PPS: Pose-to-Pose Skinning of Animated Meshes

Andreas A. Vasilakis
Inform. Technologies Institute
Centre for Research &
Technology Hellas
Athens, Greece
abasilak@iti.gr

Ioannis Fudos
Department of Computer
Science & Engineering
University of Ioannina
Ioannina, Greece
fudos@cs.uoi.gr

George Antonopoulos
Department of Computer
Science & Engineering
University of Ioannina
Ioannina, Greece
gantonop@cs.uoi.gr

ABSTRACT

In computer graphics, animation compression is essential for efficient storage, streaming and reproduction of animated meshes. Previous work has presented efficient techniques for compression using skinning transformations to derive the animated mesh from a reference pose. We present a pose-to-pose approach to skinning animated meshes by observing that only small deformation variations will normally occur between consecutive poses. The transformations are applied so that a new pose is derived by deforming the geometry of the previous pose, thus maintaining temporal coherence in the parameter space, reducing approximation error and facilitating forward propagated editing of arbitrary poses.

CCS Concepts

•**Computing methodologies** → **Animation;**

Keywords

skinning; compression; editing; temporal coherency

1. INTRODUCTION

Linear blend skinning (**LBS**) [10] is a popular mesh deformation technique built around the observation that a quasi-rigid animation can be compactly described by its skeleton bones. In spite of the numerous variations introduced through the years aiming to eliminate its well-known limitations [3], LBS is, still, a widely-accepted method in the animation industry due to its simplicity and straightforward GPU implementation.

In LBS, a vertex $\mathbf{v}_j^p$, $j \in [1..\mathrm{N}]$ (N is the number of vertices) in pose $p \in [1..\mathrm{P}]$ (P is the number of poses) can be approximated by the convex weighted combination of the affine transformations $\mathbf{M}_b^p \in \mathbb{R}^{4\times4}$ associated with each bone $b \in [1..\mathrm{B}]$ (B is the number of bones) that influences the vertex applied on a *reference-* or *rest-pose* $\mathbf{v}_j \in \mathbb{R}^4$ (see Fig. 1):

$$\hat{\mathbf{v}}_j^p = \mathbf{T}_j^p \mathbf{v}_j,\ \mathbf{T}_j^p = \sum_{b=1}^{\mathrm{B}} w_{b,j} \mathbf{M}_b^p \tag{1}$$

where $w_{b,j} \in \mathbb{R}_{\geq 0} \mid \Sigma_{b=1}^{\mathrm{B}} w_{b,j} = 1$ represents the amount of b-th bone influence on the j-th vertex. Due to graphics hardware considerations, it is common to assume there are at most four non-zero weights for every vertex. The bone transformations are typically the only quantity that is allowed to vary during the course of an animation. Assuming that the bone distribution and influence have been already established, the general formulation for the skinning approximation (i.e the computation of the transformation matrices) can be formulated as a least-square problem as follows:

$$\min_{\mathbf{M}_b^p} \left\{ \sum_{j=1}^{\mathrm{N}} \left\| \hat{\mathbf{v}}_j^p - \mathbf{v}_j^p \right\|^2 \right\},\ \forall p \in [1..\mathrm{P}] \tag{2}$$

where $\mathbf{v}_j^p$ corresponds to the original j-th vertex position of p-th pose of the input animation [7]. Without loss of generality, the first pose is often used as the rest-pose: $\mathbf{v}_j = \mathbf{v}_j^1$.

In the context of animation compression, James and Twigg (in **SMA** [4]) were the first to explore the use of LBS to approximately reproduce articulated characters as a function of their bone movement. Extending this work, Kavan et al. (in **SAD** [7]) presented a *dual quaternion skinning* scheme [6] that can compute approximations for highly-deformable animations by suggesting that uniformly selected points on the mesh can act as bones. SMA and SAD enhance their final skinned approximation by exploiting **EigenSkin** [9]; a PCA-based correction technique that removes a large part of skin distortion with the cost of higher storage demands. However, both methods provide limited editing of the mesh sequence by either (non-inherited) modification of the bone transformations or by propagating small changes of the rest-pose geometry over the subsequent poses. To support arbitrary animations, Kavan et al. (in **FESAM** [8]) introduced a cyclic coordinate descent algorithm that optimizes all of the skinning parameters in an iterative manner. While FESAM offers high-quality reproduction results, is limited to download-and-play scenarios.

In this work, we introduce a pose-to-pose skinning technique that exploits temporal coherence. It enables the full spectrum of applications supported by previous approaches in conjunction with a novel pose editing of arbitrary animation poses, which can be smoothly propagated through the subsequent ones generating new deformed mesh sequences. Although skinning is performed between successive poses,

hardware accelerated rendering is also possible by converting our method from an arbitrary- to a rest-pose reproduction scheme. Finally, we present vertex and weight correction techniques that reduce the skinning error without increasing the storage demands.

2. POSE-TO-POSE SKINNING

Previous approaches compute the transformations that describe the transition from the rest-pose to an arbitrary pose of the animation sequence (**RPS**). While these perform well for a variety of deformations, artifacts tend to be more persistent the farther a deformation deviates from the rest-pose shape due to insufficient degrees of freedom in the LBS. In a non-streaming scenario where the animation sequence is known a priori, utilizing a mean pose shape (computed by averaging all poses [2]) as the basis of all deformations may improve approximation, but is not enough to make it suitable for handling undefined animations (see Fig. 2). To this end, we exploit temporal coherence by observing that only small deformation variations will normally occur between consecutive poses. We reformulate Eq. 1 to handle a pose-to-pose skinning scheme (**PPS**) by using the approximation of the previous pose as the rest-pose for the current one:

$$\hat{\mathbf{v}}_j^p = \mathbf{Q}_j^p \hat{\mathbf{v}}_j^{p-1}, \;\; \mathbf{Q}_j^p = \sum_{b=1}^{\mathrm{B}} w_{b,j} \mathbf{L}_b^p \tag{3}$$

where $\hat{\mathbf{v}}_j^1 = \mathbf{v}_j$ and $\mathbf{L}_b^p$ represent the affine transformation matrices from pose $p-1$ to pose p of the b-th bone (see Fig. 1). Although skinning is performed from pose-to-pose, the fitting can be computed in the same way as with the rest-pose scheme (Eq. 2). Last but not least, a reproduction scheme from the rest-pose to an arbitrary pose can also be derived by adjusting $\mathbf{T}_j^p$ estimation of Eq. 1 as follows:

$$\mathbf{T}_j^p = \mathbf{Q}_j^p \mathbf{T}_j^{p-1} = \prod_{t=1}^{p} \mathbf{Q}_j^t, \;\; \mathbf{Q}_j^1 = \mathbf{I}_4 \tag{4}$$

Note that the although the PPS scheme can also be supported via dual quaternion skinning, we omit the details due to space requirements. Due to the sequential nature of the fitting process, the PPS cannot be implemented in parallel. Also, the skinning artifacts that occur in a single pose are propagated to the subsequent ones. To address the latter issue, we offer correction techniques yielding approximation quality comparable to previous methods.

2.1 Skinning Corrections

Following iterative coordinate descent optimization of FESAM, PPS can further adjust all the skinning parameters by applying displacements to the rest-pose positions and the vertex weights. While this iterative optimization process results in blending away most of the differences between original and skinned approximations, requires the animation sequence to be known a priori making it unsuitable for streaming application scenarios.

I) Vertex displacements. Inspired by a volume correction method which extends LBS [11], we introduce a rest-pose position displacement correction solution which is computed directly and eliminates distortion artifacts produced by transformation fitting. The corrections are embedded in the resulted skinned mesh and are not stored separately as

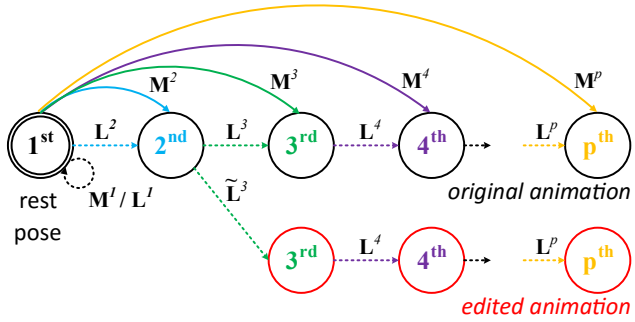

Figure 1: (top) Rest-pose versus pose-to-pose skinning approximation methods. (bottom) Observe how the pose editing is supported by our method.

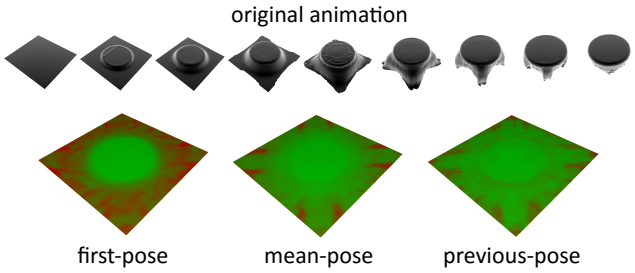

Figure 2: (top) Observe that only small deformation variations occur between consecutive frames of a *tablecloth* animation (bottom) when previous pose, instead of first and mean, is used as the rest-pose.

in Eigenskin. Note that this is possible due to the LBS-aware transition of PPS via Eq. 4. Given the computed weight values and transformation matrices from all poses, we define a displacement field $\mathbf{cv} = [\mathbf{cv}_1, \ldots, \mathbf{cv}_{\mathrm{N}}] \in \mathbb{R}^{4\times \mathrm{N}}$ that if added to the vertex positions of the rest-pose, will enhance skinning output. To ensure validity in terms of the homogeneous coordinates, (i.e. the corrected rest-pose must lie on the $w = 1$ plane), we set the w coordinates of $\mathbf{cv}$ to zero. Formally, the problem can be stated as

$$\min_{\mathbf{cv}_j} \left\{ \sum_{p=1}^{\mathrm{P}} \left\| \mathbf{T}_j^p (\mathbf{v}_j + \mathbf{cv}_j) - \mathbf{v}_j^p \right\|^2 \right\}, \;\; \forall j \in [1, \mathrm{N}] \tag{5}$$

which is equivalent to finding the least squares solution of $\overline{\mathbf{T}}_j^p \overline{\mathbf{cv}}_j = \overline{\mathbf{d}}_j^p$ where $\mathbf{d}_j^p = \mathbf{v}_j^p - \hat{\mathbf{v}}_j^p$ and an over-bar denotes the 3D vector for a homogeneous vector and the top left 3×3 sub-matrix for an affine matrix, respectively. The above minimization problem can be rewritten as a linear system of the form $\mathbf{Ax} = \mathbf{b}$ for all vertices, where $\mathbf{A}$ is a vector of N array blocks $\in \mathbb{R}^{3\mathrm{P}\times 3}$; each of contains the corresponding weighted transformation matrices. Finally, $\mathbf{b}$ is formed by stacking vectors $\mathbf{d}_1^p, \ldots, \mathbf{d}_{\mathrm{N}}^p$.

II) Weight displacements. One disadvantage of the aforementioned technique is that it cannot be directly applied for weight corrections of the PPS (by using the rest-pose scheme of Eq. 4), since it would lead to a extremely complex system of non-linear equations. To this end, we alter the weight optimization by employing the pose-to-pose scheme (Eq. 3). If $\mathbf{cw} = [\mathbf{cw}_1, \ldots, \mathbf{cw}_{\mathrm{N}}] \in \mathbb{R}^{B\times \mathrm{N}} \mid \mathbf{cw}_j = [cw_{1,j}, \ldots, cw_{\mathrm{B},j}]^{\mathrm{T}}$

represents the weight displacement field, the optimization problem can be formally stated as

$$\min_{\mathbf{cw}_j} \left\{ \sum_{p=1}^{\mathrm{P}} \left\| \sum_{b=1}^{\mathrm{B}} (w_{b,j} + cw_{b,j}) \mathbf{L}_b^p \hat{\mathbf{v}}_j^{p-1} - \mathbf{v}_j^p \right\|^2 \right\}, \ \forall j \in [1..\mathrm{N}] \tag{6}$$

This is equivalent to the least squares solution of the system $\sum_{b=1}^{\mathrm{B}} cw_{b,j}\mathbf{y}_{b,j}^p = \mathbf{d}_j^p$, $\mathbf{y}_{b,j}^p = \mathbf{L}_b^p \hat{\mathbf{v}}_j^{p-1} \in \mathbb{R}^3$, which can be expressed as a system of linear equations of the form $\mathbf{Ax} = \mathbf{b}$ for all vertices, where $\mathbf{A}$ is a vector of N array blocks $\in \mathbb{R}^{3\mathrm{P}\times\mathrm{B}}$ each of contains the transformed skinned pose vertices for each of the influencing bones as well as $\mathbf{b}$ contains the approximation error values as before. To avoid over-fitting artifacts [4]; arise when corrected weights contain either large positive or negative values, we impose (i) *convexity* and (ii) *non-negativity* constraints to Eq. 6 by solving our linear system subject to:

$$\text{(i)} \ \sum_{b=1}^{\mathrm{B}} (w_{b,j} + cw_{b,j}) = 1 \Leftrightarrow \sum_{b=1}^{\mathrm{B}} cw_{b,j} = 0 \tag{7}$$

$$\text{(ii)} \ w_{b,j} + cw_{b,j} > 0 \Leftrightarrow cw_{b,j} > -w_{b,j}, \ \forall b \in [1..\mathrm{B}] \tag{8}$$

While we are able to enhance weight influences, the total skinning approximation behavior will probably be undefined due to the sequential nature of the PPS fitting process (i.e. we consider $\hat{\mathbf{v}}_j^p$ to be fixed, but vary after the weight correction process). To solve this, either an additional transformation fitting (Eq. 2) or a vertex correction iteration (Eq. 5) is compulsory to be subsequently performed.

2.2 Arbitrary Pose Editing

Contrary to prior RPS-based methods [4, 7], where geometry editing is restricted to be conducted in the rest-pose, PPS can efficiently handle editing of any pose in the skinned sequence with the extra cost of recomputing the transformation towards the newly edited pose from the previous one. If $p_e \geq 1$ represents the edited pose and $\widetilde{\mathbf{L}}_b^{p_e}$ the matrices that transforms $p_e - 1$ pose to the edited one (see also Fig. 1), an RPS representation for all poses after the edited one ($p > p_e$) is derived by altering Eq. 4:

$$\mathbf{T}_j^p = \mathbf{T}_j^{p_e,p} \left(\sum_{b=1}^{\mathrm{B}} w_{b,j} \widetilde{\mathbf{L}}_b^{p_e} \right) \mathbf{T}_j^{p_e-1}, \ \mathbf{T}_j^{t_0,t_1} = \prod_{t=t_0}^{t_1} \mathbf{Q}_j^t \tag{9}$$

The result animation after editing an arbitrary pose of a highly-deformable animation is illustrated in Figure 3. In the case of rest-pose editing ($p_e = 1$, $\widetilde{\mathbf{L}}_b^1 = \mathbf{I}_4$), geometry manipulation is limited to be applied at the uncorrected pose (i.e. without any displacements), since users intuitively perform editing operations on the original reference shape. Figure 4 demonstrates how editing is propagated to the consecutive poses of a skeletal-based animation after densely manipulating the leftmost one.

3. RESULTS

We compare our PPS scheme against two RPS-based frameworks; SAD and FESAM, under a variety of animation sequences using as quality criterion the widely-accepted **KG** approximation error metric [5]. For a fair comparison, we follow the pipeline stages of each method; altering only the reference rest-pose from first- to pose-to-pose.

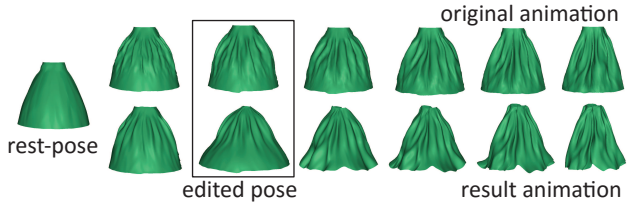

Figure 3: (top) The result of editing the third pose of a skirt deformation and (bottom) subsequently applying it to the rest pose-to-pose transformations.

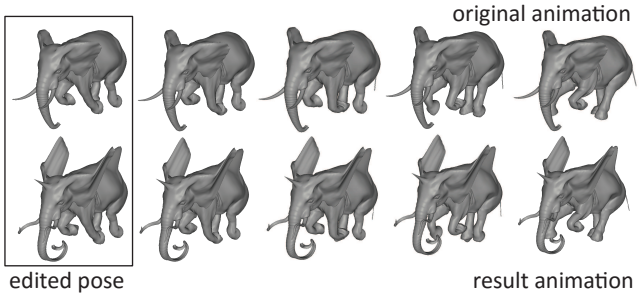

Figure 4: (top) The original sequence of an elephant animation. (bottom) A new animation is derived after propagating the editing at the first pose.

Comparison using SAD. SAD is widely suitable for streaming of animation sequences, since it does not make any assumptions about the input animation ensuring that the overall deformation is reasonably approximated. This pipeline consists of three stages: it uniformly distributes bone influences over the rest-pose mesh; optimizes their transformations using least-squares (Eq. 2); and finally employs Eigen-Skin corrections (typically lot) to achieve visually pleasing skinned reconstructions as illustrated in Figure 5. Figure 7 shows the superiority of PPS, compared to RPS, when adopted by the SAD framework on a skeletal (*horse gallop*) and a highly-deformable (*tablecloth*) animation sequence with P = 10 and B = 50.

Comparison with FESAM. FESAM is an algorithm that produces more accurate approximations than SAD by optimizing all of the skinning parameters respectively: bone transformations; vertex weights; and rest pose positions as illustrated in Figure 6. A deformation-aware region-growing clustering method is initially used for distributing bones and determining the weights per vertex. Figure 8 shows that PPS requires more optimization iterations compared to the RPS to achieve convergence when skinning a highly-deformable (*horse collapse*) animation. This is reasonable since PPS relies on a suboptimal, but necessary (gray versus yellow/orange lines), weight correction optimization step.

4. CONCLUSIONS

We have presented a novel pose-to-pose approach to processing animated meshes. To alleviate the propagation of skinning flaws that occur in a pose to the subsequent ones, we offer optimization techniques that yield approximation quality comparable to previous methods. To the best of our knowledge, this is the first skinning method to support

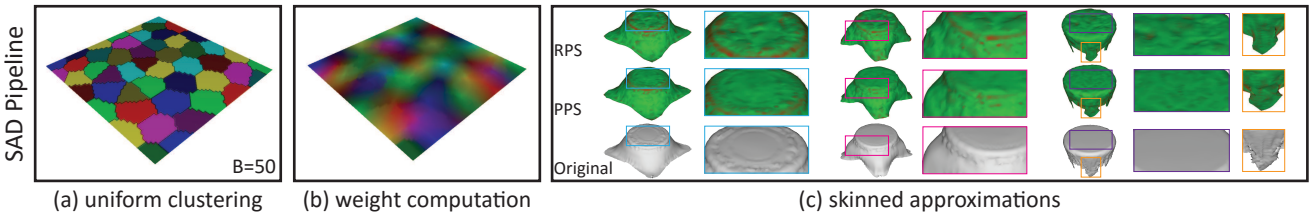

Figure 5: SAD pipeline overview. (a) A uniform clustering is initially employed to (b) compute the per-vertex weight influences of a *tablecloth* animation. (c) Skinned approximations are highly-improved when RPS is replaced by PPS. A heat-map visualization is used to highlight the errors from the original poses.

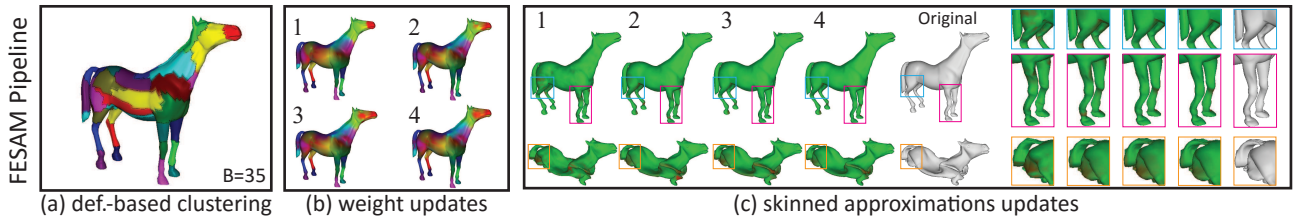

Figure 6: FESAM pipeline overview. While (a) the deformation-based clustering method produces (b-c,1) coarse weight distribution and skinned outputs, this is fixed in (b-c,2-4) the consecutive optimization steps.

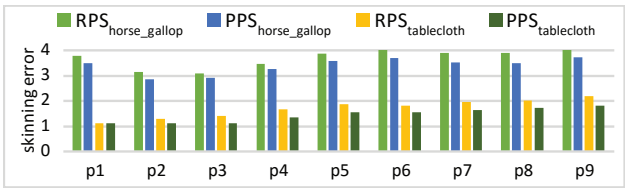

Figure 7: Per-pose error measures when RPS and PPS schemes are used on the SAD pipeline.

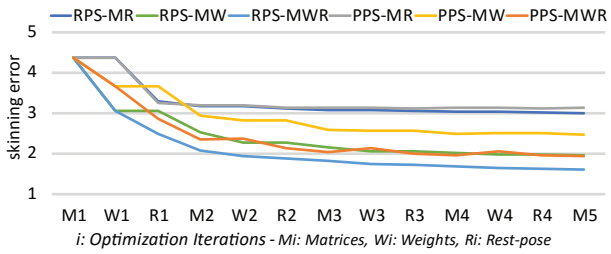

Figure 8: Error behavior comparison between RPS and PPS, with rest-pose or/and weight corrections, when increasing the number of FESAM iterations.

arbitrary pose editing operations. Finally, we believe that temporal coherence deserves further investigation; e.g. a promising direction is to conduct adaptive weight distribution guided by variable segmentation techniques [1].

Acknowledgments. The mesh data used in this paper was made available by Robert Sumner and Jovan Popovic from the Computer Graphics Group at MIT.

5. REFERENCES

[1] A. A. Vasilakis and I. Fudos. Pose partitioning for multi-resolution segmentation of arbitrary mesh animations. *Computer Graphics Forum*, 33(2):293–302, 2014.

[2] N. Hasler, T. Thormählen, B. Rosenhahn, and H.-P. Seidel. Learning skeletons for shape and pose. In *Proceedings of the 2010 ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, I3D '10, pages 23–30, 2010.

[3] A. Jacobson, Z. Deng, L. Kavan, and J. Lewis. Skinning: Real-time shape deformation. In *ACM SIGGRAPH 2014 Courses*, 2014.

[4] D. L. James and C. D. Twigg. Skinning mesh animations. *ACM Trans. Graph.*, 24(3):399–407, 2005.

[5] Z. Karni and C. Gotsman. Compression of soft-body animation sequences. *Computers & Graphics*, 28(1):25 – 34, 2004.

[6] L. Kavan, S. Collins, J. Žára, and C. O'Sullivan. Geometric skinning with approximate dual quaternion blending. *ACM Trans. Graph.*, 27(4):105:1–23, 2008.

[7] L. Kavan, R. McDonnell, S. Dobbyn, J. Žára, and C. O'Sullivan. Skinning arbitrary deformations. In *Proceedings of the 2007 Symposium on Interactive 3D Graphics and Games*, I3D '07, pages 53–60, 2007.

[8] L. Kavan, P.-P. Sloan, and C. O'Sullivan. Fast and efficient skinning of animated meshes. *Computer Graphics Forum*, 29(2):327–336, 2010.

[9] P. G. Kry, D. L. James, and D. K. Pai. Eigenskin: Real time large deformation character skinning in hardware. In *Proceedings of the 2002 ACM SIGGRAPH/ Eurographics Symposium on Computer Animation*, SCA '02, pages 153–159, 2002.

[10] N. Magnenat-Thalmann, R. Laperrière, and D. Thalmann. Joint-dependent local deformations for hand animation and object grasping. In *Proceedings on Graphics Interface '88*, pages 26–33, 1988.

[11] W. von Funck, H. Theisel, and H. Seidel. Volume-preserving mesh skinning. In *Proceedings of Vision, Modeling, and Visualization '08*, pages 409–415, 2008.