Reusable Skinning Templates Using Cage-based Deformations

Tao Ju[1] Qian-Yi Zhou[2] Michiel van de Panne[3] Daniel Cohen-Or[4] Ulrich Neumann[2]

[1] Washington Univ. in St. Louis [2] Univ. of Southern California [3] Univ. of British Columbia [4] Tel Aviv Univ.

Abstract

Character skinning determines how the shape of the surface geometry changes as a function of the pose of the underlying skeleton. In this paper we describe skinning templates, which define common deformation behaviors for common joint types. This abstraction allows skinning solutions to be shared and reused, and they allow a user to quickly explore many possible alternatives for the skinning behavior of a character. The skinning templates are implemented using cage-based deformations, which offer a flexible design space within which to develop reusable skinning behaviors. We demonstrate the interactive use of skinning templates to quickly explore alternate skinning behaviors for 3D models.

CR Categories: I.3.7 [Three-Dimensional Graphics and Realism]: Animation

Keywords: cage-based deformation, skinning, templates, animation

1 Introduction

Character skinning, also known as enveloping, is an important step in character animation. It allows for a character's geometry to be animated via a skeletal abstraction that provides a manageable and understandable set of parameters. Skinning models define how the geometric shape changes as a function of the skeleton pose and exist in many variations. This function can be modeled procedurally, as in the case of physics-based or anatomy-based approaches. It can also be modeled in a data-driven fashion by using a regression to estimate the shape for a new pose given a set of example pose-and-shape pairings. Regardless of the details, existing skinning techniques are generally model-specific, meaning that the effort put into obtaining a desired skinning behavior for one model cannot easily be transferred to obtain a similar skinning behavior for another model.

A principle goal of our work is to develop a higher-level abstraction for skinning, one that allows skinning behaviors to be reused across similar joints and similar characters. This abstraction is embodied by *skinning templates*. A skinning template achieves a particular skinning effect, such as a muscle bulging or pinch-free elbow bending and is defined in a way that is not specific to any given model. Skinning templates allow for the rapid exploration of different skinning behaviors, something that is lacking in traditional skinning models. Skinning templates can be swapped in-and-out with a keystroke during an animation preview cycle, thereby allowing for "I'll know it when I see it" selection of an appropriate skinning template. Skinning templates can be shared by users because they are not represented in a model-specific fashion. We expect that they can also be used to improve upon skinning solutions for automatically-rigged characters.

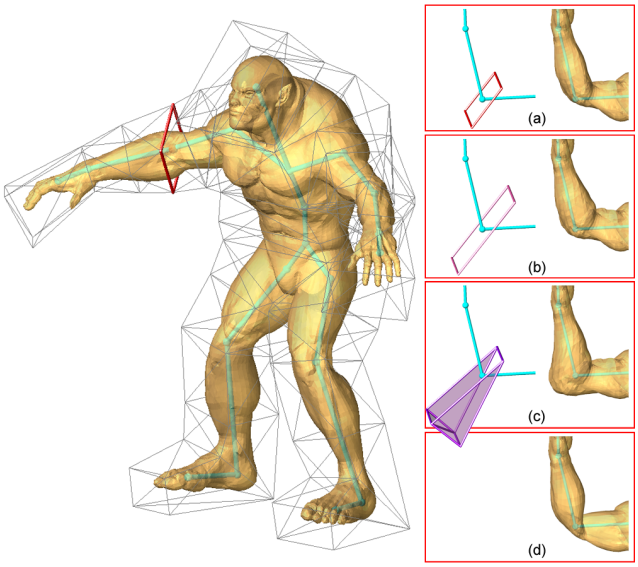

Figure 1: *A character skinned by a cage constructed from templates, with the right elbow template highlighted. (a),(b),(c): The effect of three different skinning templates applied to the right elbow. (d): Linear-blend skinning.*

We use cage-based deformations [Ju et al. 2005; Joshi et al. 2007; Lipman et al. 2007] to implement skinning templates. A cage is defined by a fixed-topology control mesh that is fitted to the character skin. While cage-based techniques allow smooth deformation of skin geometry, posing the cage in these works requires manual manipulation of the cage vertices. In our work, the skeleton drives the motion of the cage vertices using an example-based skinning technique, and the cage then smoothly deforms the character model. The cage has a simple structure that is loosely decoupled from the character model, thereby making it an easy task to design deformation behaviors that can be shared. Cage-based deformations can be efficiently computed, thereby supporting interactive switching of skinning templates. A semi-automatic fitting step is used to ensure that the deformation behavior of the cage remains similar when applied to varying geometries. For example, an elbow-driven muscle-bulge skinning template achieves qualitatively similar muscle-bulge deformations for both skinny arms and fat arms.

Figure 1 illustrates an example usage of the skinning templates for the elbow of a character. The result of applying three different template types are shown, along with a linear-blend skinning result. A library of skinning templates allows for the rapid exploration of a variety of skeleton-driven skinning effects.

2 Related Work

A wide variety of skinning techniques have been developed. They implement a variety of compromises between user control, rigging effort, storage requirements, and speed. Linear-blend skinning (LBS) [Magnenat-Thalmann et al. 1988] is the current predominant technique and is commonly supported in hardware. However, in its naive form, it exhibits well-documented artifacts such as undesired pinching and it offers limited user control over the behavior [Sloan et al. 2001; Merry et al. 2006]. The artifacts can be diminished with the use of extra bones and joints, at the expense of significant additional rigging effort, or these can be inferred with the help of example poses [Mohr and Gleicher 2003]. However, this must be done on a model-specific basis.

Physics-based or anatomy-based techniques model the surface geometry as the by-product of below-the-surface anatomical details [Wilhelms and Gelder 1997], with the help of volume-preserving deformations [Angelidis and Singh 2007], or using dynamic elastic bodies [Capell et al. 2007]. In some cases it is also possible to infer the anatomy-based design from known surface geometry [Pratscher et al. 2005]. These approaches are suitable where user time to setup the internal details of the model is not an issue, and where anatomical realism trumps artistic control. While these types of skinning models can be expensive to compute if a simulation is involved, such models can be a good means to generate high-quality examples for example-based techniques, which we discuss next.

An alternative approach has been the development of example-based techniques, representative examples of which include [Lewis et al. 2000; Sloan et al. 2001; Allen et al. 2002; Kry et al. 2002; Wang and Phillips 2002; Mohr and Gleicher 2003; Merry et al. 2006; Wang et al. 2007]. Broadly viewed, these treat skinning as a regression problem, using sparse sets of example skinned poses and coupled with appropriate regularization techniques. When coupled with appropriate pre-computation effort, these techniques can run efficiently on modern GPUs. These techniques can excel when provided with high quality example poses, although the time needed to create such poses means that these techniques are not suited for fast experimentation with a variety of skinning alternatives.

Character geometry can also be readily manipulated by a variety of geometric deformation methods, including space-based deformation techniques [Sederberg and Parry 1986; Moccozet and Magnenat-Thalmann 1997; Singh and Fiume 1998; Singh and Kokkevis 2000; Ju et al. 2005; Lipman et al. 2007], variational surface deformation methods [Botsch and Sorkine 2008], and example-based surface deformation techniques [Sumner et al. 2005]. These are most commonly driven by control vertices rather than skeletons and do not offer a convenient framework for the exploration of alternative skeleton-driven skinning behaviors. The work of [Joshi et al. 2007] is similar to ours in that it also uses cage-based deformations to achieve rich character poses and discusses the reuse of the same cage animation to work with both preview geometry and detailed final-render geometry.

The use of a skeleton to drive a spatial deformation has deep roots in animation [Burtnyk and Wein 1976; Chadwick et al. 1989]. These are also supported by some commercial animation systems. For example, Maya [Maya 2007] supports the notion of *flexors*, of which several variants exist. While joint and sculpt flexors do not support skinning transfer because they are represented on a specific mesh, the lattice flexor uses a local FFD lattice which can then be driven by joint keyframes. Fitting to a new joint is done using isotropic scaling of the lattice. The use of flexors can require significant setup and tweaking because of the multitude of lattice points and the manual editing of linear-blend skinning weights when binding these to the skeleton. The isotropic scaling may not provide a good fit in many circumstances and may lead to changes in the way adjacent lattices abut or overlap. The spatially-variant deformation behavior of FFD lattices can also make the induced deformations sensitive to the result of the fitting.

More recently, skinning deformations have been developed that generally exhibit significantly better default behavior for bending and twisting joints. Spherical blend skinning [Kavan and Žára 2005] and Dual quaternion blending [Kavan et al. 2008] introduce deformation schemes that develop effective rotation-based interpolations to replace the linear interpolation of frames used by LBS. Spline-based deformations [Forstmann et al. 2007] create a spline curve that spans a desired joint and uses the resulting Frenet frame to create a binding for mesh vertices. The skeleton then drives the motion of the spline. Deformation styles are introduced on top of this scheme using two variants of sweep-based FFDs that allow anisotropic scaling during the sweep. A painting metaphor is suggested for specifying the required 'scaling texture'.

The NeuroEnveloping system of [Guo and Wong 2004] examines the problem of transfering skeleton-driven skin deformation behaviors. A neural network is used to estimate the shape as a function of pose. The resulting shape is then modified for a target character by adding a residual offset to take into account its different shape. The results are demonstrated from a single source model to a single target model for one skinning style, and thus its effectiveness as a transfer technique is difficult to assess.

2.1 Contributions

The method we present is distinct from previous work in a number of ways. It builds connected bone and joint deformations that integrate together in a single seamless cage. This avoids potential artifacts that may arise when more local joint-specific and bone-specific schemes, such as FFD lattices, begin to interact with each other. The fitting of the templates to a new geometry is significantly automated – fitting a full set of 13 joint templates and 8 bone templates is accomplished in minutes using a small number of manual edits to a default automated fit. Fitting needs only be done once, after which different deformation styles can be swapped in through simple selection from a library of predefined styles.

Deformation behaviors are easily specified using an example-based approach and involve a small number of vertices or control parameters, as compared to FFD-lattice or spline-scale approaches. Underlying both the joint and bone templates is the use of rotation-based interpolation techniques that avoid LBS-type artifacts. Of particular significance is that the joint and bone template representations are specifically designed *so as to preserve the intent of the deformation even after the template is adapted to fit a new geometry*.

Most importantly, skinning deformation styles are a first-class abstraction in our work. They support rapid experimentation with many different common skinning effects. While individual effects could be achieved on a specific mesh or character by other skinning techniques, these techniques do not allow the same skinning style to be easily applied to a different character geometry. When combined with a complementary technique such as automatic rigging [Baran and Popović 2007], skinning templates help provide an end-to-end path for quickly proceeding from an unrigged skin to a fully rigged character with desired styles of skeleton-driven deformations.

3 Overview

Cage-based deformations lie at the heart of our method. As illustrated in Figure 2, a *cage* is constructed by piecing together *tem-*

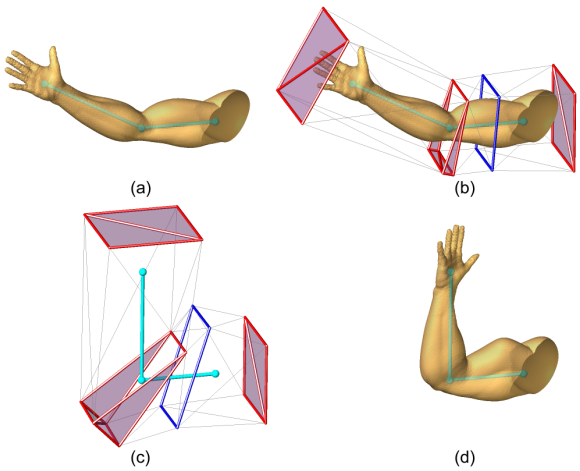

Figure 2: *Skinning with cage templates: (a) Input geometry with skeleton. (b) An initial cage constructed from four templates. From left to right, these are associated with the hand joint, elbow joint, upper arm bone, and the shoulder joint. Joint templates are red while the bone template is blue. (c) The skeleton deforms the template vertices. (d) The geometry is deformed by the cage, yielding a non-pinching elbow and muscle bulging.*

plates. A cage is a coarse control mesh which wraps around the character skeleton and loosely encloses the character skin. A template is a partial cage. It consists of a set of vertices, connected by edges and triangles, which wraps around a specific skeleton joint or bone. Figure 2(b) illustrates an example cage construction. Given the arm geometry and an arm skeleton, a cage is constructed using four templates associated with the hand joint, elbow joint, upper arm, and shoulder joint, respectively (joint and bone templates are colored red and blue). Each template has a topology that fits the relevant skeleton part, and its geometry loosely fits the embedded skin. Note that the templates at the extremal joints, such as the hand, are shaped in a way so that the cage fully encloses the skin geometry.

Our skinning method deforms character geometry in a two-step process. First, the skeleton drives the deformation of the cage, as shown in Figure 2(c). The cage then drives the deformation of the character geometry, as shown in Figure 2(d). In essence, we first "skin the cage" instead of directly skinning the geometry. Reusable skinning templates are made possible because the cage has a simple structure and is loosely decoupled from the geometry. The cage vertices are themselves skinned in an example-based fashion, i.e., the template designer has a degree of control over how they move as a function of the coordinate frames of the relevant bones. Given an arbitrary closed triangular surface as the cage, several recent methods can be applied to achieve the cage-driven deformation of the embedded geometry [Ju et al. 2005; Joshi et al. 2007; Lipman et al. 2007]. We note that cages generate smooth deformations of the contained geometry in contrast to linear-blend skinning schemes. This is illustrated in Figure 3.

A particular challenge in our framework is that the templates should support the intent of example-based deformations of a template even after being adapted to fit the specific geometry of a character. This means that we cannot "skin the cage" with standard techniques. We develop specific representations and deformation functions that support these unique requirements.

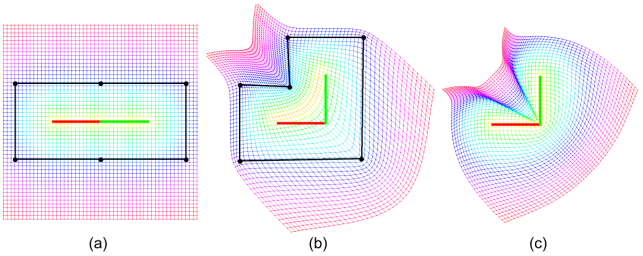

Figure 3: *Deforming a 2D grid (a) by two bones (colored red and green) using a 2D version of the proposed skeleton-driven cage-based deformation (b) (the cage is drawn in black) and using linear-blend skinning (c). The grid points are colored by their distances to the bones in the rest pose in (a). Observe that cage-based deformation yields smooth deformation interior to the cage, while linear-blend skinning easily results in fold-backs.*

3.1 Templates

A template has a fixed topology, which is related to the topology of the skeleton, and a dynamic geometry, which is driven by the pose of the skeleton.

3.1.1 Template topology

Each template is a triangular mesh with boundaries, where each boundary consists of a ring of vertices to be connected with neighboring templates. In particular, each bone template has two boundary rings, and each joint template has the same number of boundaries as the number of outgoing bones from that joint. Often, a template contains no triangles (e.g., the bone template at the upper arm in Figure 2 (b)), and different boundaries may share common vertices.

3.1.2 Template geometry

To drive the cage using the skeleton, each template vertex is associated with a pre-designed deformation function that yields a specific animation effect for the nearby skin region. This function determines the location of a template vertex by the pose of its *relevant* skeleton bones, which are attached to the template joint or to an end of the template bone.

To help avoid artifacts such as the pinching effects of linear-blend schemes, a key feature of this deformation function (to be discussed in Section 4) is that it preserves the radial distance from template vertices to the template joint or bone after deformation. In a joint template, the deformation function consists of rotations around the joint. In a bone template, the function consists of rotations around and translations along the bone . Examples of such radial-type deformations are shown in the top row of Figure 4 for a joint template and Figure 5 for a bone template, given locations of the template vertices in an initial pose of the relevant bones (drawn as cyan and red arrows on the left). Observe that the deformations respond naturally to both bending and twisting bones.

While the radial-type deformations yield plausible skin behaviors, they cannot express "interesting" effects such as muscle bulging. To this end, we allow the user to provide a set of examples, each consisting of a *key pose* of the relevant bones, as well as the locations of template vertices at that pose. The template deformation function then interpolates these key pose vertices while still striving to preserve the radial distances of template vertices. Some examples are shown in the last two rows in Figure 4 and 5. Observe that

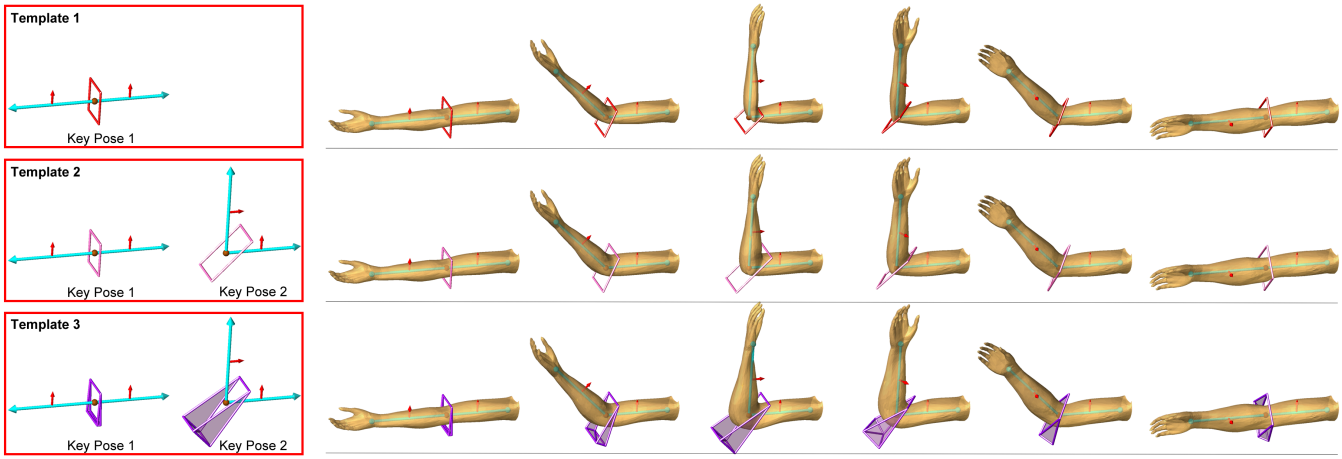

Figure 4: *Left: several joint templates and the examples used to define them. Right: the deformation of the template vertices defined by the examples as the skeleton pose changes, and the effect on a geometry. Bones are colored cyan, and the template joint is colored brown.*

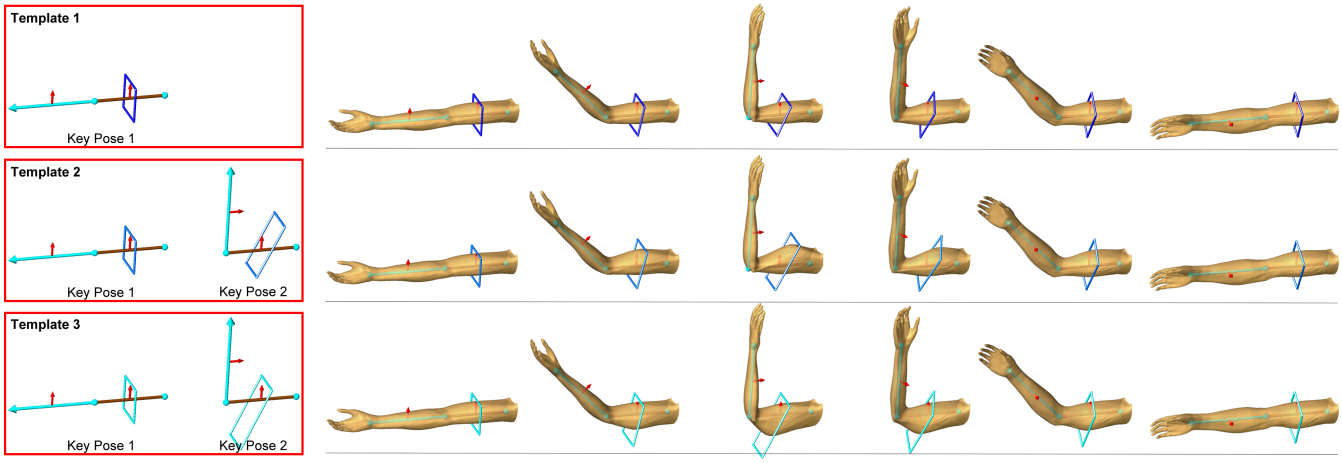

Figure 5: *Left: several bone templates and the examples used to define them. Right: the deformation of the template vertices defined by the examples as the skeleton pose changes, and the effect on a geometry. Bones are colored cyan, and the template bone is colored brown.*

the key poses can express a variety of deformation styles, including ones that are physically unrealistic. For example, Template 3 of Figure 4 yields an exaggerated elbow bulging, and Template 3 of Figure 5 expresses a muscle bulging in the opposite direction of the bend.

3.1.3 Template library

A key feature of a template is that, once defined, it can be used for different joints or bones on various skeletons. Template re-using is based on the notion of *compatibility*. In particular, a joint template can be used at (or *compatible* with) any joints that has the same number of outgoing bones as the template joint, and a bone template is compatible with any bone with the same number of relevant bones.

Compatibility allows a complex character to be skinned using a small *library* of templates. In our current implementation, the six templates on the left of Figures 4 and 5 make up the bulk of the library. These templates are used on all bones and two-bone joints (including shoulder and belly) in our examples. The library includes two additional joint templates with three or four incident bones (see Figure 13 for a 3-bone joint), each containing a single key pose. The simplicity of the templates allows the library to be easily populated for additional skinning effects. For example, the exaggerated muscle behaviors in Figure 12 are achieved by simply modifying the second key pose in the bone templates in Figure 5.

3.2 Skinning using templates

As mentioned above, the templates are defined once by artists, and are placed into a library for re-use. With a library of pre-designed templates, skinning is simple and interactive. To set up for skinning, a cage is first constructed at the rest skeleton pose, while respecting the size and shape of the geometry. Skinning then involves selecting an appropriate template at a skeleton joint or bone, typically during an animation preview cycle, where the user can observe the deformation effect updated on the fly.

Cage setup Given the geometry at a rest pose with an embedded skeleton, the system first selects the *default* template compatible

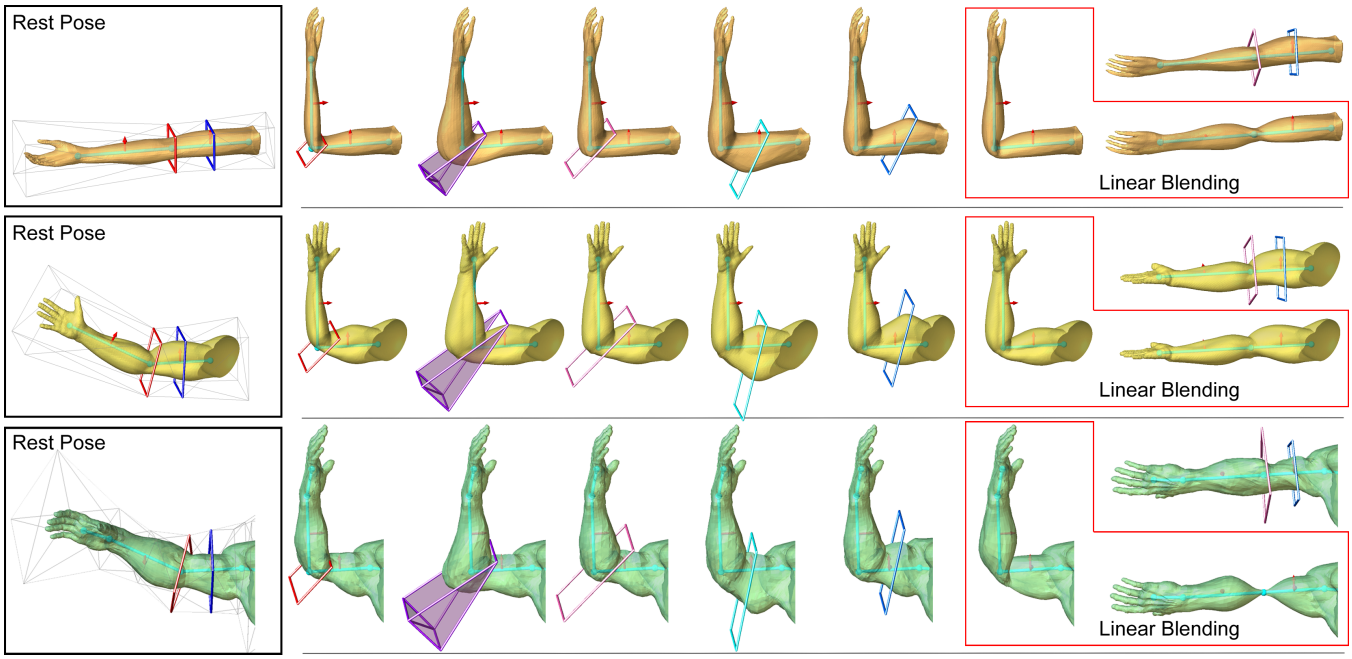

Figure 6: *Exploring deformation effects on different geometry (one in each row) using the templates defined in Figure 4 and 5. Observe that one template achieves similar deformation effects on different models. Additionally, the use of templates and cages avoids the artifacts associated with linear blending. The third geometry is a portion of the Monster model in Figure 1.*

with each joint and bone. The default template for a joint or bone type is designed to contain a single key pose (e.g., Templates 1 in Figure 4 and 5). The default templates of neighboring joints and bones on the skeleton are then interconnected into a cage, which is adjusted to fit the actual skin geometry. We discuss in Section 5 our semi-automatic procedure for cage fitting. Note that, since the cage only needs to loosely surround the geometry, a tight and accurate fit is not necessary for the purpose of transferring deformation styles.

Skinning After the cage is constructed, the geometry is "bound" to the cage using cage-based deformations. In our implementation, we adopt deformations based on Positive Mean Value Coordinates [Lipman et al. 2007]. As the skeleton pose changes, the template vertices are deformed (or skinned) according to their fitted deformation functions, which in turn drives the character geometry. The user may select a different template from the library at a joint or bone during the animation, and observe the deformation effect gets updated. Since both cage-based deformations and template deformations are efficient, such changes can take place on the fly. In Figure 6, we demonstrate the templates from Figure 4 and 5 applied to three different geometry examples. Observe that a common template results in similar deformation effects.

4 Template deformation

We now discuss in detail how a template vertex deforms as a function of the pose of a set of relevant bones. In particular, such deformation should interpolate the vertex locations in a set of key poses. Following the traditional linear-blend schemes, we construct the deformation in two blending steps. First, the vertex is transformed by each relevant bone starting from a key pose, and the transformed vertex locations are blended. Next, the deformed locations associated with each key pose are blended together to yield the final vertex location.

To avoid artifacts such as skin collapsing, the key is to maintain the radial distance from a template vertex to a joint or bone during this deformation. We differentiate between a joint template and a bone template in how a vertex is transformed by a bone and how vertices are blended. In a joint template, such transformation involves only rotations around the joint. In a bone template, it involves rotations around or translations along the bone. The blending is done in a spherical or cylindrical coordinate frame around the template joint or bone.

Specifically, consider a set of examples $\{B_i, v_i\}$ where B_i is a key pose of relevant bones and v_i is the location of the template vertex in that pose. Here the pose of each bone is represented by two orthogonal vectors (cyan and red arrows in Figures 4 and 5 left). Our goal is to construct a deformation function $f(B)$ for any input pose B, so that $v_i = f(B_i)$ for all examples i.

We first construct a deformation function $f_i(B)$ for each key pose B_i so that $v_i = f_i(B_i)$. This is done by blending the locations of v_i transformed by each relevant bone from key pose B_i to B:

$$f_i(B) = \bigoplus_j u_j * M_j({M_j}^{-1}(v_i, B_i), B). \quad (1)$$

Here $\oplus$ is a summation operator, u_j is the influence of the j-th bone to the vertex, and $M_j(v, B)$ transforms a vertex v by aligning the world coordinate frame to the local frame of the j-th relevant bone in pose B. Note that the formula is similar to that used in linear blending, where M_j consists of unrestricted rigid-body transformations, and $\oplus$ is a simple linear sum in the x, y, z coordinates. In contrast, we will define M_j and $\oplus$ differently for each type of template to preserve radial distance of a template vertex, which will be discussed below.

Assuming there are multiple key poses in a template, we obtain the final deformed vertex $f(B)$ in a second blending phase involving

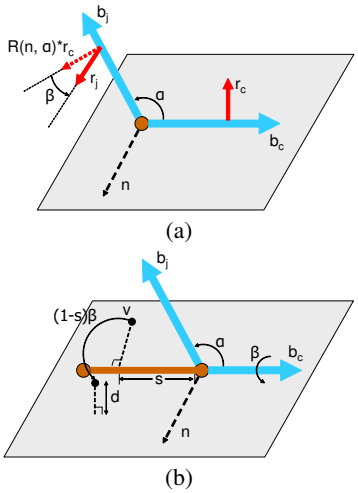

Figure 7: *Notations for template deformation functions.*

deformed locations $f_i(B)$,

$$f(B) = \bigoplus_i w_i * f_i(B), \quad (2)$$

where w_i is the influence of key pose B_i on the input pose B. As suggested by [Lewis et al. 2000], we obtain w_i using radial-basis functions, where the distance between two poses B and B_i are set as the sum of the angles between corresponding bones in each pose, after all poses are aligned to the local frame of the template bone or a selected bone (e.g., the parent bone in a hierarchical skeleton) attached to the template joint. Note that w_i can be efficiently computed, since each template only has a small number of relevant bones (maximally 4 in a humanoid skeleton) and key poses (typically 2 as shown in Figures 4 and 5).

4.1 Joint template deformation

Assuming the world origin lies at the template joint, the transformation $M_j(v, B)$ in a joint template is simply the rotation aligning the world coordinate system to the bone. Denote the first two axis of the world coordinate frame as $\{\vec{b_c}, \vec{r_c}\}$, and the two orthogonal vectors associated with the j-th bone in the pose B as $\{\vec{b_j}, \vec{r_j}\}$. $M_j(v, B)$ can be written as the product of two 3D rotations,

$$M_j(v, B) = R(\vec{b_j}, \beta) * R(\vec{n}, \alpha) * v, \quad (3)$$

where R is the rotation matrix around an axis passing the origin with a given direction, $\vec{n} = \vec{b_c} \times \vec{b_j}$, and α, β are respectively the angle between $\{\vec{b_c}, \vec{b_j}\}$ and between $\{R(\vec{n}, \alpha) * \vec{r_c}, \vec{r_j}\}$, as shown in Figure 7 (a). While there are many ways of expressing $M_j(v, B)$, we choose this form in order to be consistent with the following discussion regarding bone template deformations.

To blend the transformed vertices while preserving their distances to the joint, the weighted sum $\oplus$ in Equations 1 and 2 is performed separately for the magnitude and direction of the operand vertices. We employ the spherical averaging method of [Buss and Fillmore 2001] based on least-square minimization for averaging unit vectors in a robust manner. For the influence u_j, we found that setting it to be inversely proportional to the angle between v_i and the j-th bone in the key pose B_i yields reasonable results in all our test examples.

Figure 8 top compares our deformation function at a joint with linear blending. To demonstrate the distance preserving feature of our

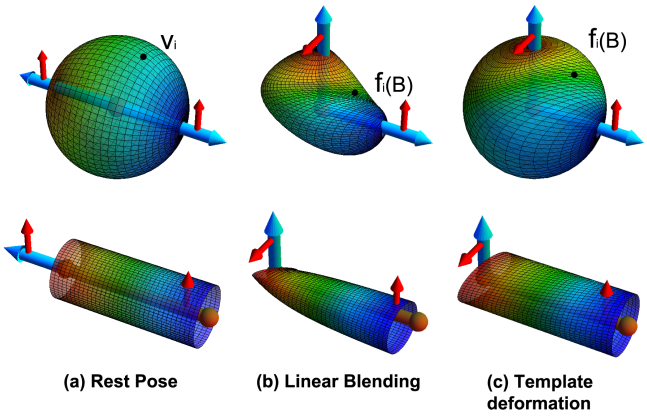

Figure 8: *Comparing our template deformation functions with linear blending at a joint (top) and bone (bottom), as a relevant bone bends and twists. The template vertices form a sphere (top) or cylinder (bottom) and are colored by the influences from the deforming bone. While linear blending exhibits shrinking, our method preserves distances to the joint or bone (i.e., the sphere or cylinder stays as a sphere or cylinder).*

deformation function, we let the template vertices lie on a sphere centered at the joint (see Figure 8(a)). The input pose B differs from the key pose B_i by a 90 degree bend and twist of one of the bones. Observe that linear blending produces shrinking of distances to the joint, resulted from the linear sum in Cartesian coordinates.

4.2 Bone template deformation

To construct transformations M_j for a vertex in a bone template, we observe that such vertex typically exhibits two types of deformations as a relevant bone attached to an end of the template bone deforms. First, it may rotate around the template bone as the relevant bone twists around its own axis. Second, it may shift along the template bone away from the relevant bone as it bends toward the template bone. Note that the amount of rotation and shifting decreases with the distance from the vertex to the joint where the relevant bone is attached, while the amount of shifting may increase as the vertex lies further away from the template bone.

Based on these observations, M_j for a bone template is constructed by a rotation followed by a skewed translation. We assume that the world origin lies at the joint where the j-th relevant bone is attached to the template bone, whose first axis points in the *opposite* direction of the template bone, representing the unbent pose of the j-th bone (see Figure 7 (b)). As the world coordinate frame twists and bends into that of the j-th bone, v is transformed as

$$M_j(v, B) = T(\vec{b_c}, -(1-s)d\tan(\frac{\alpha}{2})) * R(\vec{b_c}, (1-s)\beta) * v, \quad (4)$$

where α, β, R are as defined in Equation 3, T is the translation matrix along a given vector, s is the ratio of the distance from v to the joint (where the j-th bone is attached) in the direction of the template bone over the length of the template bone, and d is the distance from $R(\vec{b_c}, (1-s)\beta) * v$ to the plane spanned by $\vec{b_c}$ and $\vec{n} = \vec{b_c} \times \vec{b_j}$, as shown in Figure 7 (b). Intuitively, the translation models a "skewing" of space that aligns the bisecting plane between the template bone and the first axis of the world coordinate frame to that between the template bone and the j-th bone. The former is orthogonal to the template bone while the latter is at $\frac{\alpha}{2}$ angle to the template bone.

The weighted sum operator $\oplus$ in Equations 1 and 2 is defined similarly to that in joint template deformation, except that now there are three components to be independently summed for the transformed vertices: the distance from each vertex to the template bone, the distance from each vertex to one joint in the direction of the bone, and the unit direction of each vertex in the plane orthogonal to the bone. For the influence u_j, a reasonable choice we found is inversely proportional to the ratio of the distance of v_i to the joint (where the j-th bone is attached) in the direction of the template bone over the bone length.

Figure 8 bottom compares the result of our deformation function to linear blending near a bone, where the template vertices form a cylinder (see (a)). In linear blending, M_j are the affine transformations aligning the world coordinate frame to the local frame of the relevant bones, which are not restricted to translations along or rotations around the template bone. Combined with the linear sum in Cartesian coordinates, linear blending produces pinching of the cylinder, while our method exactly preserves the cylindrical shape.

5 Template fitting

To animate the skin by the template deformation functions computed above, we take advantage of a recently developed cage-based deformation technique, Positive Mean Value Coordinates (PMVC) [Lipman et al. 2007]. Given a skin geometry and its skeleton at the rest pose, we first construct a cage by connecting templates at neighboring bones and joints, and express each vertex on the skin as a weighted combination of cage vertices whose weights are computed by PMVC. As the skeleton pose changes, cage vertices are deformed by the template deformation functions, and the skin is deformed in turn by applying the PMVC weights to the deformed cage vertices. In comparison to the original Mean Value Coordinates technique [Ju et al. 2005], PMVC (and also Harmonic Coordinates [Joshi et al. 2007]) ensures positive weights at each skin vertex and avoids artifacts when deforming body parts at close proximity.

A pre-requisite of cage-based deformations is a cage that loosely surrounds the skin at the rest pose. If the skin has a different shape than what is used initially to define the templates, the templates (and their deformation functions) need to be first adjusted to properly embed the skin geometry. Our fitting is completed in two steps. First, an initial cage constructed from default templates at each joint or bone is fitted onto the skin at rest pose, using automatic alignment followed by user correction if necessary. After this initial fitting, the deformation function of these default templates as well as of all other compatible templates in the library are automatically adjusted, so that no further interaction is needed when the user deforms the skeleton or switches between templates.

5.1 Initial cage construction and fitting

The initial cage is constructed from the default template at each joint and bone, which are first deformed from their respective example key poses to the rest pose in the given skeleton. Template vertices at neighboring bones and joints are then interconnected by triangles with minimal surface area [Fuchs et al. 1977] to create a closed cage.

To adjust the initial cage to the geometry, each template vertex is automatically scaled radially from their respective joint or bone. Specifically, we obtain the thickness of the skin at a joint or bone by the shortest Euclidean distance from the joint point or the bone line to the skin, and uniformly scale all vertices in a template by the ratio between the thickness of the actual skin over that of the skin from which the template is defined. Vertices in a bone template are additionally scaled along the bone direction by the ratio of the actual bone length over the length of the bone used to define the template. To avoid intersection of the cage with the geometry, further up-scaling is applied at small increments to a template intersecting with the skin until either the intersection is resolved or a user-defined maximum scale is reached.

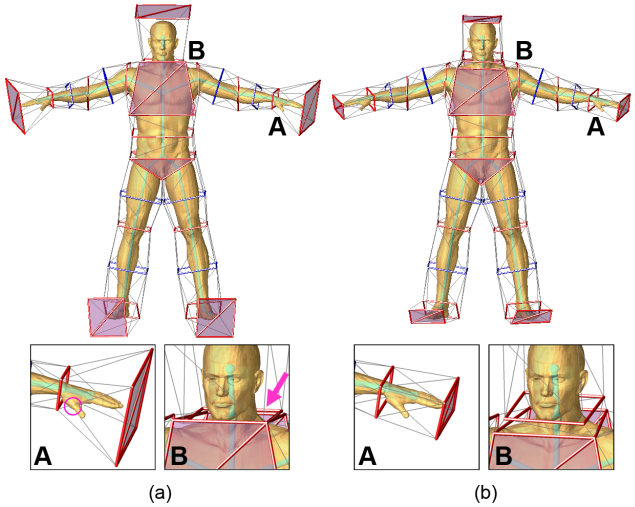

Figure 9: *An initial cage automatically constructed and adjusted to a human model (a) and after manual correction (b). Joint and bone templates are colored red and blue respectively.*

An example of automatically scaled cage for a human model is shown in Figure 9 (a). Note that while the cage has a satisfactory shape in most places, it needs further improvement in two scenarios. First, the skin geometry may still intersect with the cage, possibly at the triangle faces connecting neighboring templates (circled in insert A) or where uniform scaling cannot resolve the intersections. Next, the cage may contain triangles that do not conform well to the shape of the skin because of congestion of neighboring templates (where the arrow points in insert B), which may cause skin artifacts when applying cage-based deformations. While it is difficult to express these scenarios as constraints for optimization, we found that they can be corrected rather easily and quickly by a user with only a few mouse clicks, the result of which is shown in Figure 9 (b). We have observed that such user interaction is typically small in an animation exploration session, which is dominated by user switching between templates and examining the deformations after the initial fitting.

5.2 Adjusting deformation functions

Once the initial cage is adjusted to the skin at the rest pose, the deformation function of the default and all compatible templates will be automatically adjusted as follows. We first consider adjusting the deformation function of the default template, whose k-th vertex is associated with a deformation function $f^k(B)$. Let the rest pose of the skeleton be $\overline{B}$ and the fitted template vertices in this pose be $\overline{v^k}$. This boils down to constructing adjusted functions $\overline{f^k}(B)$ such that $\overline{v^k} = \overline{f^k}(\overline{B})$ (see Figure 10). To do so, we first construct transformation D_j^k that aligns $f^k(\overline{B})$ to fitted location $\overline{v^k}$, in the local frame of the j-th bone in $\overline{B}$. That is,

$$D_j^k * M_j^{-1}(f^k(\overline{B}), \overline{B}) = M_j^{-1}(\overline{v^k}, \overline{B}) \quad (5)$$

The matrix D_j^k consists of rotations and scalings that align the corresponding components between the two vertices. As discussed in

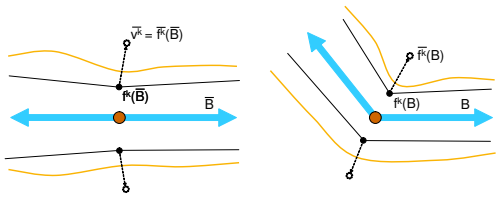

Figure 10: *2D illustration of template fitting. Left: The cage (black lines) constructed from template vertices (black dots) may not embed the rest-pose geometry (brown curves), and the locations of the fitted vertices (circled dots) are provided, either automatically or by manual adjustment. Right: The deformation function ($f^k(B)$) is adjusted ($\overline{f^k}(B)$) accordingly for all other poses.*

template deformation, the components include a magnitude and a direction for a joint template vertex, and two magnitudes and a direction for a bone template vertex. Next, for a new pose B, the adjusted deformation is an weighted average of the transformed locations,

$$\overline{f^k}(B) = \bigoplus_j \mu_j * M_j(D_j^k * M_j^{-1}(f^k(B), B), B), \qquad (6)$$

where μ_j is the influence of the j-th bone in the rest pose $\overline{B}$ on $f^k(\overline{B})$, computed similarly to u_j in Equation 1, and $\oplus$ is the summation in the corresponding direction and magnitude components as in Equations 1 and 2. One can verify from Equations 5 and 6 that $\overline{f^k}(\overline{B}) = \overline{v^k}$.

The deformation functions associated with a different, compatible template can be adjusted automatically in a similar manner. Note that different templates (e.g., 2 and 3 in Figure 4) may differ in the number of vertices that embody specific deformation effects. To accommodate such difference, we re-write Equation 6 in a more general form,

$$\overline{g^m}(B) = \bigoplus_j \mu_j * (\bigoplus_k \omega_k * M_j(D_j^k * M_j^{-1}(g^m(B), B), B)).$$

where $g^m(B)$ is the deformation function associated with the m-th vertex of any compatible template, and ω_k is the influence of $f^k(\overline{B})$ on $g^m(\overline{B})$ in the rest pose. In our experiments, we found that setting the weights ω_k to be inversely proportional to the Euclidean distance between the two vertices yields reasonable results in our examples.

5.3 Tightness of cage embedding

The adjustment of template deformation functions is determined by how the initial cage is fitted to the skin at the rest pose. With the involvement of user interaction, the initial fitting is subjective to user's judgement and the cage may not embed the skin "perfectly" with a same tightness everywhere. One of our key observations is that cage-based deformations, such as PMVC, are rather insensitive to the tightness of cage embedding. We demonstrate the stability of skin deformation under varying cage tightness in Figure 11. Note that tightness of embedding has only small impact on the style of deformation conveyed by the templates, where tighter cages tend to yield slightly more localized deformations.

6 Results

We implemented the template-based skinning method in an interactive tool. Given a character with skeleton, the cage is constructed automatically followed by interactive editing. A number of full-body cages constructed using the tool are shown in Figure 9 (b) and 12 left. After the cage is constructed, users can observe changes in deformation effects as they switch templates while joints and bones are undergoing cyclic motions (e.g., bending and twisting). The interaction is demonstrated in the accompanying video.

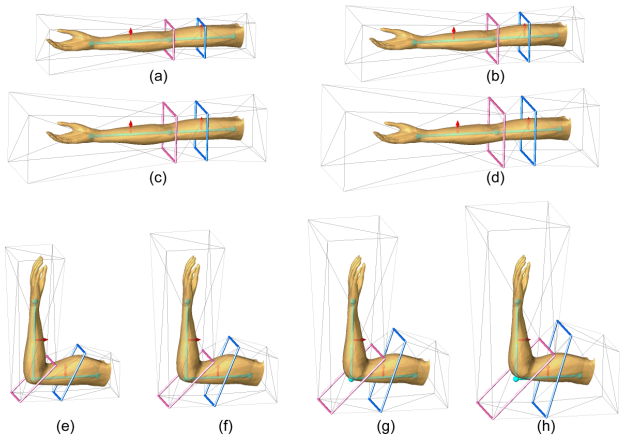

Figure 11: *Cages that fit a geometry with decreasing tightness (a to d), and their resulting deformations (e to h) using a same combination of templates (number 2 in Figure 4 and 5).*

We demonstrate template re-using in Figure 6 for three arm geometry with various combinations of joint and bone templates defined in Figures 4 and 5. In the accompanying video, we further show how the same templates yield similar deformation styles on the arms and legs of two full-body models (a human and an alien) with very different shapes.

Unlike previous example-based approaches, the templates in our method have a simple structure, making it easy to design the examples that serve to define the deformation (see Figures 4 and 5). As a result, our approach can offer a multitude of easily-creatable deformations styles. Besides common skin behaviors such as muscle bulging, Figure 12 and the accompanying video demonstrate a set of exaggerated, non-physically realistic skin behaviors, such as upward muscle bulging, sliding and twisting. These templates achieving these behaviors are as simple as in those in Figure 5 while differing in the second key poses.

The use of cage-based deformations achieves interactive speed desirable for real-time animation. At run time, the animation is dominated by computing the deformed mesh vertex locations from the deformed cage (driven by the skeleton), which involves no more than a linear sum of cage vertices weighted by pre-computed PMVC weights [Lipman et al. 2007]. We recorded roughly 0.03 second for deforming a mesh with 15K triangles and a cage with 200 triangles. The pre-computation consists of two stages: fitting the cage to the model, which takes place only once and typically no more than five minutes including user interaction, and computing the PMVC deformation weights at each mesh vertex with respect to the fitted cage vertices. Note that the per-vertex weights have to be re-computed when a new template is selected by the user at a joint or bone, which results in a new cage. To facilitate fast switching between templates in the interactive tool, we pre-compute the PMVC weights for all compatible templates at a selected joint or bone using local updates. This computation takes no more than a few seconds, after which compatible templates can be switched instantly (using the Tab key in our tool).

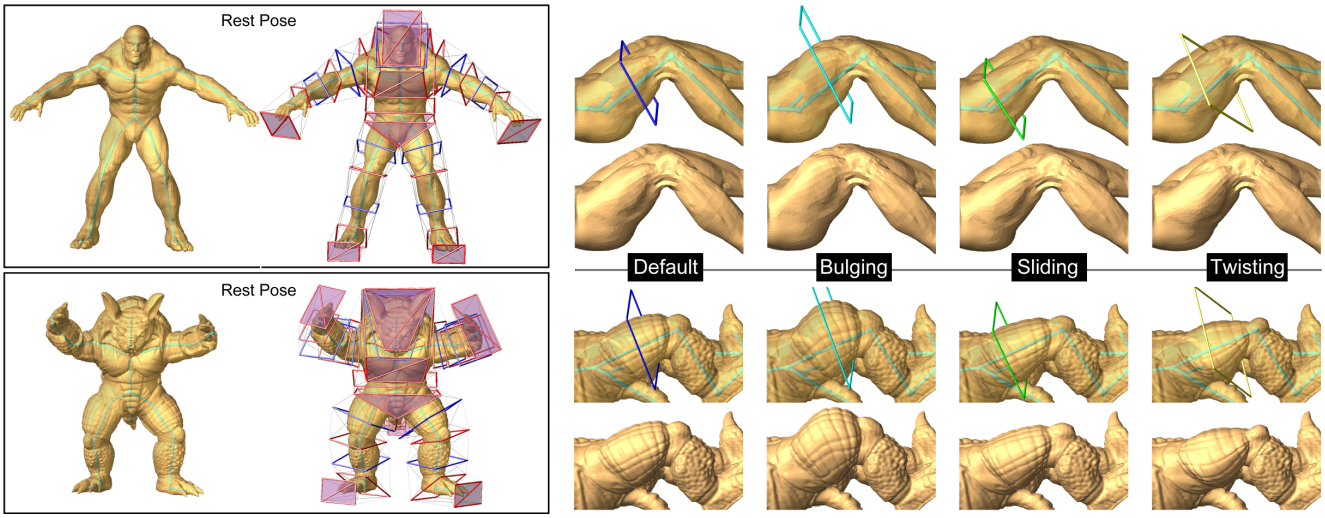

Figure 12: *Skinning templates applied to the Monster (top) and Armadillo (bottom) model. Left: the rest pose geometry and fitted cages. Right: the deformed leg in the bending pose when different bone templates are used, exhibiting upward muscle bulging, sliding, and twisting.*

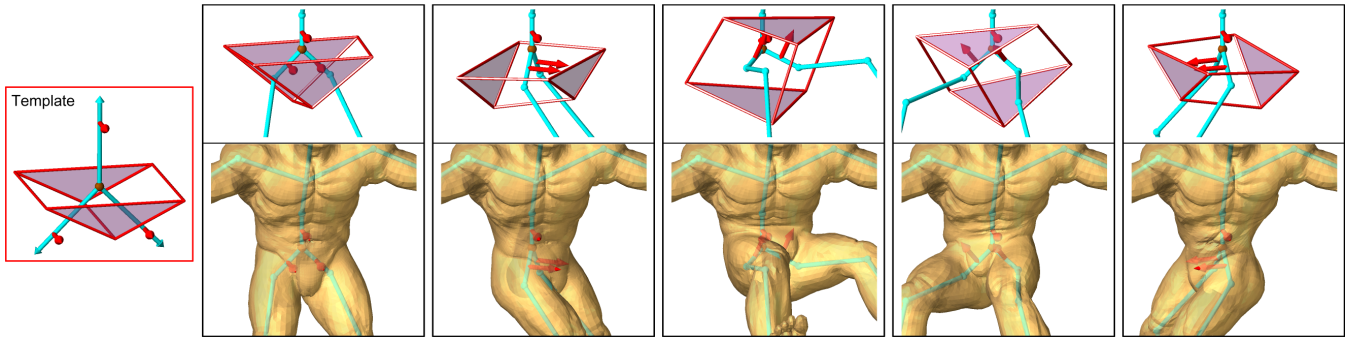

Figure 13: *Left: a waist template. Right: the deformation of the template vertices (and the skin) as the skeleton pose changes (top).*

7 Conclusions

Skinning templates provide a novel abstraction of skinning behavior. While model-specific skinning techniques provide detailed control for the consumate artist, skinning templates provide fast, generic skinning solutions that target common desired skinning behaviors. This class of technique will become increasingly important with the rapid adoption of user-created content as a key feature of many games and simulations. A key challenge addressed in our work is that the templates need to be generic in nature, but still apply to highly variable geometry. Recent developments in cage-based deformations are also a key component of the proposed solutions.

7.1 Discussions

Limited by the detail found in the templates pieces, cage-based skinning templates cannot by themselves create detailed skin behaviors desired in high-end applications. However, rich-and-detailed skinning effects require rich-and-detailed user input, such as the effort of developing example poses for example-based skinning techniques. Skinning templates are complementary to rich-detail skinning techniques in two ways. First, skinning templates allow for fast exploration of alternative skinning styles, providing medium-detail skinning effects at interactive speed that are often sufficient for applications where less complicated models are used and where real-time animation is desired (e.g., in games). Second, if additional detail is desired beyond what templates can provide, the skinning-template results can be used to bake out example poses that are already quite good and can then be refined and used as input for example based skinning techniques.

Our method is built upon the recent development in cage-based deformations [Ju et al. 2005; Lipman et al. 2007; Joshi et al. 2007], which have their own limitations. For example, each skin vertex is globally affected by all cage vertices. In addition, most techniques only work on cages with triangle faces. Developing new cage-based deformation techniques that address these and other limitations is still an active research subject [Lipman et al. 2008]. These new techniques can be easily integrated into our method, which is independent of the specific type of cage-based deformation.

We are investigating in a number of directions to improve and extend our current method. We are looking into developing robust optimization techniques that would reduce the need for user intervention during initial cage fitting and would allow automatic fitting on an army of characters. With automated fitting, skinning templates can be integrated into an automatic rigging tool to allow for an increase in the skinning quality of automatically-rigged charac-

ters. We further wish to investigate templates that implement dynamic effects and that may be 'model aware' in a variety of other ways. This also points towards using semantics-informed models and abstractions in tools for constructing characters.

8 Acknowledgement

We would like to thank Johannes Kopf for providing the PMVC code. The models used in the paper and accompanying video are provided by 3DM3.COM and Turbo Squid (except the Monster and Armadillo). The work is supported in part by NSF grants IIS-0705538 and CCF-0702662.

References

ALLEN, B., CURLESS, B., AND POPOVIĆ, Z. 2002. Articulated body deformation from range scan data. *ACM Trans. Graphics (Proc. SIGGRAPH) 3*, 21, 612–619.

ANGELIDIS, A., AND SINGH, K. 2007. Kinodynamic skinning using volume-preserving deformations. In *Proc. Symposium on Computer Animation*, 147–156.

BARAN, I., AND POPOVIĆ, J. 2007. Automatic rigging and animation of 3d characters. *ACM Trans. Graphics (Proc. SIGGRAPH) 26*, 3, Article 72.

BOTSCH, M., AND SORKINE, O. 2008. On linear variational surface deformation methods. *IEEE Trans. on Visualization and Computer Graphics* (Jan/Feb).

BURTNYK, N., AND WEIN, M. 1976. Interactive skeleton techniquese for enhancing motion dynamics in keyframe animation. *Communications of the ACM 19*.

BUSS, S. R., AND FILLMORE, J. P. 2001. Spherical averages and applications to spherical splines and interpolation. *ACM Trans. Graph. 20*, 2, 95–126.

CAPELL, S., BURKHART, M., CURLESS, B., DUCHAMP, T., AND POPOVIĆ, Z. 2007. Physically based rigging for deformable characters. *Graphical Models 69*, 1.

CHADWICK, J., HAUMANN, D., AND PARENT, R. 1989. Layered construction for deformable animated characters. *ACM SIGGRAPH Computer Graphics 23*, 3, 243–252.

FORSTMANN, S., OHYA, J., KROHN-GRIMBERGHE, A., AND MCDOUGALL, R. 2007. Deformation styles for spline-based skeletal animation. In *Symp. on Computer Animation*, 141–150.

FUCHS, H., KEDEM, Z. M., AND USELTON, S. P. 1977. Optimal surface reconstruction from planar contours. *Commun. ACM 20*, 10, 693–702.

GUO, Z., AND WONG, K. C. 2004. Neuroenveloping: A transferable character skin deformation technique. In *Proc. Pacific Graphics*, 77–86.

JOSHI, P., MEYER, M., DEROSE, T., GREEN, B., AND SANOCKI, T. 2007. Harmonic coordinates for character articulation. *ACM Trans. Graphics (SIGGRAPH) 26*, 3, #71.

JU, T., SCHAEFER, S., AND WARREN, J. 2005. Mean value coordinates for closed triangular meshes. *ACM Trans. Graphics (Proc. SIGGRAPH) 24*, 3, 261–566.

KAVAN, L., AND ŽÁRA, J. 2005. Spherical blend skinning: a real-time deformation of articulated models. In *Symp. on Interactive 3D graphics and games*, 9–16.

KAVAN, L., COLLINS, S., ZARA, J., AND O'SULLIVAN, C. 2008. Geometric skinning with approximate dual quaternion blending. ACM Press, New York, NY, USA, vol. 27.

KRY, P., JAMES, D., AND PAI, D. 2002. EigenSkin: real time large deformation character skinning in hardware. *Symp. on Computer Animation*, 153–159.

LEWIS, J., CORDNER, M., AND FONG, N. 2000. Pose space deformation: A unified approach to shape interpolation and skeleton-driven deformation. *Proc. ACM SIGGRAPH*, 165–172.

LIPMAN, Y., KOPF, J., COHEN-OR, D., AND LEVIN, D. 2007. Gpu-assisted positive mean value coordinates for mesh deformations. In *Symp. on Geometry processing*, 117–123.

LIPMAN, Y., LEVIN, D., AND COHEN-OR, D. 2008. Green coordinates. *ACM Trans. Graph. 27*, 3, 1–10.

MAGNENAT-THALMANN, N., LAPERREIRE, R., AND THALMANN, D. 1988. Joint dependent local deformations for hand animation and object grasping. In *Graphics Interface*, 26–33.

MAYA, 2007. Autodesk inc.

MERRY, B., MARAIS, P., AND GAIN, J. 2006. Animation space: A truly linear framework for character animation. *ACM Trans. Graphics 25*, 4, 1400–1423.

MOCCOZET, L., AND MAGNENAT-THALMANN, N. 1997. Dirichlet free-form deformation and their application to hand simulation. In *Proc. Computer Animation*.

MOHR, A., AND GLEICHER, M. 2003. Building efficient, accurate character skins from examples. *ACM Trans. Graphics (Proc. SIGGRAPH) 22*, 3, 562–568.

PRATSCHER, M., COLEMAN, P., LASZLO, J., AND SINGH, K. 2005. Anatomic rigging of characters from the outside-in. In *Symposium on Computer Animation*, 329–338.

SEDERBERG, T., AND PARRY, S. 1986. Free-form deformation of solid geometric models. *ACM SIGGRAPH Computer Graphics 20*, 4, 151–160.

SINGH, K., AND FIUME, E. 1998. Wires: A geometric deformation technique. In *Proc. ACM SIGGRAPH*, 405–414.

SINGH, K., AND KOKKEVIS, E. 2000. Skinning characters using surface oriented free-form deformations. In *Proceedings of Graphics Interface*, 35–42.

SLOAN, P., ROSE III, C., AND COHEN, M. 2001. Shape by example. *Symp. on Interactive 3D graphics*, 135–143.

SUMNER, R., ZWICKER, M., GOTSMAN, C., AND POPOVIĆ, J. 2005. Mesh-based inverse kinematics. *ACM Trans. Graphics (Proc. SIGGRAPH) 24*, 3, 488–495.

WANG, X. C., AND PHILLIPS, C. 2002. Multi-weight enveloping: Least-squares approximation techniques for skin animation. In *Symposium on Computer Animation*, 129–138.

WANG, R., PULLI, K., AND POPOVIĆ, J. 2007. Real-time enveloping with rotational regression. *ACM Trans. Graphics (Proc. SIGGRAPH) 26*, 3, Article 73.

WILHELMS, J., AND GELDER, A. V. 1997. Anatomically based modeling. In *Proc. ACM SIGGRAPH*, 173–180.