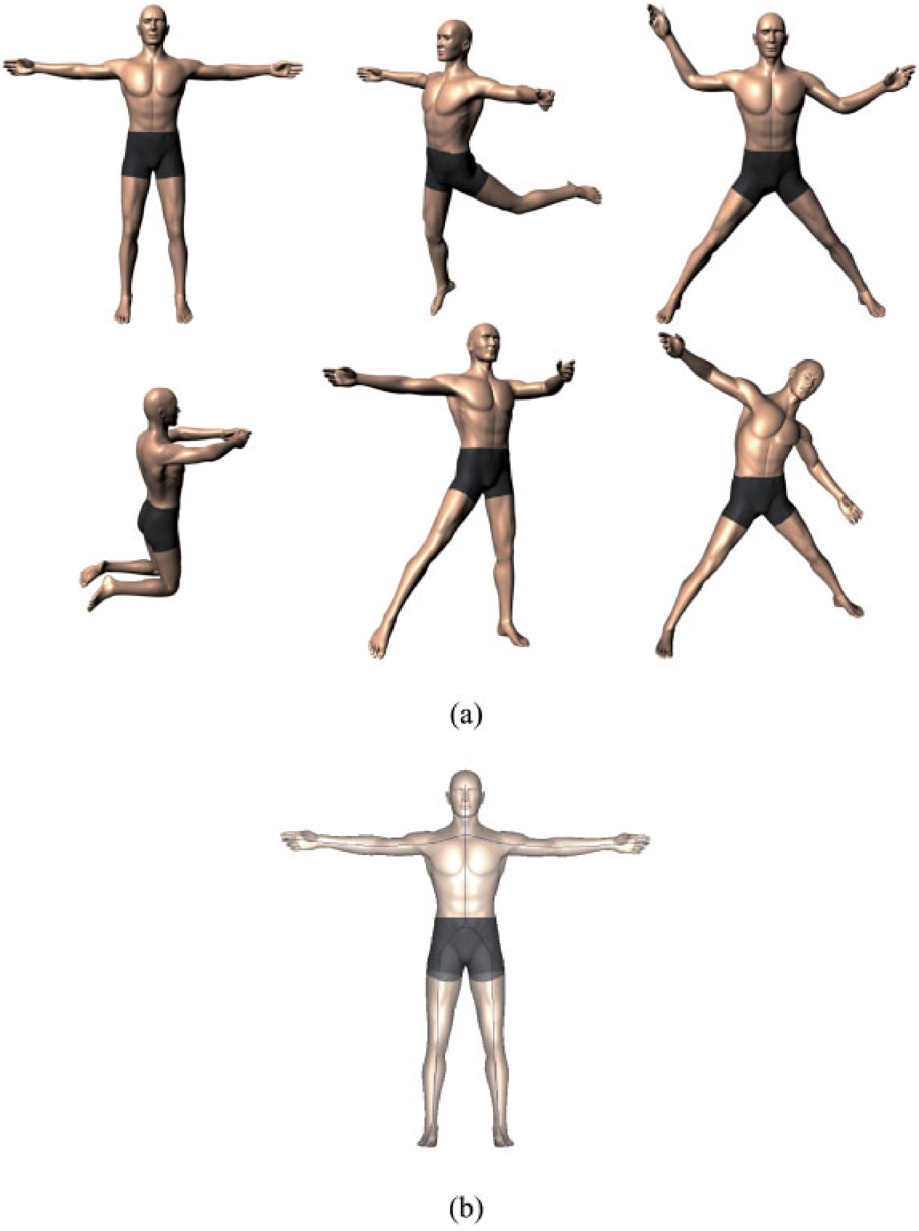


Figure 4. Whole body deformation: (a) Rendered result for the character deformation in different stretched poses; (b) Internal curve skeleton layout. The curve skeleton is shown as lines.

structures. But muscles will give an added layer of realism to the deformation, especially in regions where the skin is visibly influenced by the underneath muscles. With our curve skeleton technique, muscle deformation can be fully integrated where the muscles are driven and animated by our curve skeletons. One of the best third party muscle simulation systems available called *muscleTK*[17] deforms the muscle using the so-called *action lines*. The action line is basically a curve, which defines the direction of deformation. But the disadvantage is that the action line has to be manually animated each frame during animation. Using the proposed curve skeleton, we can realistically deform not only the skin directly (as explained earlier), but also the muscles, in a unified manner. Effectively, each action line is deformed by a curve skeleton, and the action line in turn deforms the muscle. Therefore, we can achieve sophisticated muscle deformations without the tediousness of animating the action lines manually every frame. When the effect of a muscle bending around the joint or the bone is required, we can first transform the control points (CP) of the action line from world space to the associated curve skeleton local frame. These CPs will then be transformed with the curve skeleton, resulting in the muscle bending around the joint or bone being automatically created (Figure 3b).

Implementation of the Curve Skeleton in Maya as a Plug-In

Maya is the most widely used 3D animation package in the industry. In order for scalability and increasing the

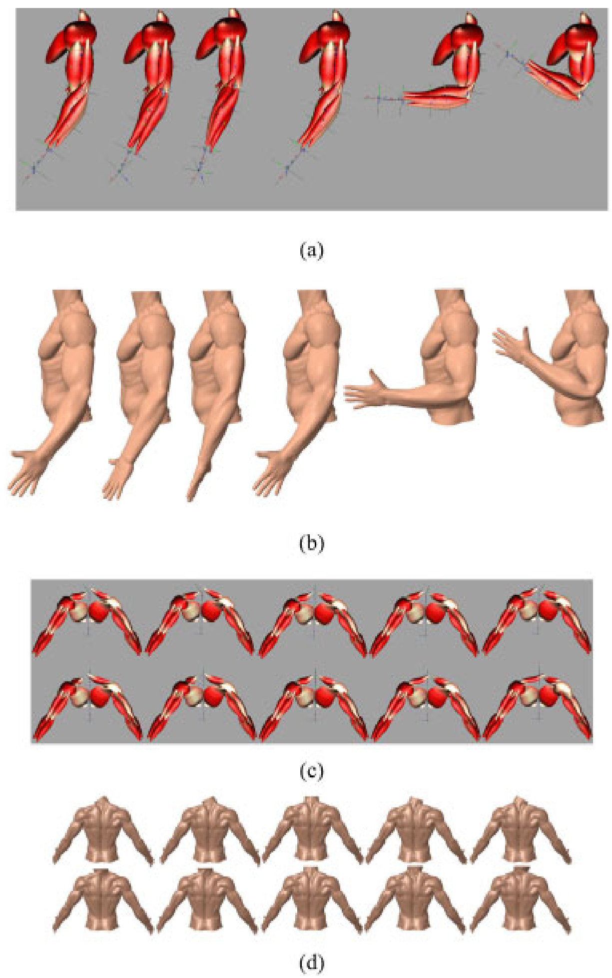


Figure 5. Muscle deformation: (a) Muscle deformation at the elbow during twist and bend. The action line deformed under the curve skeleton movement; (b) skin deformation for the actions in (a); (c) muscle deformation around the neck; (d) Shows the corresponding skin deformation.

feature base of Maya, Autodesk has provided Maya APIs for developers to expand the functionality of Maya. Maya 6.5 and Visual Studio. NET were used in the implementation of the curve skeleton.

From an interface point of view, the artist basically works with normal edit point (EP) curve tools to generate the curves according to his/her wish. Once the curve is selected and the plug-in activated, the curves become the skeleton for the skin mesh.

Internally, the curve cluster is bound to the joint skeleton so that any movement of the joints affects the curvature of the curve. The local transformations of the curve points are applied to the skin mesh vertices thereby generating deformations on the skin.

Conclusion and Future Work

Skin deformation is closely linked with the movement of the skeleton of a character. It is understandable that the relationship between both is highly non-linear, which poses a challenge if the relationship is to be modeled mathematically correctly. Existing skeleton-driven techniques regard it as a much-simplified linear problem, which however, has resulted in unrealistic skin deformation in certain regions of the character body.

In this paper we have presented a technique, known as the *curve skeleton based skinning*, by considering it as a proper non-linear problem. The main advantage of this technique is its consistency with the current animation production practice and the ability to overcome the undesirable drawbacks of skeleton-driven skinning. From the algorithmic point of view, the technique reduces a level of complexity in the skinning and deformation. By layering the curve skeleton on top of the existing joint skeleton, we allow the animator to work conventionally (as in a joint-based system) and yet receive good results. Through a combination of existing practices and newly designed ones, we have successfully created a fusion, which maximizes the efficiency of surface deformation during animation.

For an articulated character, we use no more than three weights for any skin point. In fact, for the majority

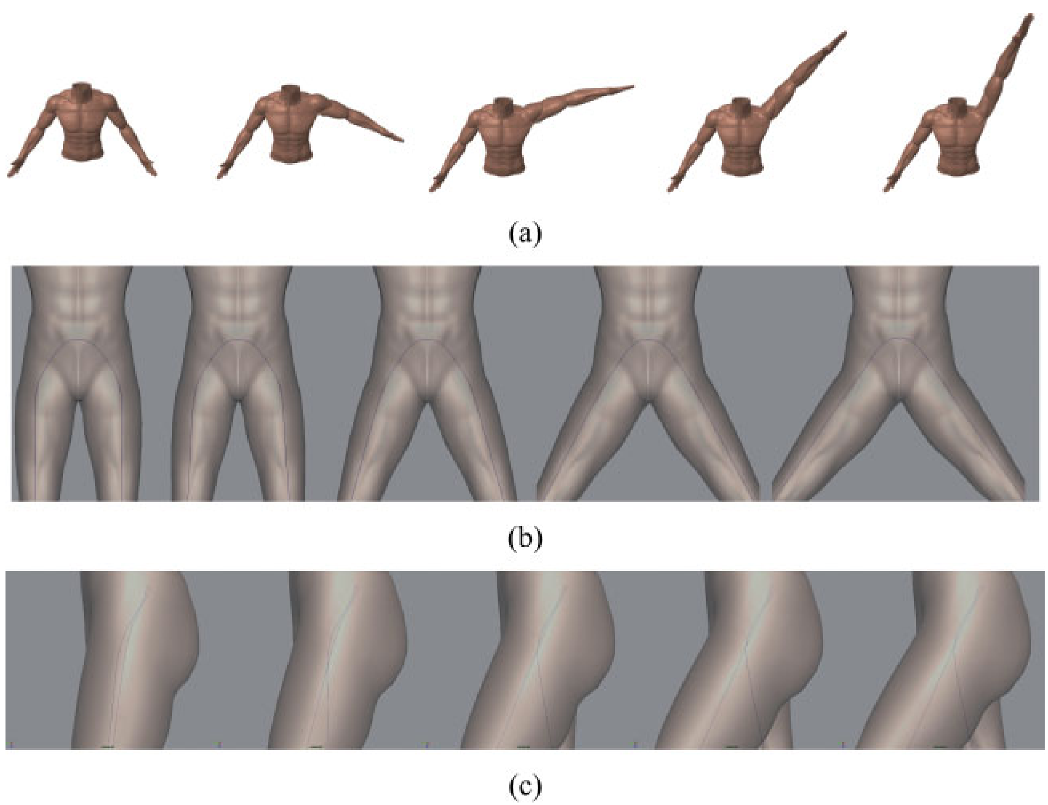


Figure 6. Limb stretch: (a) shoulder deformation; (b) frontal view of the hip and lower abdomen during limb stretching. The curve skeleton is visible as a curving line. (c) Side view hip and lower abdomen while limb stretching.

of cases, there is only one weight, which is 1, to be is used. In comparison with the traditional smooth skinning technique that usually requires on average 3–5 weights, our computation speed is faster.

One should not confuse the curve skeleton technique with the inverse kinematics (IK) spline handle tool provided by the animation package Maya. Despite their seeming similarity, the objective of the Maya IK spline handle tool is to control the joint positions using a spline. Skin deformation is achieved using the traditional smooth skinning technique. Our curve skeleton is controlled by the joints of a character. The skin is directly deformed by the curve skeleton.

The Maya plug-in implementation of the curve skeleton technique has given satisfactory results. One of the main advantages of the curve skeleton skin deformation technique is that, the curve skeleton needs not necessarily be placed on the underlying joint skeleton. With a slight modification utilizing a linear mapping of curve points to the joint skeleton, the plug-in can make use of a displaced curve skeleton which would be useful for subtle deformation on anatomical areas like the armpits. The results of the Maya plug-in can be seen in Figures 4–7.

Our current implementation allows both skin and muscle deformation to be modeled within a unified framework. As future work, we will further improve the skinning realism by adding the fat effect.

Fat usually deposits between the skin and the muscles. Effective realism occurs when the skin actually slides over the fat. This is especially true in the areas near

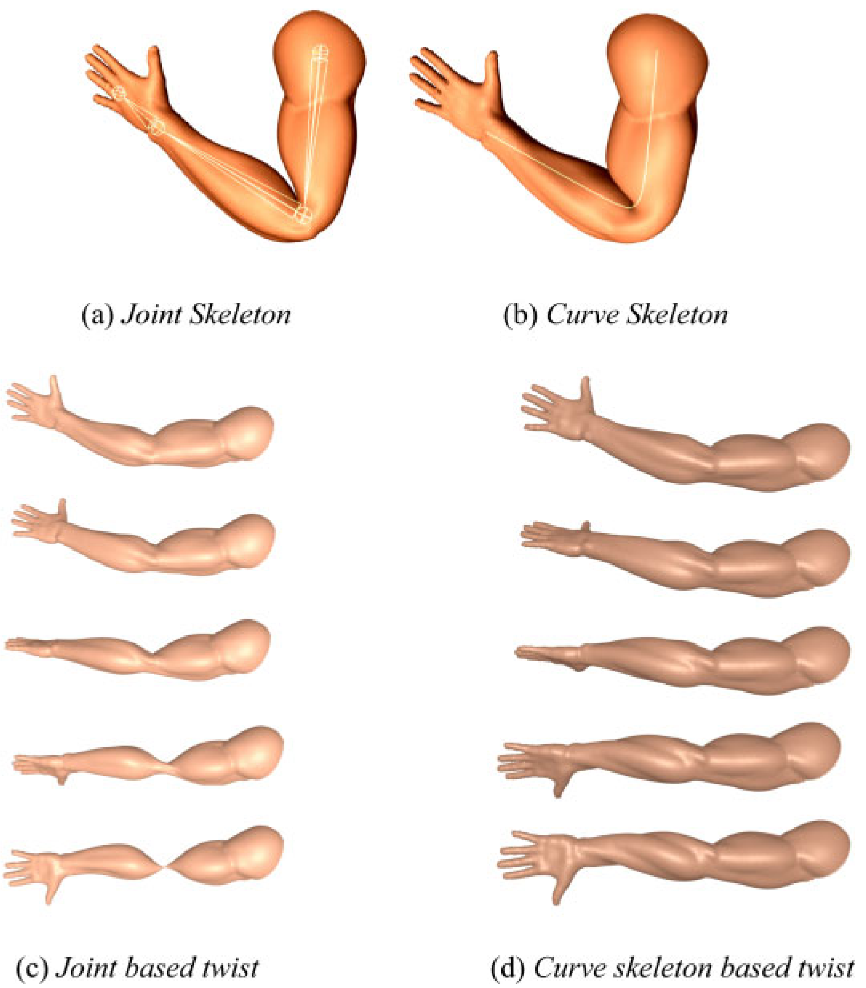


Figure 7. Compare joint skeleton with curve skeleton: (a) collapsing elbow while using joint skeleton; (b) natural effect using the curve skeleton; (c) the candy wrapper problem in joint-based twist. (d) It is absent in the curve skeleton based twist.

joints where acute deformation happens. Turner and Thalmann[8] defines the fat layer as a thickness specified at each point on the skin surface and make use of reaction constraints to push the skin the required distance out from the underlying layers. Yang and Zhang[18] devises a fast method for simulating fat in which a fat bulge distribution function is described. They have used a geometric method instead of resorting to a physical simulation method, and gives convincing results without the computational expense of physical simulation.

With a small modification, the fat bulge effect can be made even in a curve skeleton-based skinning. The function can be defined under the local frame of the curve skeleton. In the present context, since the skeleton is a curve, distribution and deformation can be linked with the tangent angle at a given number of curve points around the joints. As fat is largely incompressible, when a joint bends, flesh between the adjacent bones will be squeezed, producing bulges immediately near the joint and at the sides. Using curve tangents will provide for an accurate distribution in any given time frame because of the integrated results from the sample multiple curve points.

ACKNOWLEDGEMENTS

We are grateful to Autodesk for its donation of the Maya software licenses. We are also grateful to 500 3D-Objects (http://www.taschen.com) without whose high-detailed models practical results would have been impossible.

References

1. Mohr A, Gleicher M. Building efficient, accurate character skins from examples. *ACM Transactions on Graphics* 2003; **22**(3): 562–568.
2. Cornea ND, Silver D, Min P. Curve skeleton applications. In *Proceedings IEEE Visualization*, 2005; 95–102.
3. Singh K, Kokkevis E. Skinning character using surface-oriented free-form deformations. In *Proceedings of Graphics Interface*, 2000.
4. Singh K, Fiume E. Wires: a geometric deformation technique. *Computer Graphics* 1998; 405–414.
5. Aubel A, Thalmann D. Interactive modeling of the human musculature. In *Proceedings of Computer Animation*, 2001.
6. Chadwick J, Haumann D, Parent R. Layered construction for deformable animated characters, computers, SIGGRAPH'89 proceedings, 243–252.
7. Capell S, Burkhart M, Curless B, Duchamp T, Popović Z. Physically based rigging for deformable characters. In *Proceedings of the 2005 ACM Siggraph/Eurographics Symposium on Computer Animation* (Los Angeles California, July 29–31, 2005). SCA'05; 2005.
8. Turner R, Thalmann D. The Elastic surface layer model for animated character construction. In *Proceedings of CG International*, Lausanne, Switzerland, 1993; 399–412.
9. Wang XC, Phillips C. Multi-weight enveloping: least-squares approximation techniques for skin animation. In *SCA'02: Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*, 2002; 129–138.
10. Lewis JP, Cordner M, Fong N. Pose Space Deformation: a unified approach to shape interpolation and skeleton driven deformation. In *SIGGRAPH'00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, 2000; 165–172.
11. Allen B, Curless B, Popovic Z. Articulated body deformation from range scan data. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, 2002; 612–619.
12. Kavan L, Zara J. Spherical blend skinning: a real-time deformation of articulated models. In *Proceedings of the 2005 Symposium on Interactive 3D Graphics and Games*,2005; 9–16.
13. Hyun D, et al. Sweep-based human deformation. *The Visual Computer* 2005; **21**(8–10): 542–550.
14. Weber J. Run-Time Skin Deformation. In *Proceedings of Game Developers Conference*, 2000.
15. Cornea N, Silver D, Yuan X, Balasubramanian R. Computing hierarchical curve-skeletons of 3D objects. *The Visual Computer* 2005; 945–955.
16. http://graphics.ucsd.edu/courses/cse169_w05/3-Skin.htm
17. http://www.cgtoolkit.com/muscletk.htm
18. Yang X, Zhang JJ. Realistic skeleton driven skin deformation. In *Third Technical Session on Computer Graphics and Geometric Modeling, TSCG'2005 Suntec Singapore*, Springer-Verlag: Singapore, 2005.

Authors' biographies:

Xiaosong Yang is a research fellow in the National Centre for Computer Animation, Bournemouth Media School, Bournemouth University, United Kingdom. He received his Bachelors (1993) and Masters degree (1996) in computer science from Zhejiang University (China), Ph.D. (2000) in computing mechanics from Dalian University of Technology (China). He worked as postdoc (2000–2002) in the Department of Computer Science and Technology of Tsinghua University for 2 years, and research assistant (2001–2002) at the 'Virtual Reality, Visualization and Imaging Research Centre' of Chinese

University of Hong Kong. His research interests include 3D modeling, animation, real-time rendering, virtual reality, virtual surgery simulation and computer-aided design.

Arun Somasekharan After completing his B. Tech. in computer systems (2001), Arun Somasekharan worked as a software engineer for 3 years during which he helped in the development of an architectural visualisation program. He also worked on client-server based technologies and database systems. During 2004–2005, he took a break from the rigors of the software industry to pursue his MSc in computer animation from Bournemouth University. Upon graduation, to further his academic interests, he is currently pursuing his PhD in computer graphics and animation from Bournemouth University. His research interests include game-related technologies, character rigging and animation and programmable visual art.

Jian J. Zhang is professor of computer graphics at the National Centre for Computer Animation, Bournemouth Media School, Bournemouth University and head of research at Bournemouth Media School. His research interests include computer graphics, computer animation, physically based simulation, geometric modeling, medical simulation and visualisation.