Local volume preservation for skinned characters

Damien Rohmer, Stefanie Hahmann, Marie-Paule Cani

Local Volume Preservation for Skinned Characters

Damien Rohmer[†1,2], Stefanie Hahmann[‡1], and Marie-Paule Cani[§1,2]

[1]LJK Lab, Grenoble Universités, France
[2]INRIA, France

Abstract

Generating plausible deformations of a character skin within the standard production pipeline is a challenge. This paper presents a volume preservation method dedicated to skinned characters. As usual, the character is defined by a skin mesh at some rest pose and an animation skeleton. At each animation step, skin deformations are first computed using standard SSD. Our method corrects the result using a set of local deformations which model the fold-over-free, constant volume behavior of soft tissues. This is done geometrically, without the need of any physically-based simulation. To make the method easily applicable, we also provide automatic ways to extract the local regions where volume is to be preserved and to compute adequate skinning weights, both based on the character's morphology.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism

1. Introduction

Fast and realistic deformation of organic shapes is of major interest for the Computer Graphics industry. Such tools are needed for characters animations in movies, video games and other virtual environments. To ease the artists task, skin deformations should ideally be generated at interactive rates. They should also capture some natural behavior having a strong visual impact on the realism of deformation such as self-penetration avoidance and the generation of bulges and creases due to the deformation of muscles and fatty tissues.

Smooth skinning, also called Skeletal Subspace Deformation (SSD), is well suited to character animation thanks to its intuitive way of using the underlying skeleton motion and to its computational efficiency. In contrast with physically-based skinning techniques, SSD requires no specialized knowledge from the user such as the ability to tune simulation parameters. Moreover, computations are independent from one frame to the next, enabling the rendering of an animation sequence to be distributed among different processors or computers. However, this method suffers from

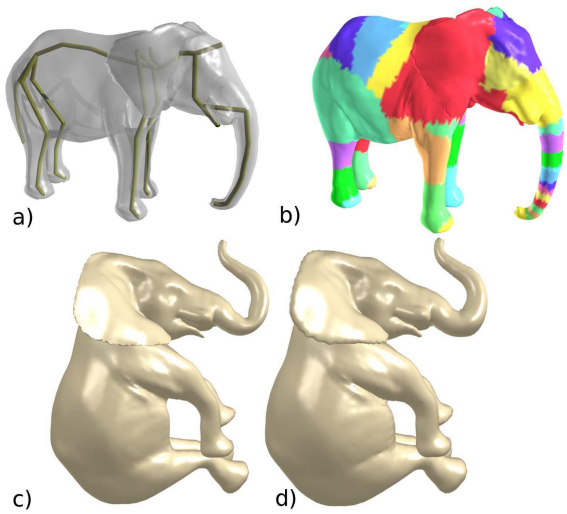

Figure 1: *Illustration of our method in a complex case. a) Input data: skinned mesh and skeleton. b) Automatic segmentation. c) Standard SSD. d) Our method where the volume is locally preserved (see belly and trunk).*

† Damien.Rohmer@imag.fr
‡ Stefanie.Hahmann@imag.fr
§ Marie-Paule.Cani@inrialpes.fr

many artifacts which may impair realism, such as an obvious loss of volume and a local fold-over when an articulation bends. Volume preservation is well known in Computer Graphics for making deformations look more natural. In particular, this property is essential for organic shapes (as in Figure 1), whose soft tissues are mostly made of water. Achieving a plausible result within the standard production pipeline - thus in a geometric way, with no further knowledge than the structure of the skeleton and its position with respect to the skin surface at rest is a challenge. The goal of this work is to enhance the simple and widely used SSD technique with adapted volume preserving, fold-over free constraints, yielding natural looking animations without giving up the standard pipeline.

Our main contributions, detailed in Sections 3 to 5, are: a mathematical framework for restoring the volume of a closed mesh after any deformation while using maps to control the regions where the corrections mainly take place; a method for generating adapted correction maps, which prevent the local losses of volume due to SSD artifacts; and an extension to full characters, where the volume preservation method is applied to a set of local sub-volumes that are automatically extracted based on morphology. Although our method can be combined with user-defined skinning weights, we also provide an automatic way to compute them. As results show, our method corrects the main artifacts of SSD and achieves appealing visual results, such as plausible local bulges when neighboring skin parts are in contact due to extreme joint postures.

2. Related Work

Let us first review the recent advances in skinning techniques which inspired our work.

Smooth skinning (or SSD), which links a mesh to an underlying animation skeleton via a set of influence weights, is widely used for deforming organic shapes. In the production process, many computer artists specialized in character setup have acquired impressive skills for painting the weights associated with each skeleton element over the character mesh. To make the resulting deformations more plausible, they sometimes spend hours binding internal collision volumes to the skeleton, in combination with SSD. This insures that the local volume of soft tissues will not shrink too much in specific regions. In some cases, the volumes are animated depending on the skeleton posture, to restore bulging effects.

While the original SSD method was not published in literature, many papers addressed its artifacts in the last few years, aiming to alleviate the artist's task. Most of these papers address the famous issue of volume shrinkage, also known as the "collapsing elbow" effect [LCF00], which occurs for large bending angles at joints. The improvements to the original method can be classified into two categories:

Firstly, *example-based techniques* interpolates between a set of deformed mesh poses [LCF00] or learn from them. An optimization process fits some parameters such as vector correction [KJP02], matrices of influences [WP02], or extra joints within in the skeleton [MG03]. This enables the skinning process to recreate visual effects such as bulges, if they were present in the training meshes. For instance Wang et al. [WPP07] propose a linear regression between triangle deformations of the mesh and the joint angles in the learning step to catch such effects. Some recent methods [JT05, SY07, dATTS08] automatically build the skeleton, joint motion and skinning weights from the deformed mesh at some typical poses or from a mesh animation. The major drawback of these methods is the need for training poses. When they cannot be captured on a real actor, creating these poses requires either tremendous effort from an artist, or to run a complex physical simulation on a volumetric version of the skin mesh. In both cases, the mesh and its associated skeleton at rest are not sufficient, and further human intervention is required.

The second family of approaches suggested improvements of the skinning calculation itself. This includes for instance the use of a medial axis [Blo02], rigidity preservation [SZT*07], non-linear interpolation of blending matrices [Ale02] or the use of quaternions [KZ05]. Compared with standard SSD, these methods can handle larger deformations and reduce the usual artifacts while calculation time remains acceptable for interactive use. However, volume preservation is not treated as a goal. It is worth noticing that these methods are using the same parameters as classical skinning. Therefore, the volume correction method we are presenting in this paper can be applied to any of them.

More recently, several impressive works promoted a better representation of surfaces. Laplacian coordinates are for example very well suited to mesh deformation [SCOLA04]. Application of such representations to SSD has been explored by Zhou et al. [ZHS*05]. The method ensures that the volume is almost preserved thanks to a volumetric tetrahedrization of the mesh. Exact global volume conservation is also performed by Huang et al. [HSL*06] at the expense of a large sparse matrix to be inverted at each deformation step. Still, volume conservation is performed globally, or requires the use of human interaction to specify some closed part where it has to be conserved. Therefore no local bulging effect can be observed. In contrast, we promote the idea that for being adapted to organic shapes, a volume preservation method should act locally. Moreover, we think that rather than asking a skilled user to manually specify regions of influence, these regions can be automatically computed from the characters morphology.

Lastly, some recent work was inspired from constant volume space deformation. Using von Funck's [FTS06] divergence-free vector field, Angelidis and Singh [AS07] generated a SSD-like deformation which is both fold-over

free and volume preserving. The method is well suited to create bulges around deformed joints. However, its major limitation is the need for streamline integration at every vertex, which may affect speed for larges meshes. Contrary to this approach, we first rely on standard SSD before applying a fast, local post-correction step which prevents fold-over and restores volume.

3. Locally controlled Volume Preservation

Preserving volume in character animation raises specific issues: this preservation should be local, being due to the specific nature of muscles and to the incompressibility of fatty tissues, in which deformations never propagate very far.
This section introduces a general method for locally controlling the amount of volume-restoring deformation over a mesh. We rely on a correction map defined over the mesh to explicitly control the regions where the correction is to be applied.

3.1. Computing volume

Let $S = (\mathcal{V}, \mathcal{F})$ be a closed triangular mesh, surrounding an interior domain. V denotes the set of vertices, and F the set of oriented triangular faces. Let $N = |\mathcal{V}|$ be the number of vertices of S. Its embedding $\mathbf{p}(S)$ maps any vertex k of S onto a position $\mathbf{p}_k \in \mathbb{R}^3$, $k = 1,\ldots,N$. It can be shown [LK84, HML99, HJCW06] that the exact expression of this oriented interior volume is given by the following trilinear functional

$$V = \frac{1}{6} \sum_{(i,j,k)\in F} (z_i + z_j + z_k) \begin{vmatrix} x_j - x_i & x_k - x_i \\ y_j - y_i & y_k - y_i \end{vmatrix}, \quad (1)$$

where $(x_\alpha, y_\alpha, z_\alpha)$ are the coordinates of vertex $\mathbf{p}_\alpha$. This formula can be interpreted geometrically (see Figure 2 a)).

Let us denote in the following: the mesh vertices after deformation by the $(3N)$-vector $\mathbf{p}$ with $\mathbf{p}^T = (\mathbf{p}_1^T, \ldots, \mathbf{p}_N^T) \in \mathbb{R}^{3N}$ and the vertices of the rest pose by $\bar{\mathbf{p}}$. Computing the volumes $V(\bar{\mathbf{p}})$ and $V(\mathbf{p})$ for the rest pose and the deformed shape (after application of the SSD method) respectively, the *global volume variation* is given by $\Delta V = V(\bar{\mathbf{p}}) - V(\mathbf{p})$.

Let us now assume that the deformed shape has been pointwise corrected by $\mathbf{u}^T = (\mathbf{u}_1^T, \ldots, \mathbf{u}_N^T) \in \mathbb{R}^{3N}$. In the cases where $\|\mathbf{u}\|$ is small enough, the non-linear volume functional can be linearized by using the following first order approximation:

$$V(\mathbf{p} + \mathbf{u}) \simeq V(\mathbf{p}) + < \mathbf{u}, \nabla V(\mathbf{p}) >_{\mathbb{R}^{3N}} , \quad (2)$$

where $<,>_{\mathbb{R}^{3N}}$ denotes the dot product in $\mathbb{R}^{3N}$

3.2. Efficiently correcting volume

In order to make the correction dependant on the local shape of the surface, we constrain the vector $\mathbf{u}$ to be normal to the deformed surface. Therefore, we are looking for a set of scalar values $\rho = (\rho_1, \ldots, \rho_N)$ such that $\mathbf{u} = \rho\mathbf{n}$ describes the volume crrecting deformation, where $\mathbf{n}^T = (\mathbf{n}_1^T, \ldots, \mathbf{n}_N^T)$ is the set of unitary normal vectors and $(\rho\mathbf{n})^T = (\rho_1\mathbf{n}_1^T, \cdots, \rho_N\mathbf{n}_N^T)$.

The volume correction is expressed as the solution of a constrained minimization problem:

$$\begin{cases} \min \|\rho\|^2_{\mathbb{R}^N} \\ \text{subject to } V(\mathbf{p} + \rho\mathbf{n}) = V(\bar{\mathbf{p}}) \, . \end{cases} \quad (3)$$

Practicaly, we use the linearized expression (2) of the constraint. This is thus expressed as $< \rho\mathbf{n}, \nabla V(\mathbf{p}) >_{\mathbb{R}^{3N}} = \Delta V$. We provide in (Section 6) an error measurement of this approximation.

To increase efficiency, we derive a closed form solution for the volume correction coefficients ρ_k as follows: We use a Lagrange multiplier λ to solve equation 3. The minimization problem with the linearized volume constraint is thus equivalent to solve the unconstrained min-max problem with the energy functional $E(\rho, \lambda) = \|\rho\|^2_{\mathbb{R}^N} - \lambda(< \rho\mathbf{n}, \nabla V(\mathbf{p}) >_{\mathbb{R}^{3N}} - \Delta V)$. Solving this linear system for ρ_k at a vertex k leads to the closed form analytical solution given by:

$$\rho_k = \Delta V \frac{< \mathbf{n}_k, \nabla V(\mathbf{p}_k) >}{\sum_{j=1}^{N} < \mathbf{n}_j, \nabla V(\mathbf{p}_k) >^2} , \quad (4)$$

Practicaly, we compute the volume variation ΔV and gradient ∇V before adding the volume corrections to each vertex. Eq. (4) can then be evaluated with a dot product evaluation and a single division per vertex resulting in interactive rates.

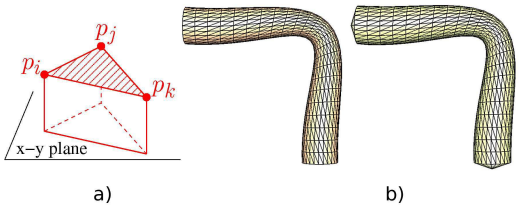

Figure 2: *a) The volume of a closed triangular mesh is computed by summing up the (signed) volumes of the prisms spanned by the mesh triangles and their projections onto the (xy) plane. b) Left: A constant radius cylinder has been deformed. Right: Naïve volume-restoring generates a general inflation (most visible at the extremities). The center region stays artificially shrunken.*

The result of the global volume correction presented above is shown in Figure 2 b). As expected, applying a uniform volume correction based on (3) only results in general inflation of the mesh. In case of skinned characters, this would not improve the shape after a loss of volume near a joint. A mechanism, presented next, is therefore needed to control the way volume-restoring displacement is distributed over the mesh during the process.

3.3. Taking a correction map into account

In order to get local volume corrections, let us introduce a local correction map γ. These γ-values are N positive weights $(\gamma_i)_{i\in[\![1,N]\!]}$ defined to localize in a more accurate way the volume correction.
Used in (3), the norm is now defined as: $\|\rho\|^2 := \sum_{k=1}^{N} \frac{\rho_k^2}{\gamma_k}$. The resulting modification of the closed form solution is:

$$\rho_k = \Delta V \frac{\gamma_k < \mathbf{n}_k, \nabla V(\mathbf{p}_k) >}{\sum_{j=1}^{N} \gamma_j < \mathbf{n}_j, \nabla V(\mathbf{p}_k) >^2} , \qquad (5)$$

The use of local maps does not affect the efficiency of computations, γ_k being a predefined constant value. Note that, in order to preserve the smoothness of the shape after volume correction, the variations in the correction map have to be sufficiently smooth.

The next section presents two automatic methods to compute adapted local correction maps enhancing the realism of skinned characters.

4. Generating Volume Correction Maps for SSD

Although the correction map γ controlling the distribution of volume-restoring displacements could directly be painted onto the surface by user interaction, providing an automatic method for generating it saves time and effort. This section explores method for computing correction maps adapted to smooth skinning. We build on the a priori knowledge on the regions where the loss of volume occurs in SSD and thus should be corrected, to generate these maps. Let us briefly review the SSD method.

4.1. Smooth skinning (SSD)

Let's consider a triangulated surface $\mathcal{S}$ of N vertices defining the rest pose of the shape and an associated skeleton composed of a hierarchy of local frames. A *bone* is the segment linking two consecutive joints. The SSD algorithm calculates the deformed position of a vertex $\mathbf{p}_k \in \mathbb{R}^3$ of $\mathcal{S}$. $k \in [\![1,N]\!]$ as a linear combination of the frames position and orientation

$$\mathbf{p}_k = \sum_{i=1}^{b} \omega_{ki} \, \mathrm{T}_i \overline{\mathrm{T}}_i^{-1} \, \overline{\mathbf{p}}_k , \qquad (6)$$

where b is the number of frames. T_i is the transformation matrix of the frame i, and $\overline{\mathrm{T}}_i$ is the transformation matrix of the same frame at rest. The $(\omega_{ki})_{i=[\![1,b]\!]}$ are called "skinning weights". They satisfy $\omega_{ki} \geq 0$ and $\sum_{i=1}^{b} \omega_{ki}$.

As a basic example, a cylinder is used all over this section to illustrate the behavior of standard SSD and the results we get using correction maps. The cylinder has a skeleton composed of two bones connected by one joint. A loss of volume of about 11% can be measured for a 90° bent when using classical SSD (see Figure 2 (left)).

4.2. First correction map: Rubber effect

To compensate the local loss of volume due to the collapsing elbow effect, the correction should be concentrated near joints. A first idea is to compute our correction map from the skinning weights ω_{ki}: mesh parts corresponding to joints are regions where no skinning weight value is predominant over the others. On the opposite, regions influenced by a single bone will undergo very little deformation so almost no volume correction should be applied (see Figure 3 (a)). The gamma correction map can therefore be defined at a vertex p_k as $\gamma_k = (1 - \max_i \omega_{ki})^{\alpha}$, where α is a tilting parameter.

Figure 3 (b) shows the result of this approach: the deformation is now located near the joint. However, due to the radial nature of our volume corrections, it results on a plastic-looking deformation, as if the object was made of rubber. Although this visual effect can be useful to animate such a material, these results are not acceptable for organic shapes, which are the main target of skinning techniques.

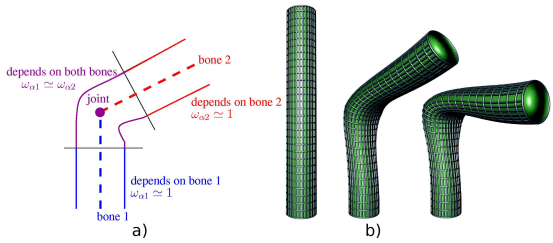

Figure 3: *a) Representation of the dependencies of vertices to the two bones in the example of the deformed cylinder. b) Example of an SSD deformed cylinder with correction map based on skinning weights. Note the rubber effect due to radial deformations uniformly applied near joints.*

4.3. Correction map for an organic behavior

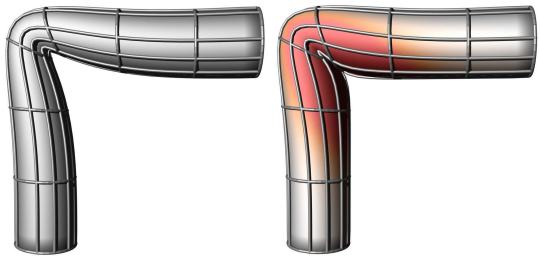

Figure 4: *Deformed cylinder with non-centered skeleton. Left: classical SSD. Right: Cylinder after volume correction. The red color encodes the correction intensity.*

In the case of real organic shapes, flesh compression due to the articulation of the skeleton results in a bulging behavior for the fatty tissues located between joints, with the effect

being greater near them (look at your hand with your fingers bent). Moreover, skeletons of organic shapes are usually not centered, as underlined by Aujay et al. [AHLD07]. One can note that these natural bulges occur in the regions which are the further away from the skeleton, while regions located just over bones deform almost rigidly.

Let us consider from now on a non-centered skeleton in the cylinder example. We are looking for a correction map which corrects volume by deforming regions far away from the skeleton in priority, and thus will not act uniformly all around a joint. This new correction map can be defined for instance by

$$\gamma_k = (1 - \max_i \omega_{ki})^\alpha \left(\min_s d_{ks}\right)^\beta , \qquad (7)$$

where d_{ks} is the distance of the vertex k to the bone s. α and β are tilting parameters.
Figure 4 depicts a result of this method. The result clearly shows the bulging effect due to the dependence of the correction map on the distance from the skeleton. The rubber effect is eliminated while the location of the correction remains close to the joint.

5. Extension to Full Characters

The method we have just defined captures the way organic shapes deform (the deformed cylinder in Figure 4 now behaves like a finger), but is not yet applicable to a full character: If we keep a global computation of volume variations, two opposite local variations could annihilate themselves and therefore remain uncorrected; moreover, it would be impossible to apply the correction locally, in the right region, since the volume computation would only output a single variation value.

We therefore propose an automatic way to segment an organic shape into a set of regions corresponding to the main muscle and fatty tissue areas, in which volume will be computed and locally corrected. We also use the regions to prevent skin fold-over between neighboring parts of the mesh. Before explaining how these regions and their volume are computed, let us give some attention to the way adequate skinning weights can be set in case they were not already defined by an artist.

5.1. Skinning weights

Very often, the only input we have for a character is a skin mesh. An automatic method such as [AHLD07] can then be used to extract an adapted animation skeleton. In such a case, we also use an automatic method for generating skinning weights.

Although any of the previous methods, such as those based on geodesic distance [MTG03] or on heat equation [CBDP05] could be used for generating skinning weights, we found out that a very simple geometric consideration, coherent with the observations on organic shapes we already made, gives sufficient results. Here is its outline:
First, skinning weights are automatically defined using the minimum distance d_{ks} from vertex k to the segment defining the bone s rather than distance to joint as usual [†]. The skinning weights are expressed by $\omega_{ki} = \left(\frac{l_i}{d_i}\right)^\alpha$, where l_i is the bone length and $\alpha > 0$.
Since this simple computation depends on Euclidean distance rather than on structure (so that the skin over one finger would be influenced by the skeleton of the neighboring fingers), we set an influence weight value back to zero when the ray between the vertex and the bone, along which the closest distance was computed, crosses the mesh. (see figure 5). In

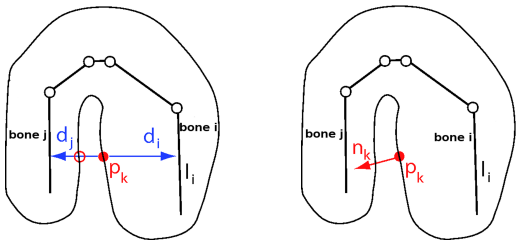

Figure 5: *When computing skinning weights, we avoid influence of skin parts which are geometrically close, but which belong to separated parts. Left: Ray-Tracing and intersection detection is one possibility. Right: Test of normal direction compared to the ray is another possibility.*

order to avoid ray-tracing and intersection detection, we use a simple approximation by comparing the dot product between this ray and the surface normal. If the angle between these two directions is too small, the ray escapes the body domain [‡]. A heuristic is used for parts that cannot be reached from bones along rays which stay inside the body (such as the horn of a goat for which all weights would be zero): in this case, we consider this part of the mesh as a rigid part, and thus set to one the skinning weight associated with the closest bone.
In a last step, the skinning weights are normalized to one.

Note that our method results in almost rigid deformation of skin parts which are close to the skeleton, such as the back of the elephant, since the influence of the closest bone will

[†] Which is done because the local frames of the skeleton hierarchy are typically positioned at joints, although they define the frame of a given bone.

[‡] Note that it can happen that ray-shooting does not give the right association due to very complex geometry such as a skin which would be heavily wrinkled at rest. In such cases, the animator would have to paint more appropriate skinning weights, or the later would have to be computed on a simplified version of the skin mesh. As it is, our method gave satisfactory results in all the examples we tested.

be highly dominant. In contrast, parts which are evenly far from many bones, such as the belly, have evenly distributed influences: more volume will be lost in those parts using the SSD method, but this is also where the volume-correcting displacements will mostly take place.

5.2. Computing local volumes

Once skinning weights have been defined, the character is segmented into regions where the volume is to be locally preserved. We use these as a Voronoi-like definition of regions: they are set to be the parts of the mesh for which a given skeleton element has the maximal skinning weight as seen in Figure 1 (b) and 6 (a). It is required that the volume difference in one part does not influence the correction in other parts, therefore the different segments do not overlap.

For each region, the local variation of volume ΔV_s is now estimated. As the segmented regions are not closed (the partitions being just defined over the mesh), the variation of volume is only estimated. Regions of the deformed shape and rest pose are compared to their image after being transformed into a local frame given by the inverse of the transformation matrix T_i (see (6)) of the bone of main influence. Then the local signed volume difference of every pair of triangles $(\mathbf{p}_i, \mathbf{p}_j, \mathbf{p}_k)$ and $(\overline{\mathbf{p}}_i, \overline{\mathbf{p}}_j, \overline{\mathbf{p}}_k)$ between rest and deformed pose is calculated by considering the prism spanned by these two triangles as shown in Figure 6 (b). Since quadrilateral faces of the prism are generally not planar, we proceed following [BK03], by splitting them into 4 subtriangles at their centroid. This guarantees that neighboring faces share a common edge. Finally, the small signed volume variations are summed up on the entire region.

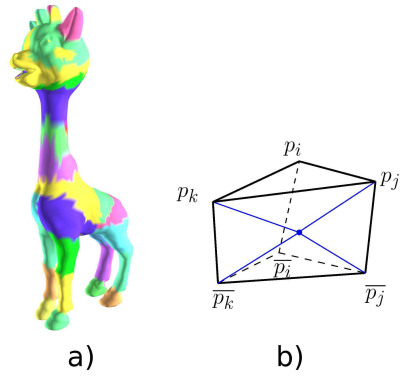

Figure 6: *a) Example of segmentation. The different colors separate regions where local volumes are computed. b) Prism used for the local volume computation.*

5.3. Correcting local volumes

Once a local volume variation is computed, the correction has to be localized on the corresponding part of the skin mesh. Let's call γ^s the correction map associated to the region s. In order to avoid discontinuities, the γ^s values should smoothly decrease to zero in the neighborhood of the boundaries. Therefore the non-smoothness of the boundary of a given region does not imply an irregular correction. Eq. (7) is modified to take into account the distance to the boundary (characterized by $\omega_{ki} \leq \frac{1}{2}$)

$$\gamma_k^s = \left(2(\max(\max_i \omega_{ki}, 0) - \frac{1}{2})\right)^{\alpha} \left(\min_s d_{ks}\right)^{\beta} . \quad (8)$$

For each segment s the final correction magnitude ρ_k of its vertices $k \in s$ is computed using (5) where ΔV is remplaced by ΔV_s.
In addition to improving results, the local volume preservation method speeds up computations: If the volume variation $\Delta V_s = 0$, no other calculation is needed for the sections s. Therefore, if only some parts of a complex shape are animated, the correction process automatically focusses on those parts.

5.4. Preventing Fold-overs

The SSD method does not garantee that the deformed mesh will not auto-intersect when an articulation bends a lot. Further benefit of our automatic segmentation method is to be able to automatically detect and prevent such fold-overs: instead of auto-intersections, the skin vertices of a region are constrained to stay on a contact surface with their neighboring region, resulting in more local compression, which will be automatically corrected by an extra volume preservation step. More precisely, the idea is to recompute the bone of main influence for a vertex at each deformation step using the weight computation method in section 5.1: If the bone of main influence is not, at the deformed pose, the one associated to this vertex during segmentation, a fold over is detected. This vertex k_a is then translated back to the contact surface. The direction of translation $\mathbf{t}_{k_a s_a}$ for the vertex k_a is given by $\overrightarrow{\mathbf{p}_{k_a} I_{s_a k_a}}$, where s_a is the original segment linked to k_a and $I_{s_a k_a}$ is the closest point from $\mathbf{p}_{k_a}$ on the bone s_a. The displacement is iterated such that

$$\mathbf{p}_{k_a}^{i+1} = \mathbf{p}_{k_a}^{i} + \Delta s \, \mathbf{t}_{k_a s_a}^{i} ,$$

until k_a reaches the border of its original region (see Figure 7). In order to speed up calculation, this computation is only done at vertices inside a bending joint.

Once this step has resulted in at least one translated vertex, another volume preservation step is applied to this region (see Figure 10).

6. Results and Discussion

Our method has been tested on animal models such as an elephant and a giraffe toy. In the giraffe case, the mesh, the skeleton, skinning weights and motion examples were defined by an artist. For the elephant example, the

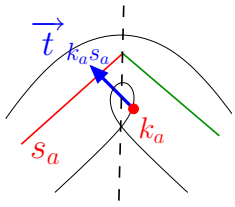

Figure 7: *Method to avoid self-intersection. Vertex k_a is detected in the region linked to the green bone while its original region was the red one. k_a is thus translated along $\mathbf{t}_{k_a s_a}$ until it reaches border of its associated region.*

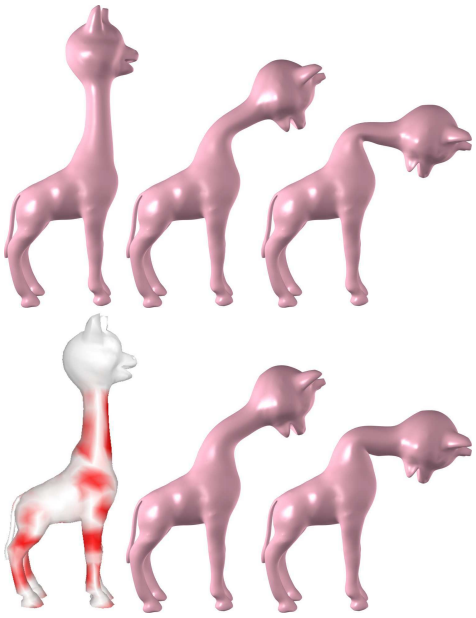

Figure 8: *Results for the organic effect. First line: Classical SSD. Second line: Our organic effect, the left picture illustrates the γ intensity in red (note that gamma values are shown for each local region).*

skeleton was extracted from the mesh using Aujay's algorithm [AHLD07], and our method for computing skinning weights was used.

Figure 8 and 9 compare the rubber and the organic effects on the giraffe model for an unrealistic bent of the neck. As expected, the giraffe on Figure 9 looks-like the real rubber giraffe we used as a model. Our experiments, also shown in the joint video, included bending the elephant into extreme poses (see Figure 1). Improvement can be noticed in the belly region due to its large compression, but also in trunk and legs parts.

Finally, the fold-over free effect is illustrated by the example of the hand in Figure. 10. The finger is bent until self-intersection occurs when no prevention over fold-over is applied.

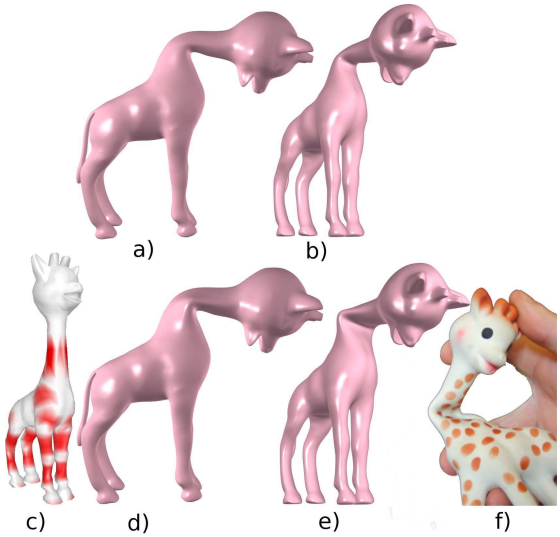

Figure 9: *Illustration of the rubber effect. a) & b) Deformation with classical SSD. c) γ-map for the rubber effect. d) & e) Our rubbery effect. f) Real rubber giraffe toy with bent neck.*

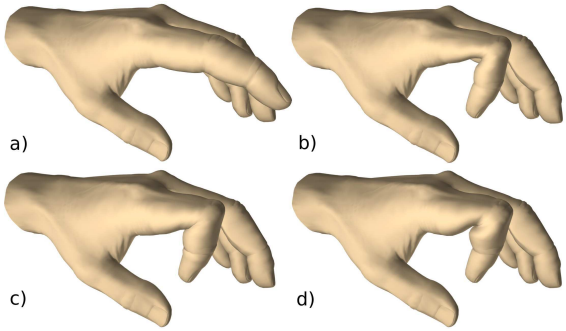

Figure 10: *Illustration of the fold-over free application. a) Original mesh. b) Bended finger with classical skinning. c) Simple volume restauration (self-intersection at the junction) d) Fold-over free effect, self-intersections are prevented and the bulge effect is increased.*

Performance : Although our algorithm is fully implemented on CPU, the volume deformation was calculated at interactive rate for all our examples. Table 1 shows some speed calculation for the different models

Limitations

Our results have shown that the correction of volume gives a more plausible visual appearance to deformed characters. The process is almost exact for small deformations (due to the linearization) in the case where only a single interior do-

model	cylinder	giraffe	elephant
Number of vertices	256	1677	5720
FPS (classical skinning)	35	27	23
FPS (volume correction)	30	19	16

Table 1: *Performances in frames per second.*

main is defined. However, when segmentation is used, the

model	cylinder 10°	cylinder 50°	cylinder 90°	elephant	giraffe
error before correction (%)	0.2	4	11	18	10
error after correction (%)	0.05	1.5	4.9	9	3

Table 2: *Relative error in the case of the cylinder (256 vertices) before anf after correction for varying angles.*

variations of the volume are only approximated due to the calculation performed on a local region with no well defined boundary. The variations are still well approximated if the surface inflates; however if twisting or blending in the local referential occurs, the error increases (see Table 2).
Reapplying the correction vector from a step to the next one could be a solution to start with a better guess. We can also consider to correct M times a variation $\frac{\Delta V}{M}$.

Also, our correction depends on the connectivity of the mesh via the computation of $\nabla V(\mathbf{P})$ on a triangulated surface. This dependence could however be limited by using a ring operator to compute the gradient value [MDSB02].

The described fold-over free method is mainly oriented to deal with self intersections due to the neighboring of two bones and the associated flesh.
For the provided example of the hand, 4 to 7 displacement steps were needed to treat the fold-over with a small displacement fixed to $1/20^{\text{th}}$ of the finger width. The complete hand counted 8636 vertices. The fold-over detection ran over the four fingers leading to the weights-recalculation for 760 vertices. With the given deformation, a correction was needed for 95 vertices and was followed by a volume correction for two segments. This extra consideration was slowing down our algorithm by a factor of 25% and ran at 9 fps.
However, for complex geometry, two regions of the flesh associated to the same bone could intersect each other, and in this case our solution for preventing fold-overs would fail.
The displacement iteration could also be more-robust and speeded-up. First of all, the current implementation of this method with a constant displacement Δs implies that the orthogonal flesh width has to be larger than Δs in order to guarantee the convergence of the algorithm. This cross-section width could be taken into account to apply a larger displacement where the section is large and an exact contact is not needed. Moreover, combined with the knowledge of the distance to the associated bone, this small displacement could also be automatically adjusted to be larger in regions located far from the bone.

The assumption that more skin bulging happens in places farther from the bone may not always hold when the bones-graph becomes complex inside the body. Parameters can still be set-up in the algorithm to take into account some physiological a-priori. For instance, in the elephant case, belly deformation is principally function of the distance to the spine, thus the deformation should probably not be influenced by legs bones. In such a case, when flesh-dependencies are known, the correction weights can integrate this constraint in explicitly defining the set of bones to take into account for a given region. As show in fig. 11, the distance term d_{ks} associated to the belly region in (8) only takes into account distance to the spine.

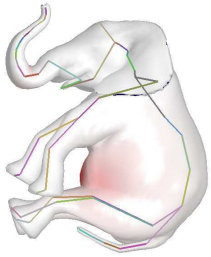

Figure 11: *Example of weight distribution in a complex case. The bulging in the belly parts depends on the distance to the spine. To give nicer results, the distances to the bones of the legs are not taken into account.*

7. Conclusion and future work

We have presented a volume correction method that preserves the volume of a character to a constant value during deformations, while acting as a correction step for standard SSD skinning, and thus enabling us to use the standard production pipeline. The input is a skin mesh and an associated animation skeleton. If not available, skinning weights can be automatically computed. Our method first applies a pre-computation stage in which the model is segmented into local correction maps distributing the strength of volume restoring displacements. Then, at each animation step, a closed-form solution to the volume restoration problem computes displacements of the vertices along their normal. Our method does not only correct the most obvious artifact of SSD (collapsing elbow) - but also generates adequate

local bulges due to the local preservation of the main sub-volumes of flesh.

The method is directly usable on preexisting models. In applications where a realistic character needs to be designed, it saves a lot of time compared to the usual set up process, since there is no longer need for binding influence shapes to the skeleton. In real-time applications, our method can replace the popular blend shapes, thus greatly reducing the amount of user input. Lastly, our plausible guess for skinning weights enables to couple the method with an automatic extraction method of the animation skeleton, and thus use it when only a skin mesh is provided.

Future work includes providing a more efficient implementation based on GPU. Although our test has been performed on the most classical, standard SSD method, our volume constraint method acts as a post-correction layer. Therefore any other better deformation method based on skinning weight such as [KZ05] is suitable. Combining our work with the dynamic skinning method of [LCA05] could also be very interesting since the addition of dynamic vibration of the flesh region would further enhance realism.

References

[AHLD07] AUJAY G., HETROY F., LAZARUS F., DEPRAZ C.: Harmonic skeleton for realistic character animation. In *Symp. on Computer Animatiom* (2007), pp. 151–160.

[Ale02] ALEXA M.: Linear combination of transformations. In *SIGGRAPH* (2002), pp. 380–387.

[AS07] ANGELIDIS A., SINGH K.: Kinodynamic skinning using volume preserving deformations. In *Symp. on Computer Animation* (2007).

[BK03] BOTSCH M., KOBBELT L.: Multiresolution surface representation based on displacement volumes. *Computer Graphics Forum 22*, 3 (2003), 483–492.

[Blo02] BLOOMENTHAL J.: Medial-based vertex deformation. In *Symp. on Computer Animation* (2002), pp. 147–151.

[CBDP05] CAPELL S., BURKHART M., DUCHAMP B. C., POPOVIC Z.: Physically based rigging for deformable characters. In *Symp. on Computer Animatiom* (2005), pp. 301–310.

[dATTS08] DE AGUIAR E., THEOBALT C., THRUN S., SEIDEL H.-P.: Automatic Conversion of Mesh Animations into Skeleton-based Animations. *Computer Graphics Forum 27*, 2 (2008).

[FTS06] FUNCK W. V., THEISEL H., SEIDEL H.-P.: Vector field based shape deformations. *Trans. on Graph. 25*, 3 (2006).

[HJCW06] HONG M., JUNKG S., CHOI M., WELCH S.: Fast volume preservation for a mass-spring system. *IEEE Computer Graphics and Applications 26*, 5 (2006).

[HML99] HIROTA G., MAHESHWARI R., LIN M. C.: Fast volume-preserving free form deformation using multi-level optimization. In *Symp. on Solid Modeling and Applications* (1999), pp. 234–245.

[HSL*06] HUANG J., SHI X., LIU X., ZHOU K., WEI L., TENG S., BAO H., GUO B., SHUM H.: Subspace gradient domain mesh deformation. In *SIGGRAPH* (2006), pp. 1126–1134.

[JT05] JAMES D. L., TWIGG C. D.: Skinning mesh animations. *Trans. on Graph. 24*, 3 (2005).

[KJP02] KRY P., JAMES D. L., PAI D. K.: EigenSkin: Real time large deformation character skinning in hardware. In *Symp. on Computer Animation* (2002), pp. 153–159.

[KZ05] KAVAN L., ZARA J.: Spherical blend skinning: a real-time deformation of articulated models. In *Symp. on Interactive 3D Graphics and games* (2005), pp. 9–16.

[LCA05] LARBOULETTE C., CANI M.-P., ARNALDI B.: Dynamic skinning: Adding real-time dynamic effects to an existing character animation. *Spring Conference on Computer Graphics* (2005).

[LCF00] LEWIS J. P., CORDNER M., FONG N.: Pose Space Deformation: A unified approach to shape interpolation and skeleton-driven deformation. In *SIGGRAPH* (2000), pp. 165–172.

[LK84] LIEN S., KAJIYA J.: A symbolic method for calculating the integral properties of arbitrary nonconvex polyhedra. *IEEE Computer Graphics and Applications 4*, 9 (October 1984).

[MDSB02] MEYER M., DESBRUN M., SCHRÖDER P., BARR A. H.: Discrete differential-geometry operators for triangulated 2-manifolds. *VisMath* (2002).

[MG03] MOHR A., GLEICHER M.: Building efficient, accurate character skins from examples. *Trans. on Graph. 22*, 3 (2003).

[MTG03] MOHR A., TOKHEIM L., GLEICHER M.: Direct manipulation of interactive character skins. In *Symposium on Interactive 3D Graphics* (2003), pp. 27–30.

[SCOLA04] SORKINE O., COHEN-OR D., LIPMAN Y., ALEXA M.: Laplacian surface editing. In *Symp. on Geometry Processing* (2004).

[SY07] SCHAEFFER S., YUKSEL C.: Example-based skeleton extraction. In *Symp. on Geometry Processing* (2007), pp. 153–162.

[SZT*07] SHI X., ZHOU K., TONG Y., DESBRUN M., BAO H., GUO B.: Mesh puppetry: Cascading optimization of mesh deformation with inverse kinematics. *Trans. on Graph.*, 3 (2007).

[WP02] WANG X. C., PHILLIPS C.: Multi-weight enveloping: Least-squares approximation techniques for skin animation. In *Symp. on Computer Animation* (2002), pp. 129–138.

[WPP07] WANG R. Y., PULLI K., POPOVIC J.: Real-time enveloping with rotational regression. *Trans. on Graph. 26*, 3 (2007).

[ZHS*05] ZHOU K., HUANG J., SNYDER J., LIU X., BAO H., GUO B., SHUN H.: Large mesh deformation using the volumetric graph laplacian. *Trans. on Graph. 24* (2005).