Real-time C^2-Weighting Based Character Skinning Powered by GPU

Chuhua Xian[1], Junxian Huang[1], Shuo Jin[2,*], Guoliang Luo[3], Charlie C.L. Wang[4]

[1]South China University of Technology, [2]INRIA, [3]Jiangxi Normal University, [4]Delft University of Technology

Abstract

Handle driven character skinning is widely favored in animation applications due to its advantages of intuitiveness, effectiveness and simplicity. A research thread to realize this is to compute a proper weighting distribution on points associated with specified handles like points, bars or skeleton, which is critical to the quality of manipulation and can be utilized to produce animation by controlling user handles. In this work, we introduce a new skinning method that is compatible with different model representations, with a key idea of evaluating the local influence of each handle by decomposing the shape domain into small overlapped regions. Thanks to its well-designed formulation, the computation of weights and update of models can be conducted in parallel on GPU, leading to high efficiency and good visual quality supported by the provided experimental and statistical results in this paper.

Keywords: real-time skinning, linear blending, region decomposition, GPU

1 Introduction

Skinning methods are vital to animate characters in various applications. With good skinning techniques, high-quality and delicate deformations can be realized on character models by manipulating a small set of predefined handles. There have been a large number of researches that can be exploited for this purpose. A common practice is to explicitly establish the relationship between model and handles which is usually a collection of local/global influence of all handles whose quality heavily decides the behavior of character skinning and then specify the rule of each point's movement following the handles. Recent efforts made in this research direction include *embedded deformation*[1] which allows users to manipulate defined deformation graph consisting of equally distributed graph nodes and edges indicating overlapping influence between nodes on models, *bounded biharmonic weights*[2] which requires minimizing a biharmonic energy to obtain weights for linear blending, *interior RBF*[3] which adopts a local/global approach to calculate coefficients, *optimized centers of rotation*[4] that computes the optimal rotation centers for skeletal handles and *efficient C^2-weighting*[5] that introduces a closed-form formulation to achieve high-quality weights for linear blending based skinning. Another research thread is to determine skinning result by solving preset energy minimization on models where user manipulations are usually encoded as constraints, such as *spokes and rims*[6] that computes deformed output based on an elastic differential energy, *as-rigid-as-possible* strategy based techniques[7] which evaluate deformation on models in a near rigid manner and *local subspace design*[8] which proposes an identical variational form to realize local linear deformations. Some researchers consider about utilizing the power of parallel computing to speed up computational performance in their efforts. For example, Schafer et al.[9] propose to conduct shape editing and rendering directly on GPU to avoid pre-computation and data transfer with CPU.

In this paper, we propose a weighting based

*Corresponding Author: jerry.shuojin@gmail.com

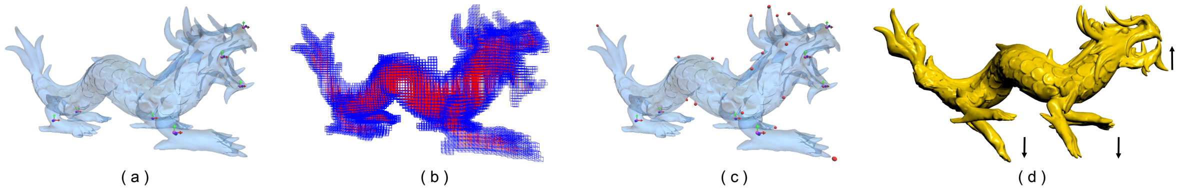

Figure 1: An illustration of C^2-Weighting: (a) user-specified interactive handles on the dragon model; (b) voxelization built to approximately compute intrinsic distances among points; (c) virtual handle insertion; (d) interactive manipulation by controlling the handles.

method for real-time skinning that belongs to the former research direction, which is inspired by an idea of weighting described in [5]. Given a model and prescribed handles, a computation of weight values on all points is executed to guide linear blending based deformation driven by handles. Our method has a good compatibility with different representations of models (e.g., meshes, point clouds, segments), guarantees a high computational efficiency with a parallelized implementation on GPU, and shows comparable visual quality with ARAP based methods.

2 Method

Our method employs linear blending strategy to update the position of every point $\mathbf{p}$ on a character model $\mathcal{M}$, which can be expressed uniformly as

$$\mathbf{p}' = \sum_i w_i(\mathbf{p})\mathbf{T}_i\mathbf{p}, \tag{1}$$

where $w_i(\mathbf{p})$ represents the weight of i-th handle on $\mathbf{p}$ and $\mathbf{T}_i$ stands for the transformation specified by user on that handle. For every user handle $\mathbf{h}_i$, a corresponding weighting distribution $\{w_i(\cdot)\}$ indicating the influence of this handle on all points needs to be computed. In this work, we extend the 2D weighting method in [5] to 3D, leading to an efficient, high-quality and parallelizable skinning method that can deal with different representations of 3D animation character models.

2.1 C^2-Weighting

The core idea of computing weighting in [5] is to decompose the whole domain $\mathcal{D}$ into small regions $\{\mathcal{R}_i\}$, each of which definitely and uniquely centers at the position of a handle $\mathbf{h}_i$, and calculate the weights $\{w_i(\cdot)\}$ of every point according to the regions it belongs to and estimated intrinsic distances to them (see Fig.1). Intrinsic distance $d(\cdot)$ refers to the shortest length of a interior path connecting two points, which is usually treated as shortest distance in a discrete sense. By exploiting this distance as the metric, Voronoi diagram sited at the origins of handles is constructed to retrieve a decomposition of input domain $\mathcal{D}$ into small regions. A virtual handle insertion algorithm is developed to add more necessary handle sites in order to ensure that every handle position is exclusive to its own region, leading to exact interpolation of transformations specified at each user handle. The transformations on virtual handles are decided by interpolating the translations and rotations of real handles on a series of harmonic fields dependent on a dual graph from the Voronoi diagram. Taking all real and virtual handles into account, the computation of weighting is formulated as

$$w_i(\mathbf{p}) = \frac{\phi_i(d(\mathbf{p}, \mathbf{h}_i)/r_i)}{\sum_{j=1}^{m} \phi_j(d(\mathbf{p}, \mathbf{h}_j)/r_j)}, \tag{2}$$

where $\phi_i(\cdot)$ and r denote the basis function and the size of a Voronoi cell respectively. As analyzed in [5], this weighting formulation holds many good properties such as C^2-continuity, interpolation, consistency, non-negativity etc., and can be seamlessly extended to 3D models. Moreover, it has the flexibility of handling any point based 3D models (see Section 2.2) and can be accelerated on GPU (see Section 2.3).

2.2 Model Compatibility

Our method is able to deal with various types of 3D character models that can be point set with well-defined connectivity information like triangle/volume mesh or points with partial or no connectivity such as point cloud, polygon soup, separate segments, etc. Given an input

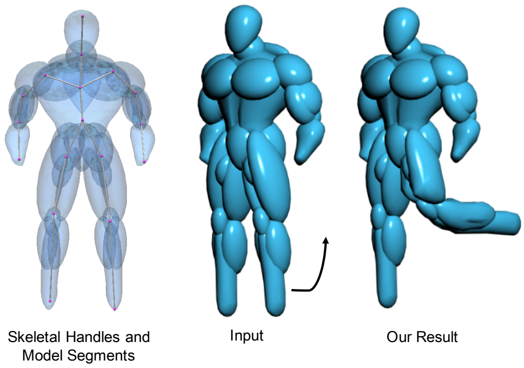

Figure 2: A series of skeletal handles are defined on the input human model which consists of separate segments. Prior methods that hold the assumption of manifoldness on input cannot deal with this model while our method successfully realizes interactive manipulation on it.

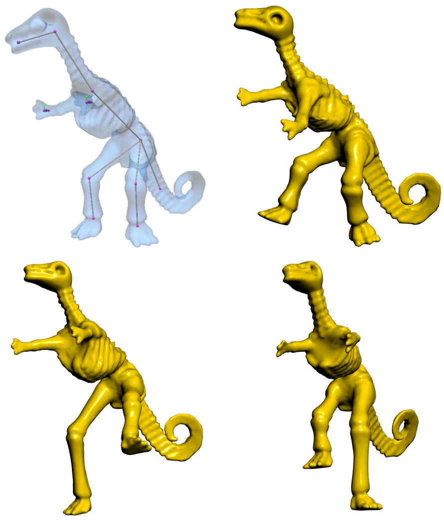

Figure 3: An example of interactive point and bar handles on Dino model.

model together with prescribed user handles, a voxelization is performed to facilitate the computation of approximate intrinsic distance that is very fundamental to the weighting, with which classic Dijkstra's algorithm is able to be utilized. Therefore, our skinning method realizes a low barrier on the requirements of initial models (see an example in Fig.2), bringing significant convenience to users.

2.3 GPU Powered Algorithm

According to the above two sections, our skinning algorithm mainly consists of 1. voxelization, 2. distance computation, 3. virtual handle insertion and Voronoi diagram construction, 4. weighting initialization, and 5. interactive manipulation. There are already plenty of researches on realizing parallel computation of step 1. Parallelizability of step 2, 4, and 5 is obvious since there exists no computational interference among points, i.e. the computation on each point solely depends on the information related with itself. It is worth mentioning that step 2-5 must be rerun when new handles are added as they will avoidably change the weighting. In applications where handles are predefined on models (e.g., skeletal handles on human models), step 1-4 can be treated as precomputation to be conducted only once.

3 Experimental Results

Implementation. Our implementation is built in C++ and CUDA on a PC with Intel Core i5-4460 3.20GHz, RAM 8 GB, and Nvidia Geforce GTX 760 2GB. For voxelization, an efficient parallel technique from [10] is adopted in our program. We generate animation sequence with standard functions provided in *Autodesk 3DS MAX* and key frames produced with our skinning method (see the supplementary video).

Interactive skinning with handles. A user-friendly interface is achieved in our experiment allowing users to freely add control handles on a model to be manipulated. As shown in Fig.3, users can place point handles and bar handles anywhere on Dino model and skinning results following the transformations specified on handles are computed in real time. Table.1 shows a summary of time cost of the examples shown in this paper.

Table 1: Time cost (ms): t_v - voxelization; t_h - virtual handle insertion; t_w - weight computation; t_u - each update.

	t_v	t_h	t_w	t_u
Dragon	200	1537	80	11
Human	205	2901	140	19
Dino	266	986	16	4
Armadillo	422	21968	94	16

Interestingly, a realization of rigid manipulation can be easily gotten with our method by simply fixing the lengths of skeletal handles. A comparison with other two methods is provided in Fig.4, showing that our method can generate similar output with better computational efficiency.

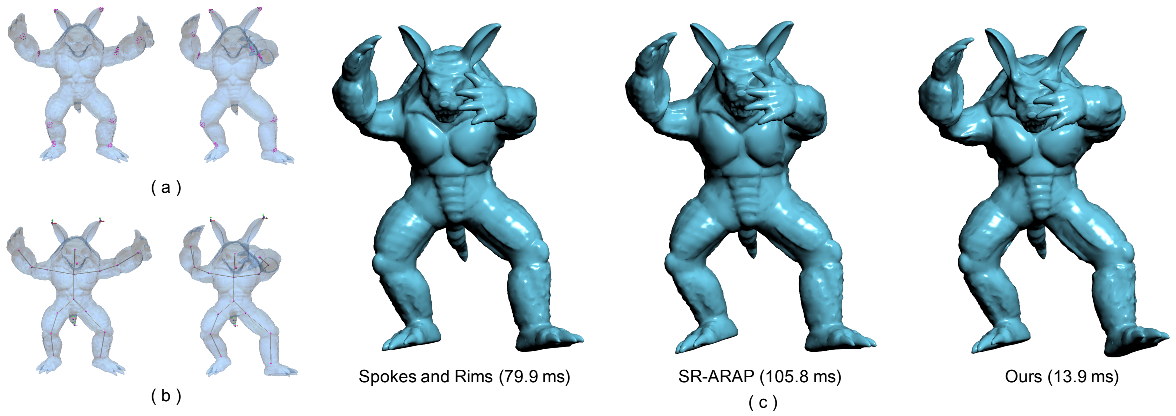

Figure 4: Comparison with Spokes and Rims method[6] and SR-ARAP method[7]: (a) Specified manipulation for the two compared methods; (b) predefined handles for our method; (c) results obtained from different methods together with the average time cost of each iterative update.

4 Conclusion & Future Work

We have proposed a GPU powered character skinning method which relies on a C^2-weighting technique to determine the transformations of points in a linear-blending manner. By taking advantage of voxelizing 3D models, our method demonstrates good compatibility of handling various types of input. Our GPU accelerated implementation proves its efficiency and effectiveness whose results can be used as key frames to generate animation sequences.

Currently, an obvious bottleneck is the step of virtual insertion and Voronoi diagram construction which slows down the overall efficiency of our method. We would like to seek the possibility of realizing this step in parallel to further boost its performance. We are also interested in combining our method with state of the art techniques (e.g., [4] gives optimized rotation centers of skinning) to enhance the skinning quality.

Acknowledgements

This work is partially supported by NSFC (No. 61300136, 61602222, U1611461), NSF of Guangdong Province (No. 2015A030313220, S2013020012795) and Open Project Program of the State Key Lab of CAD&CG (A1715), Zhejiang University.

References

[1] Robert W Sumner, Johannes Schmid, and Mark Pauly. Embedded deformation for shape manipulation. In *ACM Transactions on Graphics (TOG)*, volume 26, page 80. ACM, 2007.

[2] Alec Jacobson, Ilya Baran, Jovan Popović, and Olga Sorkine. Bounded biharmonic weights for real-time deformation. *ACM Transactions on Graphics (proceedings of ACM SIGGRAPH)*, 30(4):78:1–78:8, 2011.

[3] Zohar Levi and David Levin. Shape deformation via interior rbf. *IEEE transactions on visualization and computer graphics*, 20(7):1062–1075, 2014.

[4] Binh Huy Le and Jessica K. Hodgins. Real-time skeletal skinning with optimized centers of rotation. *ACM Transactions on Graphics (proceedings of ACM SIGGRAPH)*, 35(4):37:1–37:10, July 2016.

[5] Chuhua Xian, Shuo Jin, and Charlie C.L. Wang. Efficient C^2-weighting for image warping. *IEEE Computer Graphics and Applications*, in press.

[6] Isaac Chao, Ulrich Pinkall, Patrick Sanan, and Peter Schröder. A simple geometric model for elastic deformations. *ACM Trans. Graph.*, 29(4):38:1–38:6, July 2010.

[7] Zohar Levi and Craig Gotsman. Smooth rotation enhanced as-rigid-as-possible mesh animation. *IEEE transactions on visualization and computer graphics*, 21(2):264–277, 2015.

[8] Yu Wang, Alec Jacobson, Jernej Barbič, and Ladislav Kavan. Linear subspace design for real-time shape deformation. *ACM Transactions on Graphics (TOG)*, 34(4):57, 2015.

[9] Henry Schäfer, Benjamin Keinert, Matthias Nießner, and Marc Stamminger. Local painting and deformation of meshes on the gpu. In *Computer Graphics Forum*, volume 34, pages 26–35. Wiley Online Library, 2015.

[10] Gokul Varadhan, Shankar Krishnan, Young J. Kim, Suhas Diggavi, and Dinesh Manocha. Efficient max-norm distance computation and reliable voxelization. In *Proceedings of the 2003 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, SGP '03, pages 116–126, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association.