Atlas-Based Character Skinning with Automatic Mesh Decomposition

Lin Lu, Franck Hétroy, Cédric Gérot, Boris Thibert

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Atlas-Based Character Skinning with Automatic Mesh Decomposition

Lin Lu — Franck Hétroy — Cédric Gérot — Boris Thibert

N° ????

Janvier 2008

Thème COG

ISSN 0249-6399 ISRN INRIA/RR--????--FR+ENG

INSTITUT NATIONAL
DE RECHERCHE
EN INFORMATIQUE
ET EN AUTOMATIQUE

INRIA

centre de recherche
GRENOBLE - RHÔNE-ALPES

Atlas-Based Character Skinning with Automatic Mesh Decomposition

Lin Lu * † , Franck Hétroy ‡ * , Cédric Gérot ‡ , Boris Thibert ‡

Thème COG — Systèmes cognitifs
Équipe-Projet Evasion

Rapport de recherche n° ???? — Janvier 2008 — 18 pages

Abstract: Skinning is the most tedious part in the character animation process. Using standard methods, joint weights must be attached to each vertex of the character's mesh, which is often time-consuming if an accurate animation is required. We propose a new modeling of the skinning process, inspired by the notion of atlas of charts. Starting from the character's animation skeleton, we first automatically decompose the mesh into anatomically meaningful overlapping regions. Regions are then blended in their overlapping parts using continuous transition functions. This leads to a simple yet efficient skinning process for which the weights are automatically defined and do not depend on the Euclidean distance but on the distance on the surface.

Key-words: character animation, skinning, atlas of charts, segmentation

Part of this work was done while the first author was visiting INRIA with a "INRIA Internship" grant.

This work is part of the MEGA project (*http://evasion.imag.fr/mega/*), funded by IMAG, ELESA and INRIA.

* INRIA
† Department of Computer Science, The University of Hong Kong
‡ Grenoble Universités

Centre de recherche INRIA Grenoble – Rhône-Alpes
655, avenue de l'Europe, 38334 Montbonnot Saint Ismier
Téléphone : +33 4 76 61 52 00 — Télécopie +33 4 76 61 52 52

Skinning de personnages par une méthode basée atlas avec décomposition automatique de maillage

Résumé : Le skinning est l'étape la plus fastidieuse du processus d'animation d'un personnage. Dans les méthodes classiques, un poids associé à chaque articulation doit être attaché à chaque sommet du maillage du personnage, ce qui est souvent très coûteux en temps lorsqu'une animation précise est exigée. Nous proposons une nouvelle modélisation du processus de skinning, s'inspirant de la notion d'atlas de cartes. A partir du squelette d'animation du personnage, nous décomposons d'abord automatiquement le maillage en régions anatomiquement significatives et qui se chevauchent. Ces régions sont ensuite fusionnées dans leurs zones de chevauchement grâce à l'utilisation de fonctions de transition continues. Ceci conduit à un processus de skinning simple mais néanmoins efficace, pour lequel les poids sont automatiquement définis et ne dépendent pas de la distance euclidienne entre sommets, mais de la distance sur la surface.

Mots-clés : animation de personnages, skinning, atlas de cartes, segmentation

1 Introduction

Skeletal animation is a widespread technique to deform articulated shapes. It uses a joint hierarchy called *skeleton*; during the animation, joints are translated and/or rotated then each vertex of the shape (usually represented by a mesh) is deformed with respect to the closest joints. The process through which the mesh vertices are attached to the skeleton is called *skinning*. A lot of skinning techniques have been proposed; see Section 2.1 for a discussion. One of the first and currently the most popular one in the industry (because of its simplicity and speed) is *Linear Blend Skinning* (LBS), or Skeletal Subspace Deformation (SSD). LBS binds each vertex of the mesh to a limited number of joints (in practice, between 2 and 4): more precisely, to each vertex is associated a weighted linear blend of the transformations associated with these joints; weights represent the amount of influence of joints upon vertices. Finding these weights is not straightforward: global solutions applied the same way to all joints lead to non-realistic deformations. Thus, the animator must adjust every weight by hand or use an additional tool, which is a very tedious and time-consuming task. Several methods have been proposed to overcome the limitations of LBS (see Section 2.1). However, these methods either require a lot of memory, or remain global and deform each articulation the same way.

We propose in this paper a new skinning framework, inspired by the concept of *atlas of charts*. Indeed, on the contrary to piecewise modeling, an atlas allows us to construct a surface from pieces of surface which overlap substancially instead of abutting only along their edges. As a consequence, when one piece is stretched or moved, the overlapping pieces follow naturally with it [8]. Defining one such piece per joint of the skeleton would provide a skinning.

Our framework starts with decomposing the mesh into regions topologically equivalent to cylinders and binded to segments of a skeleton possibly with semantic information attached, such as [2]. The regions overlap around joints (Section 3). Each vertex belonging to an overlapping area is duplicated formally into several vertices binded to one region each. The relation of equivalence between duplicated vertices is conserved. The next step consists in defining weights on every vertex of each region in the input mesh (Section 4). The weights are equal to 1 in non-overlapping areas and decreased onto 0 as the vertex is closer to the boundary of the region. The relation of equivalence between duplicated vertices with these weights allows us to define a skinning for the animated mesh (Section 5).

Main advantages of this new skinning scheme are:

- it is intuitive and simple to use (mesh decomposition is automatic and equivalence relationship and weights are pre-defined entities which fit automatically to the mesh as taking intrinsically its geometry into account);
- it allows a fully automatic computation of skinning weights, based on the distance on the surface rather than the Euclidean distance, which can then be used with standard skinning techniques such as LBS;
- it does not need any manually tuned example as input, only the mesh and the skeleton;
- it is fast, because mesh decomposition is done once for all at the beginning and weight computation is simple.

2 Related work

2.1 Skeleton-driven skinning

It is usually admitted that the first paper to address the problem of skin deformation guided by an underlying skeleton is [21], in the special case of the hand. Since this seminal paper, a huge amount of methods have been proposed. As explained in the introduction, the most standard technique is currently the Linear Blend Skinning, which is an unpublished work but is described in many papers such as [16]. It has the major drawback that weights must be tuned by hand, because basic definitions, e.g. Euclidean distance to the closest joints, lead to non-realistic deformations and, most importantly, do not take into account the anatomy of the character. To generate more realistic deformations, several papers propose to use a set of example poses [16, 14, 33, 22, 34, 35]. This allows much more realistic effects, such as muscle bulging, but has some drawbacks: example usually need to be constructed by hand, and most of these methods require a lot a memory to store them.

More complex solutions exist, such as kinodynamic skinning [1]. [3] solves a heat equation for each joint in order to automatically set the weights associated to this joint: doing this, the character's geometry is better taken into account than with LBS, but there is no guarantee that generated deformations are realistic. Our approach is similar to Baran's in the sense that we do not rely on any example; however, we propose to benefit from anatomical information associated with the skeleton when available (e.g. [2]). Contrary to physically-based methods which try to simulate not only the skin but also the muscle and tissue behavior [26, 32] at high computational costs, we focus on interactive deformation, thus discarding complex simulation.

As stated before, the LBS technique has several artifacts (elbow-collapse, candy wrapping) due to the linear combination of weights. In order to overcome these limitations, several authors propose to replace matrix computation by more sophisticated tools to blend the weights, such as log-matrices [5] or dual quaternions [10]. Even if the results presented in this paper were computed using only LBS, our skinning scheme can also be applied with these methods.

2.2 Modeling with an atlas of charts

Surface modeling with an atlas of charts has properties which lends itself to the skinning problem. Indeed constructive manifold definitions [8] represent a surface as a set of blended embedded planar disks. The blending is performed as a convex combination whose weights are defined as a partition of the unity overall the planar disks. Hence the surface is made up with 3D regions which overlap substancially and are glued together. As a consequence, when one embedded planar disk is stretched or moved, the overlapping regions follow naturally with it. Defining a set of such regions per joint of the skeleton would provide a skinning.

However this construction makes sense only if the planar disks are linked together with transition functions. These functions indicate which embedded points have to be combined together in the blending process. To do so, either a *proto-manifold* associated with a mesh with a large number of pieces (at least one per vertex) is defined, [7, 24, 36], or a pre-defined manifold with a small

number of pieces but in general not adapted to the particular geometry to be represented is used [6]. These construction aim to a global highly-continuous parameterization of the surface. This implies major contraints on the definition of the transition functions.

Reversely, an atlas of charts can be constructed from the final surface to be represented in order to provide a global parameterization of it. This global parameterization is used for high-quality sampling, texture mapping [27] or reparameterization. Here again the components of the atlas have to be defined explicitely and with continuity constraints.

Real-time constraints impose us to deal with small structures and to consider meshes as C^0-surfaces. Hence, we propose to adapt this parameterization-oriented framework onto a lighter one, sufficient for skinning.

2.3 Surface decomposition

Surface decomposition, and particularly mesh segmentation (that is to say decomposition into disconnected patches), has been widely studied, for various applications such as simplification and remeshing [37, 4], parameterization [28, 11], shape analysis or modeling [23, 13], mesh editing or repair [15], shape matching [37], skeleton extraction and animation [9, 26, 18], morphing and metamorphosis [30, 37], collision detection [17, 20], etc. A recent and comprehensive study of segmentation strategies can be found in [29]. We only review here some methods intended to or with application to animation.

Some authors propose to compute a skeleton which then can be applied to animation starting from a relevant segmentation of the shape. For example, [9] first creates a hierarchical decomposition of the input mesh; then a star-shaped skeleton, where each joint corresponds to a patch of this decomposition, is inferred; a skeleton-subspace deformation method [16] is then applied to create the animation. [18] simultaneously creates a decomposition and a skeleton from a shape; this skeleton is then used for animation by re-targeting motion captured data to it. As far as we are concerned, we suppose that the animation skeleton is given as input; thus we don't need to create one. On the contrary, we benefit from this skeleton to decompose the mesh. This approach is similar to Pratscher's [26], which segments the mesh based on proximity to the skeleton's bones, and then applies some tests to correct cases when the proximity assignement of a vertex fails. As far as we are concerned, we focus on the skeleton's joints: since we want to bind regions of the mesh to the joints, it is simpler to target the segmentation boundaries directly.

[29] classifies algorithms into several categories: region growing, watershed, clustering, spectral analysis methods, graph-cut, implicit methods, inferred from a skeleton. The method we propose in this paper lies on both the last two categories, because we compute the regions by detecting their boundaries, like [15], and we use a skeleton to help the segmentation process, as [26, 20] do.

3 Skeleton-based mesh decomposition

Our framework starts with a mesh decomposition into overlapping regions. These regions are computed as a mesh segmentation; boundaries are then stretched to define overlapping areas.

Referring to the mesh segmentation problem [29], our mesh decomposition method is a skeleton-based implicit approach which first finds the region boundaries. Segmentating the mesh into meaningful patches can be treated as an optimization problem, which is not straightforward to be solved automatically, even though several methods exist in the literature [12, 19]. Using a skeleton carrying anatomical information as the input gives much potential to fulfill this decomposition automatically. In our implementation, we use the skeleton computed by [2], which contains anatomical information associated with each joint and indicates the model's symmetry axis as well. However, any other skeleton with anatomical information attached can be used.

We compute a meaningful mesh segmentation quickly and automatically in the following two steps. Then, we grow each boundary to create overlapping regions.

3.1 Boundary detection

The boundaries of our decomposition will correspond to the areas where deformation occurs. Therefore, each boundary should correspond to a skeleton joint. We have chosen to compute boundaries as closed curves with locally minimal length: each of these curves is defined as the intersection between the mesh and a plane going through the corresponding joint (see Figure 1). We compute a plane for each joint J, excluding the ones that have more than two incident bones and lie on the symmetry axis of the skeleton, such as the pelvis.

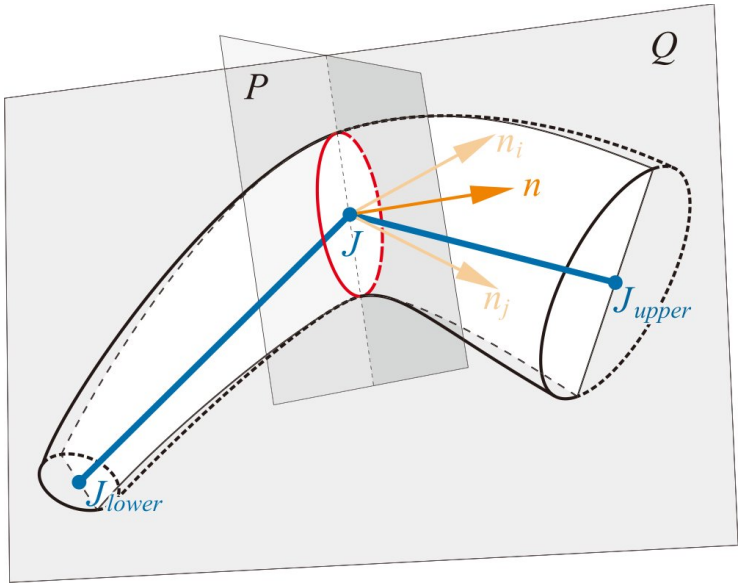


Figure 1: Each boundary is defined with respect to a plane P going through a joint J.

Let us note $P(J, n)$ such a plane. $P(J, n)$ is determined by a normal vector n and goes through J. We put the following additional condition: the normal n of $P(J, n)$ must lie in the plane Q determined by the upper bone and one lower bone of J (the *upper bone* of J is defined by J and its parent joint in the

skeleton's hierarchy). From this condition, we compute a discrete set of normals $\{n_1, n_2, \ldots, n_k\}$. Each of them defines a plane $P_1, P_2, \ldots, P_k$, since we already know the reference point J.

For each plane P_i, its boundary B_i for the joint J is computed by connecting the line segments from the intersection of P_i with intersecting triangles in sequence until it is closed (an *intersecting triangle* of P_i is a triangle incident to one or more intersecting edges of P_i [17], and an edge in the mesh is an *intersecting edge* of P_i if one of its endpoints is either on or above P_i, and the other endpoint is on or below P_i). By construction, B_i is closed and lies on the mesh surface. Note that generally the cross section between P_i and the mesh may generate more than one connected component. We keep the closed curve whose interior contains the joint, and discard the others.

Then we evaluate B_i by a function $F(B_i)$, and pick up the plane with $min(F(B_i))$. The function $F(B_i)$ can be defined as either the perimeter or the area of the boundary B_i. We use the perimeter in practice, which performs well.

After finalizing the boundary, we mark its intersecting triangles as *boundary* faces.

3.2 Region-growing segmentation

Once we obtain all the explicit boundaries, it is simple to segment the mesh into sub-meshes by a region-growing algorithm. We start with a random *non-boundary* face, and grow a sub-mesh incrementally until reaching *boundary* faces. The algorithm ends when all the faces have been visited. Each *boundary* face may be assigned to two regions, while one *non-boundary* face belongs to one region only.

Figure 2 shows a result for two characters with different poses. Note that for the first model, the skeleton has an additional joint inside the head.

Since we only handle 0-genus models, we would get k-partition of the mesh from *k-1* boundaries. Incidentally, each boundary has two adjoining regions, while each region has at least one boundary.

3.3 Overlapping area generation

We know describe how we generate a mesh decomposition into overlapping pieces from this segmentation.

Suppose that we have $k-1$ boundaries $\{B_i\}_{i=1}^{k-1}$ and k regions $\{R_j\}_{j=1}^{k}$ for the mesh. Besides, B_i has 2 adjoining regions denoted by R_{i1} and R_{i2}; R_j has n boundaries denoted by $\{B_{jk}\}_{k=1}^{n}$. Then for each vertex v in R_j, we compute its geodesic distance to B_{jk}, which can be achieved by finding the minimal geodesic distance from v to all points in B_{jk}, using methods in e.g. [25, 31]. Thus, we establish the distance field for all boundaries. Afterwards, for each B_i we simply use its approximate diameter D_i as the criterion to generate the overlapping part for its two adjoining regions. That is, we mark all the vertices with the distance to B_i less than $D_i * 0.5$ as in the overlapping part for R_{i1} and R_{i2} respectively. As a result, each region R_j with n boundaries has n overlapping parts accordingly. An example of regions with overlapping parts is provided in Figure 3.

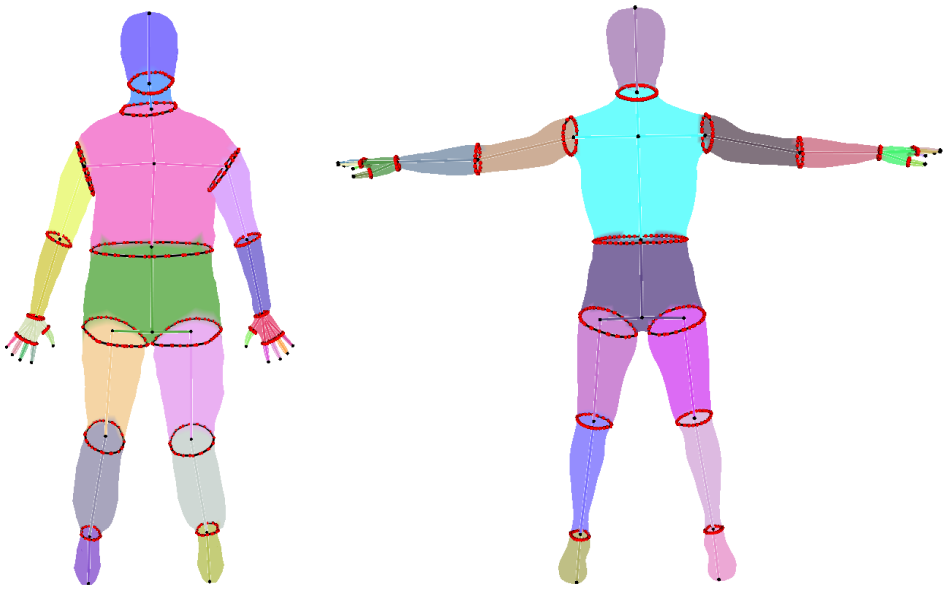
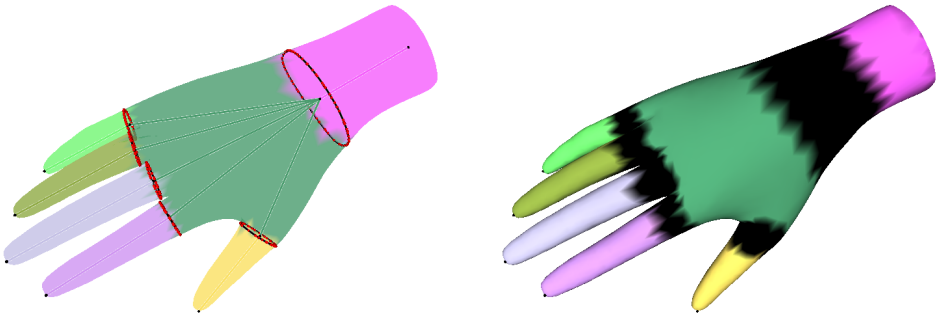


Figure 2: Segmentation results for two human models, together with their skeletons and boundary curves (each bone appears in the same color than its related region).



(a) (b)

Figure 3: Hand model. (a) Segmentation result. (b) Generated overlapping areas (in black).

Note that using the diameter of each boundary to lead overlapping is just a suggestive choice, as it can be chosen according to the type of joint in practice. We use this criterion because it is common knowledge that it helps avoiding folding artifacts when using LBS [3].

4 Weight definition

We define weights that depend on the mesh decomposition (see Section 3). We denote by V_i each region. To define the weights, we first need to introduce the

cubic function β that satisfies $\beta(0) = 0$, $\beta(1) = 1$ and $\beta'(0) = \beta'(1) = 0$. A simple calculation gives:

$$\beta(l) = -2l^3 + 3l^2.$$

Let now p be a vertex of the mesh. We denote by V_1,...,V_n all the regions containing p, and we put $d_i = d(p, \partial V_i)$ the distance on the mesh between p and the boundary of V_i (for $i \in \{1, .., n\}$). If $n = 1$, then p belongs to exactly one region V_1 and $\omega_1(p) = 1$. If $n > 1$, the weight $\omega_j(p)$ of the bone b_j for the vertex p is given by:

$$\omega_j(p) = \frac{2}{n(n-1)} \sum_{i \neq j} \beta\left(\frac{d_j}{d_i + d_j}\right). \tag{1}$$

Remark that the fact that $\beta(1 - l) + \beta(l) = 1$ implies that:

$$\sum_{j=1}^{n} \omega_j(p) = \frac{2}{n(n-1)} \sum_{1 \leq i < j \leq n} \left[\beta\left(\frac{d_j}{d_i + d_j}\right) + \beta\left(\frac{d_i}{d_i + d_j}\right)\right] = 1$$

In particular, if p belongs to exactly two regions V_1 and V_2, we have:

$$\omega_1(p) = \beta\left(\frac{d_1}{d_1 + d_2}\right)$$

and

$$\omega_2(p) = \beta\left(\frac{d_2}{d_1 + d_2}\right) = 1 - \omega_1(p).$$

Note that we have used here a function β that is cubic and independant on the articulation. It could be interesting to consider other functions that may depend on the anatomy of the articulation.

5 Weight computation and skinning algorithm

Although there are various skinning techniques in the literature, we selected the standard linear blend skinning (LBS) method because of its widespread use.

Firstly, we define the *related regions* of each vertex. It just comes from intuition, a region V of the previous mesh decomposition is called a *related region* of vertex p if p belongs to V. There are basically two possibilities: p is in the overlapping part of V; or p lies exactly in V based on the mesh segmentation (i.e. before overlapping area creation), esp. in this case we tag V as the *proto-region* of p. From the above discussion, we have the assertion that each vertex has at least one related region. Furthermore, if it is an *overlapped* vertex, it has at least two related regions. Take the human model as an example: vertices around the elbow, knee etc. have 2 related regions; some vertices close to the pelvis have 3 related regions (see Figure 4); for some vertices on the chest, without considering the clavicles, there may be 4 related regions.

Now, we have the description of our skinning algorithm based on LBS. For a vertex p, suppose p has n related regions $\{V_i\}_{i=1}^n$, T_i is the transformation of V_i, and ω_i is the weight of V_i for p, the position of the transformed p should be $\sum_{i=1}^{n} \omega_i \mathrm{T}_i(p)$, where $\sum_{i=1}^{n} \omega_i = 1$.

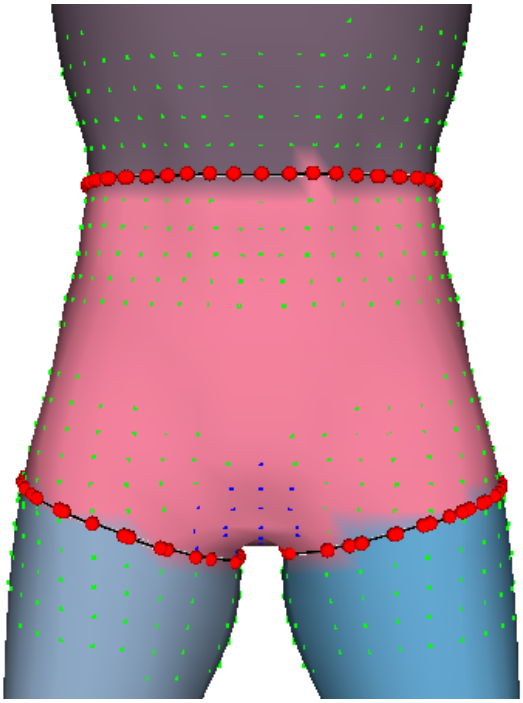


Figure 4: An example of the related regions of vertices. Vertices with green color have 2 related regions, and blue dots represent vertices with 3 related regions. The proto-region of each blue vertex in this figure is the pink region.

Note that in the current implementation, we do not ensure that each region is associated to only one bone (see the pelvis of the human model), which restricts some peculiar poses in animation, however, makes many things easier.

Then the problem left is just defining weights ω_i for the vertices as discussed in Section 4. Since non-overlapped vertices have only one related region and only need to follow the transformation of its related region rigidly, we just focus on the overlapped vertices.

According to Eqn. (1), we should find the distance d_i between p and V_i first. In the generation of overlapping regions based on boundary curves that we talked above, we have built the geodesic distance field from each vertex to each associated boundary curve. Suppose the boundary curve between V_i and V_j is $B_{i,j}$. Then denote the geodesic distance from p to B_i by $g(p, B_i)$, and let the overlapping length we designed for $B_{i,j}$ be $D(B_{i,j})$, simplified as $D_{i,j}$.

We compute the distance by:

$$d_{i,j}(p) = \begin{cases} \dfrac{D_{i,j}/2 + g(p, B_{i,j})}{D_{i,j}} & \exists B_{i,j} \& V_i \text{ is p's } \textit{proto-region} \\ \dfrac{D_{i,j}/2 - g(p, B_{i,j})}{D_{i,j}} & \exists B_{i,j} \& V_j \text{ is p's } \textit{proto-region} \\ 0.5 & B_{i,j} \text{ does not exist} \end{cases}$$

$$i = 1, 2, \ldots, n;\ j = 1, 2, \ldots, n;\ i \neq j$$

Then the weights for vertex v may be calculated by:

$$\omega_j(p) = \frac{2}{n(n-1)} \sum_{i \neq j} \beta(d_{i,j})$$

For example, if $n = 2$, i.e. vertex p has two related regions $\{V_1, V_2\}$ and V_1 is the *proto-region* of p (Figure 5). Obviously V_1 and V_2 share one boundary curve, suppose $B_{1,2} = B_{2,1} = B$, and its diameter is D, $D_{1,2} = D_{2,1} = D$. Then, the weights of p are:

$$\omega_1(p) = \beta(d_{1,2}) = \beta\left(\frac{D/2 + g(v, B)}{D}\right)$$

and

$$\omega_2(p) = \beta(d_{2,1}) = \beta\left(\frac{D/2 - g(v, B)}{D}\right) = 1 - \omega_1(p)$$

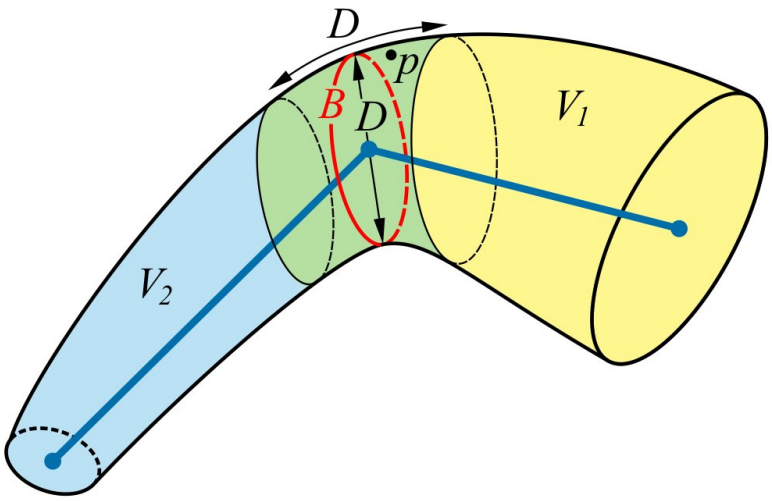


Figure 5: Vertex p with its two related regions V_1 and V_2.

6 Results

Figures 6 and 7 show some deformation results created with our skinning scheme, based on LBS.

Like Baran and Popović [3], we cannot avoid some inherent artifacts of LBS, known as "skin collapse" (see Figure 8), "twisting elbow problem" or "candy-wrapper artifact" when using our framework with LBS, because we do not change the linear nature of the weight combination. However, our weight computation framework can be applied with more sophisticated skinning techniques such as [10], which reduce these artifacts.

We tested our algorithm on several models, represented as triangle meshes with 10k to 15k faces. Computation time used for our scheme depends on different stages. The boundary detection and segmentation part is done in real-time. The most time-consuming part is the computation of geodesic distance

Figure 6: Two poses generated by our algorithm from the initial model on the left.

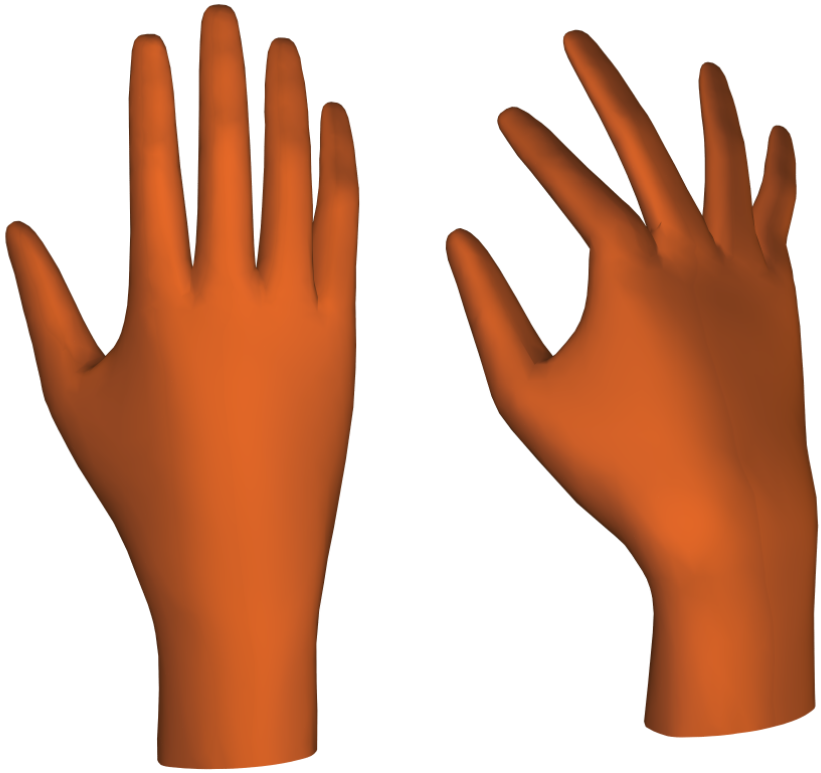


Figure 7: Deformation applied to a hand model.

field, since we need compute the geodesic distance from each vertex of the mesh to each point of all boundary curves. Suppose boundary curve B_i is composed of m points $\{c_1, c_2, \ldots, c_m\}$, $g(p, c_j)$ is the geodesic distance from vertex p to point c_j. Then we find $\min g(p, c_j)$ and use it as the geodesic distance from p

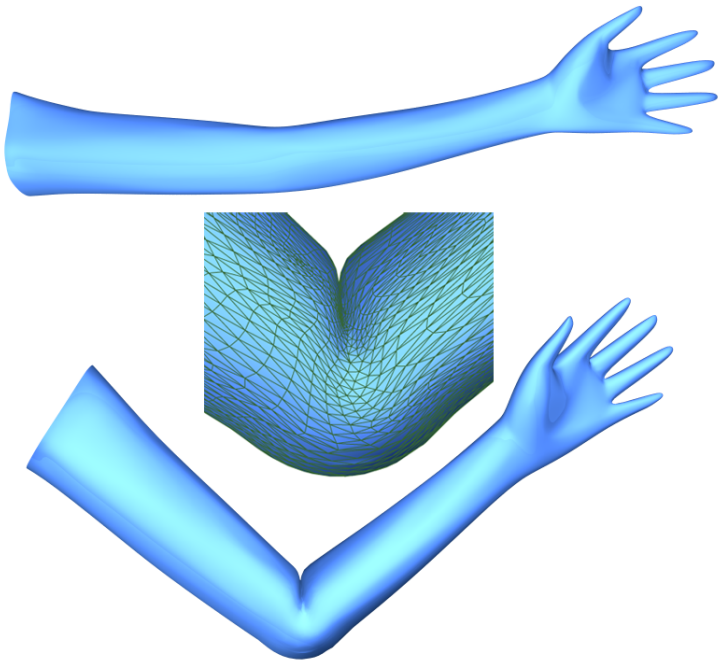


Figure 8: Skin collapse artifact observed with our framework when used with LBS.

to B_i, denoted by $g(p, B_i)$. Here, we speed it up by firstly finding the closest point c_k to p by Euclidean distance, and then only checking geodesic distance between p and a fixed number(5, for example) of neighboring points to c_k. It takes nearly 4 minutes to generate overlapping regions and compute weights for each overlapped vertex for a human model with 14k faces, 24 boundary curves and 25 parts on a 1.66GHz Pentium IV computer with 1GB RAM. The skinning stage is very fast: the deformation can be achieved in real-time, at 25 FPS for all tested models.

7 Conclusion

In this paper, we have proposed a new framework for character skinning, inspired from the mathematical concept of *atlas of charts*. We first automatically decompose the input mesh into anatomically meaningful regions, overlapping around the skeleton joints. Each vertex belonging to an overlapping area is duplicated formally into several vertices binded to one region each. The final position of this vertex through animation is given by a convex combination of this several duplicated vertices. The weights used in this blending, that is the skinning weights, are defined in a similar way as a partition of the unity : they lie in $[0, 1]$ and sum up to 1 over each set of duplicate vertices. These weights are automatically computed, based on the geodesic distance on the surface between vertices and region boundaries. Our results show that, after a few

minutes of pre-processing for the mesh decomposition, we can achieve real-time deformations.

Apart from applying our weight definition to other blending deformation methods such as those founded on dual quaternions [10], future work includes defining skinning weights adapted to each joint. In case we benefit from semantic information associated with the skeleton [2], these weights, still automatically computed, would help generate much more realistic deformations. We plan to also investigate the use of larger overlapping areas together with other weight definitions in order to mimick some effects such as muscle bulging. One can go further in the similarity with parameterization with an atlas of chart as providing a skinning framework for multiresolution animated meshes founded on our *pseudo-parameterization* on the inital mesh.

References

[1] A. Angelidis and K. Singh, *Kinodynamic Skinning using Volume-Preserving Deformations*, Symposium on Computer Animation, 2007.

[2] G. Aujay, F. Hétroy, F. Lazarus and C. Depraz, *Harmonic Skeleton for Realistic Character Animation*, Symposium on Computer Animation, 2007.

[3] I. Baran and J. Popović, *Automatic Rigging and Animation of 3D Characters*, ACM Transactions on Graphics (SIGGRAPH proceedings), 26(3), 2007.

[4] D. Cohen-Steiner, P .Alliez and M. Desbrun, *Variational Shape Approximation*, ACM Transactions on Graphics (SIGGRAPH proceedings), 23(3), 2004.

[5] F. Cordier and N. Magnenat-Thalmann, *A Data-Driven Approach for Real-Time Clothes Simulation*, Computer Graphics Forum, 24(2), 2005.

[6] C. Grimm, *Parameterization using Manifolds*, International Journal of Shape Modeling, 10(1), 2004.

[7] C. Grimm and J. Hughes, *Modeling Surfaces of Arbitrary Topology*, SIGGRAPH, 1995.

[8] C. Grimm and D. Zorin, *Surface Modeling and Parameterization with Manifolds*, SIGGRAPH Course Notes, 2005.

[9] S. Katz and A. Tal, *Hierarchical Mesh Decomposition using Fuzzy Clustering and Cuts*, ACM Transactions on Graphics (SIGGRAPH proceedings), 22(3), 2003.

[10] L. Kavan, S. Collins, J. Zara and C. O'Sullivan, *Skinning with Dual Quaternions*, Symposium on Interactive 3D Graphics and Games, 2007.

[11] V. Kraevoy and A. Sheffer, *Cross-Parameterization and Compatible Remeshing of 3D Models*, ACM Transactions on Graphics (SIGGRAPH proceedings), 23(3), 2004.

[12] V. Kraevoy and A. Sheffer, *Variational, Meaningful Shape Decomposition*, SIGGRAPH Sketches, 2006.

[13] V. Kraevoy, D. Julius and A. Sheffer, *Shuffler: Modeling with Interchangeable Parts*, The Visual Computer, 2007.

[14] P. Kry, D. James and D. Pai, *Eigenskin: Real Time Large Deformation Character Skinning in Hardware*, Symposium on Computer Animation, 2002.

[15] Y. Lee, S. Lee, A. Shamir, D. Cohen-Or and H.P. Seidel, *Mesh Scissoring with Minimal Rule and Part Salience*, Computer Aided Geometric Design, 22(5), 2005.

[16] J.P. Lewis, M. Cordner and N. Fong, *Pose Space Deformation: a Unified Approach to Shape Interpolation and Skeleton-Driven Deformation*, SIGGRAPH, 2000.

[17] X. Li, T.W. Toon and Z. Huang, *Decomposing Polygon Meshes for Interactive Applications*, Symposium on Interactive 3D Graphics and Games, 2001.

[18] J.M. Lien and N. Amato, *Simultaneous Shape Decomposition and Skeletonization*, Symposium on Solid and Physical Modeling, 2006.

[19] R. Liu and H. Zhang, *Mesh Segmentation via Spectral Embedding and Contour Analysis*, Computer Graphics Forum (Eurographics proceedings), 26(3), 2007.

[20] L. Lu, Y.K. Choi, W. Wang and M.S. Kim, *Variational 3D Shape Segmentation for Bounding Volume Computation*, Computer Graphics Forum (Eurographics proceedings), 26(3), 2007.

[21] N. Magnenat-Thalmann, R. Laperrière and D. Thalmann, *Joint-Dependent Local Deformations for Hand Animation and Object Grasping*, Graphics Interface, 1988.

[22] A. Mohr and M. Gleicher, *Building Efficient, Accurate Character Skins from Examples*, ACM Transactions on Graphics (SIGGRAPH proceedings), 22(3), 2003.

[23] M. Mortara, G. Patanè, M. Spagnuolo, B. Falcidieno and J. Rossignac, *Plumber: a Method for a Multi-Scale Decomposition of 3D Shapes into Tubular Primitives and Bodies*, Symposium on Solid Modeling and Applications, 2004.

[24] J. Cotrina-Navau and N. Pla-Garcia, *Modeling Surfaces from Meshes of Arbitrary Topology*, Computer Aided Geometric Design, 17(1), 2000.

[25] M. Novotni and R. Klein, *Computing Geodesic Distances on Triangular Meshes*, International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG), 2002.

[26] M. Pratscher, P. Coleman, J. Laszlo and K. Singh, *Outside-In Anatomy Based Character Rigging*, Symposium on Computer Animation, 2005.

[27] E. Praun, A. Finkelstein and H. Hoppe, *Lapped Textures*, SIGGRAPH, 2000.

[28] P. Sander, J. Snyder, S. Gortler and H. Hoppe, *Texture Mapping Progressive Meshes*, SIGGRAPH, 2001.

[29] A. Shamir, *Segmentation and Shape Extraction of 3D Boundary Meshes*, Eurographics State-of-the-Art Report, 2006.

[30] S. Shlafman, A. Tal and S. Katz, *Metamorphosis of Polyhedral Surfaces using Decomposition*, Computer Graphics Forum (Eurographics proceedings), 21(3), 2002.

[31] J. Tang, G.S. Wu, F.Y. Zhang and M.M. Zhang, *Fast Approximate Geodesic Paths on Triangle Mesh*, International Journal of Automation and Computing, 4, 2007.

[32] J. Teran, E. Sifakis, S. Blemker, V. Ng-Thow-Hing, C .Lau and R .Fedkiw, *Creating and Simulating Skeletal Muscle from the Visible Human Data Set*, IEEE Transactions on Visualization and Computer Graphics, 11(3), 2005.

[33] X.C. Wang and C. Phillips, *Multi-Weight Enveloping: Least-Squares Approximation Techniques for Skin Animation*, Symposium on Computer Animation, 2002.

[34] R.Y. Wang, K. Pulli and J. Popović,, *Real-Time Enveloping with Rotational Regression*, ACM Transactions on Graphics (SIGGRAPH proceedings), 26(3), 2007.

[35] O. Weber, O. Sorkine, Y. Lipman and C. Gotsman, *Context-Aware Skeletal Shape Deformation*, Computer Graphics Forum (Eurographics proceedings), 26(3), 2007.

[36] L. Ying and D. Zorin, *A Simple Manifold-Based Construction of Surfaces of Arbitrary Smoothness*, ACM Transactions on Graphics (SIGGRAPH proceedings), 23(3), 2004.

[37] E. Zuckerberger, A. Tal and S. Shlafman, *Polyhedral Surface Decomposition with Applications*, Computer and Graphics, 26(5), 2002.

Contents

Centre de recherche INRIA Grenoble – Rhône-Alpes
655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)

Centre de recherche INRIA Bordeaux – Sud Ouest : Domaine Universitaire - 351, cours de la Libération - 33405 Talence Cedex
Centre de recherche INRIA Lille – Nord Europe : Parc Scientifique de la Haute Borne - 40, avenue Halley - 59650 Villeneuve d'Ascq
Centre de recherche INRIA Nancy – Grand Est : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex
Centre de recherche INRIA Paris – Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex
Centre de recherche INRIA Rennes – Bretagne Atlantique : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex
Centre de recherche INRIA Saclay – Île-de-France : Parc Orsay Université - ZAC des Vignes : 4, rue Jacques Monod - 91893 Orsay Cedex
Centre de recherche INRIA Sophia Antipolis – Méditerranée : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399