COMPUTER ANIMATION AND VIRTUAL WORLDS
Comp. Anim. Virtual Worlds (2009)
Published online in Wiley InterScience
(www.interscience.wiley.com) DOI: 10.1002/cav.313

Compatible quadrangulation by sketching

*By Chih-Yuan Yao, Hung-Kuo Chu, Tao Ju and Tong-Yee Lee**

Mesh quadrangulation has received increasing attention in the past decade. While previous works have mostly focused on producing a high quality quad mesh of a single model, the connectivity of the quadrangulation is typically difficult to control and varies among models even with similar shapes. In this paper, we propose a novel interactive framework for quadrangulating a set of models collectively with compatible connectivity. Furthermore, we demonstrate its application to 3D mesh morphing. In our approach, the user interactively sketches a skeleton within each model, and our method automatically computes compatible base domains for all models from these skeletons, on which the models are parameterized. With this novel parameterization, it is very easy to generate a pleasing and smooth 3D morphing sequence among these compatible models. The method yields quadrangulation with comparable quality to existing approaches, but greatly simplifies compatible re-meshing among a group of topologically equivalent models, in particular characters and animals models, with direct applications in shape blending and morphing. Copyright © 2009 John Wiley & Sons, Ltd.

Received: 24 March 2009; Accepted: 25 March 2009

KEY WORDS: quadrangulation; morphing; compatible; shape blending

Introduction

Mesh parameterization and remeshing are fundamental problems in digital geometry processing and have been extensively studied. In particular, mesh quadrangulation has attracted rising attention due to its many applications such as texture mapping, texture synthesis, CAD design, and fluid flow simulation, where a quadrilateral mesh is preferred over a triangular one.

Existing approaches have focused primarily on producing quality quad mesh of a single model. A number of methods create the quadrangulation by integrating two orthogonal directions derived from specific field functions on the surface such as the curvature tensor field[1,2] and harmonic function.[3,4] A different approach taken by other researchers is the use of a base complex as the parameterization domain. Guskov[5] proposes to setup a quad base domain manually followed by refinements to approximate the original surface. To simplify the manual process, Tarini[6] uses a collection of tightly stacked cubes (called *Polycubes*) to serve as the base domain. More recently, Dong[7] uses the Morse–Smale complex as the base domain and parameterize the mesh by solving a global system.

In applications such as morphing,[8–10] a prerequisite is that a correspondence needs to be established between the mesh elements in two or more models. In this scenario, it is ideal to represent models by meshes with a common connectivity that encodes the correspondence. Unfortunately, the above-mentioned methods either do not allow direct control over the connectivity of the resulting quadrangulation, or require non-trivial effort in manually setting up a common base domain consisting of quadrilaterals.[11] While a number of techniques have been proposed for compatible triangulations[12–14] that allow a user to specify patch layout or feature correspondences directly on the mesh surface, extending these techniques for morphing is non-trivial. In addition, it is less intuitive for a user to specify corresponding feature "points" on

*Correspondence to: T-Y Lee, National Cheng-Kung University, Tainan, Taiwan, ROC. E-mail: tonylee@mail.ncku.edu.tw

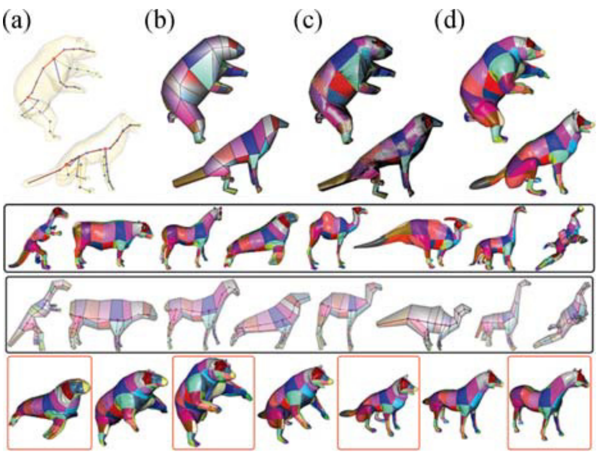

Figure 1. Overview of our interactive framework. Top: given several models, the user first creates skeletons with a common connectivity by sketching (a). Our algorithm then automatically constructs compatible quadrilateral base complexes, called poly-pipes (b). By parameterizing each model onto to its poly-pipe (c), all models are compatibly quadrangulated (d). Bottom: an army of models compatibly quadrangulated by our approach.

surface parts with less distinctive shapes, such as the back of the animal models at the bottom of Figure 1.

Contribution

In this paper, we propose a new approach for creating compatible quadrangulation among multiple models that utilizes simple and intuitive user inputs. Similar to the methods of Reference [5] and Reference [6], we ask the user to interactively build a base domain for parameterization. To simplify the manual effort, our key observation is that, for a large class of models (especially characters and animals), it is much easier to place skeleton lines inside the model than manually creating a 3D polyhedral base complex that conforms to the surface shape. As shown in Figure 1(a), the skeleton lines consist of a small number of vertices, correspond to the shape components of the model in an intuitive manner, and allows occlusion-free viewing.

The core of our method is a novel algorithm that takes a user-provided skeleton and automatically computes a base complex from the skeleton. This base complex, called the *poly-pipe*, consists of only quadrilateral faces and conforms to the shape and topology of the skeleton (Figure 1b). The model can then be parameterized and quadrangulated by mapping onto the complex (Figure 1c) using existing methods. Since the connectivity of the poly-pipe faces are determined by the skeleton, a poly-pipe can be made compatible with other poly-pipes constructed from skeletons with a same connectivity. As a result, compatible quadrangulation among multiple models can be easily achieved (Figure 1d), and we can easily generate quad-mesh morphing via compatible domains among these models, as seen in Figure 1 (bottom). Our method also achieves comparable quality with existing approaches when quadrangulating a single model, but further extends such quadrangulation compatibly onto multiple models.

Overview

Our method is composed of three main steps. First, the user provides a skeleton represented as a collection of line segments, which satisfies a number of constraints to ensure successful construction of the domain complex (Section "Sketching"). Secondly, the skeleton is "inflated" using our algorithm into a quadrilateral poly-pipe, while ensuring that the poly-pipes constructed from a group of skeletons share the same face connectivity (Section "Poly-pipe construction"). Lastly, each model is paramterized by its poly-pipe and (compatible) quadrangulation is performed.

Sketching

The input to our method is a curve skeleton sketched by the user. We utilize the interactive sketching interface in Ju,[15] which takes the advantage of a layered visualization of the model to allow even complicated skeletons to be drawn on a single view of the model. For character models, this initial process normally takes less than a minute.

The resulting skeleton provided by sketching consists of a network of line segments, as shown in Figure 2(a). We refer to these segments as *arcs* and their the junctions as *nodes*, and we distinguish between nodes of different valences (colored in Figure 2a). To facilitate the construction of poly-pipe, we additionally place a few constraints on this skeleton, which can be easily enforced during sketching. First, we require the skeleton to have the same topology as the model. That is, the skeleton will contain one ring of arcs for each topological handle on the model. Second, for each node of valence of 3 or more, we ask the user to indicate two incident arcs as the *axis* of this node (colored red in Figure 2a). The axis will be used to determine the shape of the base domain at the

DOI: 10.1002/cav

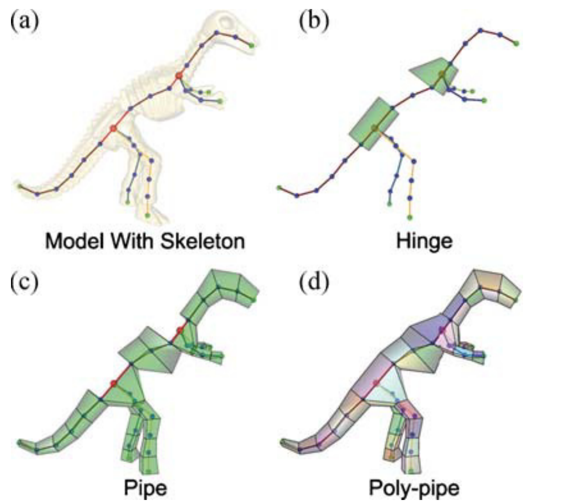

Figure 2. Constructing a poly-pipe (d) from a skeleton (a) by inflating high-valence nodes into hinges (b) and the rest of the skeleton into pipes (c). In the sketched skeleton, nodes with valence 1, 2 and more are colored in green, blue, and red, respectively, and the arcs defining the axis at each node are colored red.

node. Third, to ensure a closed base domain (see next), the system will automatically add a new node between two valence-3 nodes sharing a common arc by splitting the arc from the middle. Lastly, given a number of models that the user wishes to compatibly quadrangulate, the skeleton of each model must have the same edge-arc connectivity.

Poly-pipe Construction

The poly-pipe is constructed by building a system of pipes along the skeleton arcs, or intuitively, inflating the thin skeleton into a "fat" body. For the parameterization purpose, the poly-pipe needs to preserve the topology of the skeleton (and hence the topology of the model) while consisting of only quad faces. To simultaneously fulfill these requirements, we adopt a divide-and-conquer approach, where pieces of the poly-pipe will be constructed separately based on the skeleton structure and connected together afterwards. In particular, we will construct one *pipe* for each path of skeleton arcs bounded between two nodes of valence 1 or 2, and one *hinge* for each node of valence 3 or more. A neighboring pipe and hinge will share a common quad face at one end of the pipe. As such, all pipes and hinges will connect seamlessly into a complex with only quad faces and has the same topology of the skeleton (see Figure 2b–d).

Below, we detail the construction of these two sub-complexes, pipes and hinges. There are two main challenges that have to be resolved. Topologically, while it is trivial to construct a pipe with quad cross-section, it is more difficult to construct a quad-only hinge polyhedron at a skeleton node with possibly many outgoing arcs. In addition, geometrically, the faces of the pipes and hinges need to have relatively low distortion to serve as the base domain for parameterization.

Pipe Construction

As defined above, each pipe is constructed from a path of skeleton arcs ended at two nodes with valence 1 or 2. The pipe can be built by connecting a unit square located at each node whose normal is aligned with the tangent direction of the incident arcs. To compute the rotation of the square around its normal while minimizing twisting of the pipe, we adopt a similar idea to the sweeping surface[16,17] method, where the rotation of the squares in the interior of the pipe is propagated from those at the ends of the pipe. Note that if one (or both) end of the skeleton path is adjacent to a node with valence 3 or more, the square face at that end of the pipe will be replaced by the square located at the connecting quad face of the hinge. Hence we have two cases to consider, when either one or both ends of the pipe have given square rotations. If neither end has a fixed rotation, we arbitrarily pick one end and fix the rotation of that square.

If one end face of the pipe has a fixed orientation, we denote the two axes of that square face as u_0, v_0 and its normal as t_0 (see Figure 3b). To compute the axes of the square cross-section as the next node with tangent direction t_1, we rotate u_0, v_0 around the vector $t_0 \times t_1$ by the angle between t_0 and t_1. The orientation can thus be propagated throughout the pipe. An example is shown in Figure 3(b) and compared to an otherwise twisted configuration in (a). If both ends of the pipe have fixed orientations, we will propagate the orientation from each end and obtain the final orientation of the cross-section square at a node as a weighted combination of the orientation computed from both ends, where the weight is proportional to the distance from the node to that end. An example is shown in Figure 3(c).

Finally, to conform to the geometry of the mesh, we project each vertex of the square cross-sections radially from the square centers onto the model surface.

DOI: 10.1002/cav

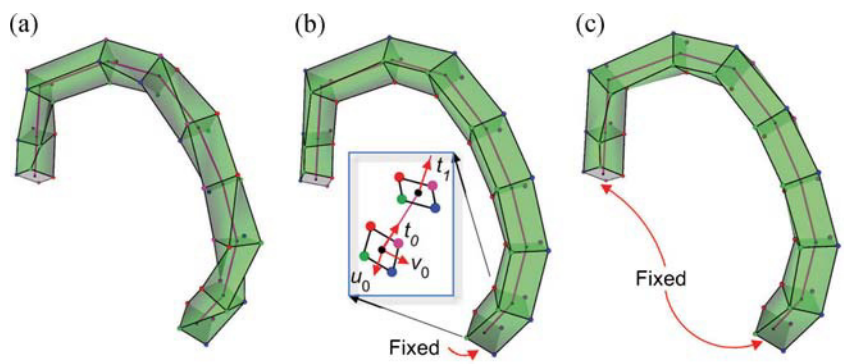

Figure 3. (a) A pipe with severe twisting. (b) A pipe constructed using our approach with orientation fixed at one end. (c) A pipe constructed with orientation fixed at both ends.

Hinge Construction

At each skeleton node with valence 3 or more, we wish to build a polyhedron consisting of only quad faces. To connect to neighboring pipes, each outgoing arc from the node will uniquely correspond to a face on this polyhedron. While some methods have been proposed in the past, most recently in Reference [18], to generate such polyhedron at a branching node, the generation either requires direct user-interaction or is based on pre-defined templates in a limited set of cases. Here we present a fully automatic approach that is applicable to a node with arbitrarily high valence and branching geometry.

The key idea of our algorithm is to perform subdivision from an initial polyhedron which consists of quad faces and is located at the skeleton node (Figure 4a). Each subdivision is performed on a quad face and splits the quad into five smaller quads (Figure 4b). The subdivision terminates when each quad face intersects with at most one incident arc. To this end, we devide a subdivision algorithm that is guaranteed to terminate and produce a well-shaped quadrilateral hinge. Below, we detail the construction of the initial polyhedron and the subdivision rules.

The Initial Polyhedron. Recall that at each skeleton node with valence 3 or more, two incident arcs are specified by the user to form an axis. To start, we place a cuboid at the node aligned with the axis, whose end squares are placed at the two neighboring nodes (light blue dots in Figure 4a). Just as in the pipe construction, we need to determine the rotation of the cuboid around this axis. While various criteria can be used here to define an optimal rotation, we found that aligning a side face of

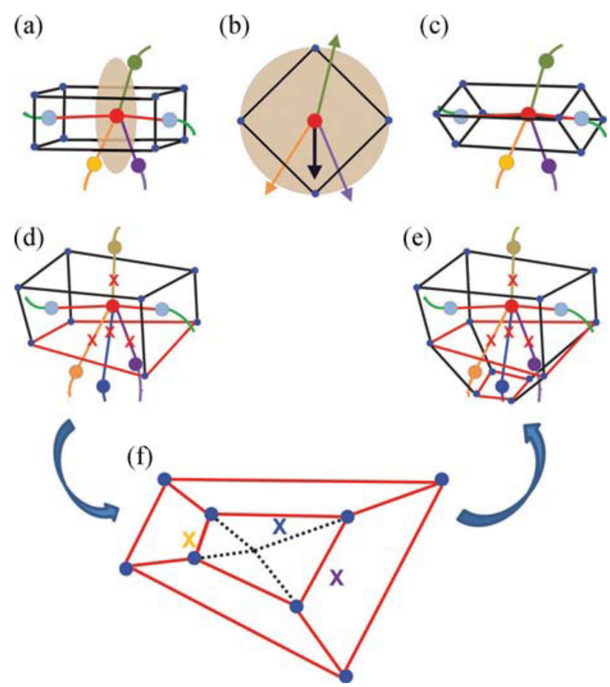

Figure 4. Construction of the initial hinge polyhedron. (a) A cuboid with the fixed axis (red) and the orthogonal plane (tan). The colored dots indicate the neighboring nodes along each arc. (b) Projected arcs on the plane and computed principle direction (black arrow). (c) Rotated cuboid. Subdivision of a quad face of a hinge (d) (red color) intersecting with more than one skeletal arcs into five quads (e), and finally projected hinge (f).

the cuboid with the direction of the non-axis incident arcs yields well-shaped hinges. To do so, we project the arcs onto the plane orthogonal to the fixed axis, and compute

DOI: 10.1002/cav

the principle direction of variation of the projected arcs using PCA (Figure 4b). The cuboid is rotated so that one of its projected diagonal axis is aligned with that principle direction (Figure 4b,c).

Quintic Subdivision. On each face of the initial cuboid where there are more than one skeleton arc intersecting (Figure 4d), the face is subdivided. To reduce the number of subdivisions that have to be performed, we prefer a way of subdividing the face so that the intersections with the skeletal arcs are uniformly distributed, that is, when the maximum number of intersecting points within each subdivided face is minimized. To do so, in the quintic subdivision shown in Figure 4(e), we place the four vertices of the interior quad at proportion $h \in (0, 1)$ along the line connecting the centroid of the face to the vertices of the original quad. To find the optimal choice of h, it suffices to check for those h that yield different distributions of the number of intersections on the subdivided faces.

Subdivision is performed repeatedly to any quad face that have more than one intersections with the outgoing skeleton arcs. To show that such subdivision always terminates, we observe that the the total number of intersections on a quad face is reduced by at least half after each subdivision following the above procedure.

To adjust the hinge shape to the actual geometry, as in pipe construction, we project the hinge vertices to the actual geometry. For the initial cuboid, we project the two end faces (situated at two neighboring nodes) radially from their centers. For vertices created during subdivision, they are first placed at the corners of a unit square situated at the nodes along the intersecting arcs, and projected radially from their centers. An example result is shown in Figure 4(f).

Compatible Hinge Construction. Note that the face structure of the hinge produced by quintic subdivision depends not only on the number of arcs at a skeleton node but also the direction of those arcs. To ensure that the face structure is the same among the poly-pipes constructed for a group of models with different shapes, we simply ask the user to select one of the models as a reference model and adopt the face structure of the hinges on the reference poly-pipe for all their poly-pipes (assuming the skeletons have the same connectivity and the correspondence between the edges and arcs in different skeletons are provided by the sketching interface).

Parameterization and Quadrangulation

Once a poly-pipe has been constructed, we partition the surface into quad patches by tracing boundaries[12–14] of the patches based on the connectivity of poly-pipe. To parameterize the surface over the patches, one can either solve a global linear system[19] or employ an iterative local parameterization method.[13,14] In our method, we adopt the latter approach to obtain a smooth and bijective parameterization. To obtain a semi-regular quadrilateral mesh, we uniformly sample each quad patch in its parametric 2D domain.

Results

We first demonstrate our method in quadrangulating a single model. The use of a user-provided sketch makes it easy to create base domains that may have a non-trivial shape or topology, as shown in Figure 5(a). The quality of quadrangulation is reported in Table 1 using the L^2 stretch metric.[20] The ideal L^2 value is equal to 1. The literature typically achieve L^2 values on similar models in the range of 1.2–0.7. We compare our method with the recently developed method by Reference [2] using the three models shown in Figure 5(b), and the result is reported in Table 2. We observe that our method achieves comparable quality in quadrangulation.

The primary benefit of our method is that a user can easily create compatible quad re-meshing of a

Figure 5. (a) Constructed poly-pipes and semi-regular quadrilateral meshes of four models with non-zero genus. (b) Models used in comparison with Ray et al. [2].

DOI: 10.1002/cav

Model	No. of vert.	Genus	Ext.	L^2 stretch
Pegaso	57 336	5	66	0.89
Elephant	40 956	3	60	0.90
Dancing children	55 794	8	80	0.90
Neptune	63 484	3	75	0.93

Table 1. The number of extra ordinary points and L^2 stretch value for models in Figure 5

	Poly-pipe	Ray *et al.* [2]
Feline	0.898	1.070
Bunny	1.078	1.029
Lion	0.930	1.123

Table 2. Quality comparison with Ray *et al.* [2]

large group of models with equivalent topology. We demonstrate this capability using 10 genus-0 models of four-footed and two-footed animals and humanoids in Figure 1, whose poly-pipes are shown in Figure 1. We further show three compatibly re-meshed high-genus models in Figure 7. Note that these three models are only topologically equivalent and have no semantically meaningful correspondences, which would be difficult for previous approaches that require surface-feature points as user input. We further compare our approach to the method of Fan[11] that utilizes Poly-cubes for

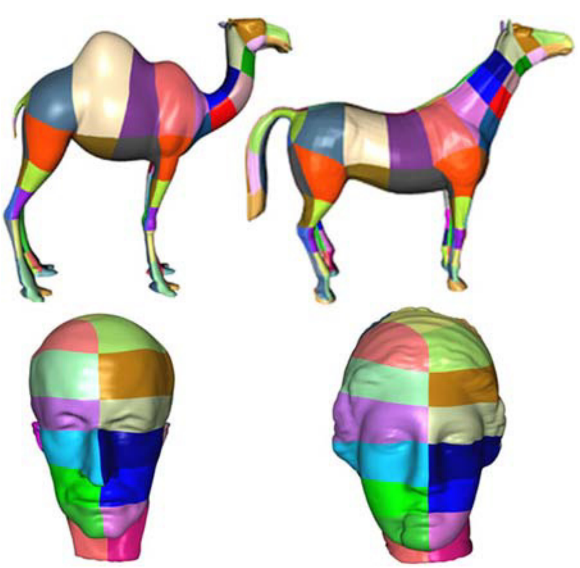

Figure 6. Two examples of compatible quadrangulations.

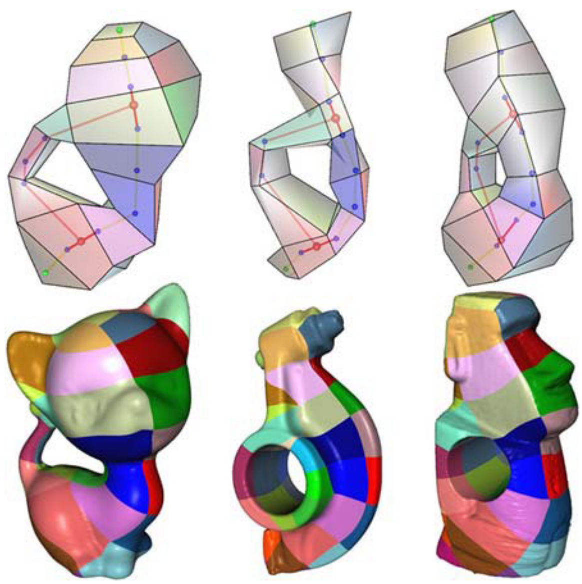

Figure 7. Poly-pipes and compatible quadrangulations of three genus-1 models.

compatible quadrangulation using the two examples in Figure 6 (Figure 7 in Reference [11]). Observe in Table 3 that, besides the ease of setting up the base domain, our method greatly improves the L^2 metric in these two examples.

As an application of compatible quadrangulation, we can compute shape morphing between quadrangulated models using the computed correspondence. Note that, since the base domain in our method is setup from hand-drawn skeletons interior to the model, the quadrangulation is not explicitly related to surface features (such as ears). As a result, the computed correspondence may not be ideal on places (such as the face) where surface features are expected to be matched. We can easily remedy this limitation by allowing the user to adjust the poly-pipe vertices and place them on the desired surface features. Figure 8 shows an example of a linear blending between a pig and a wolf before (a) and after (b) moving a poly-pipe vertex to the tip of the ear. Note that without such adjustment, the blended shape exhibits redundant ears due to incorrect

	Poly-pipe	Fan *et al.* [11]
Camel/horse	0.93	0.79
Maxplanck/venus	0.95	0.85

Table 3. Quality comparison with Fan *et al.* [11]

DOI: 10.1002/cav

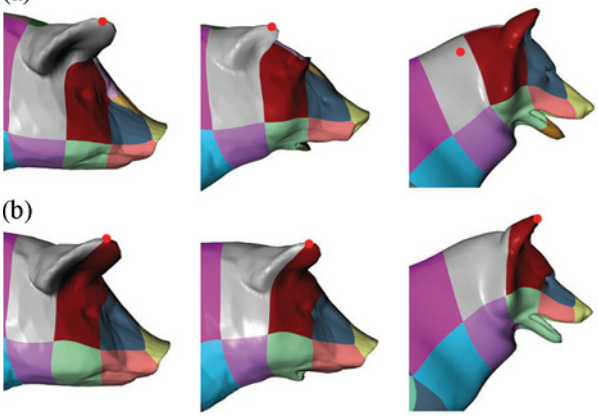

Figure 8. Linearly Blended shape between pig and wolf before (a) and after (b) adjusting a poly-pipe vertex (red sphere).

correspondence, as seen in Figure 8(a) middle. We demonstrate more linear morphing results in Figure 1 and in the accompanied video.

Conclusion and Discussions

We presented a novel interactive method for quadrangulating 3D models. The method utilizes a convenient and intuitive form of user input—sketching—to easily produce high quality, compatible quadrangulation among multiple models. The key component of our method is a novel algorithm that constructs a quadrilateral base domain automatically from a user-provided curve skeleton.

The current method has a number of limitations that we seek to address in the future. First, the quality of parameterization relies on the quality of the poly-pipe, which in turn depends on the user-provided skeleton. Although we found it is easy to create a skeleton that would yield a satisfactory quadrangulation, poor result may be generated from an incomplete skeleton. For example, Figure 9 compares the result without (a) and with (b) a tail segment in the skeleton of a dinosaur, and observe that the former results in a parameterization with large distortion at the tail. One possible way to ensure a complete skeleton is to use an automatically generated skeleton by recent methods such as References [21–23], and use this as a guide during user sketching. Second, the poly-pipe structure is designed for curve skeletons and suitable for quadrangulating shapes consisting of mostly tube-like parts, such as characters. For models with a plate-like geometry, such as the hand in Figure 9(c), poly-pipe would yield less satisfactory parameterization. To this end, we would like to investigate more general ways to construct base complexes from not only skeleton curves but also surfaces.

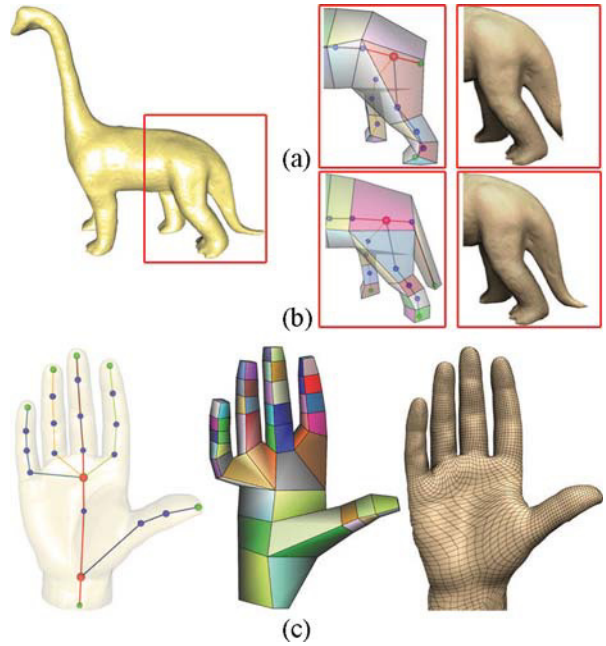

Figure 9. (a) Skeleton with missing tail part will produce poor-shaped poly-pipe and re-meshing, (b) which can be resolved by adding the tail segment. (c) A hand model with plate-like geometry (the palm) cannot be well parameterized using our curve-skeleton-based poly-pipes.

ACKNOWLEDGEMENT

We thank anonymous reviewers' helpful comments to improve this paper. This work is supported in part by the Landmark Program of the NCKU Top University Project (Contract B0008), the National Science Council (Contracts NSC-97-2628-E-006-125-MY3 and NSC-96-2628-E-006-200-MY3), Taiwan, Republic of China. This work is supported in part by NSF grant CCF-0702662.

References

1. Kälberer F, Nieser M, Polthier K. Quadcover—surface parameterization using branched coverings. *Computer Graphics Forum* 2007; **26**(3): 375–384.
2. Ray N, Li WC, Lévy B, Sheffer A, Alliez P. Periodic global parameterization. *ACM Transactions on Graphics* 2006; **25**(4): 1460–1485.

3. Dong S, Kircher S, Garland M. Harmonic functions for quadrilateral remeshing of arbitrary manifolds. *Computer Aided Geometric Design* 2005; **22**(5): 392–423.
4. Tong Y, Alliez P, Cohen-Steiner D, Desbrun M. Designing quadrangulations with discrete harmonic forms. In *Symposium on Geometry Processing*, 2006; 201–210.
5. Guskov I, Khodakovsky A, Schröder P, Sweldens W. Hybrid meshes: multiresolution using regular and irregular refinement. In *Symposium on Computational Geometry*, 2002; 264–272.
6. Tarini M, Hormann K, Cignoni P, Montani C. Polycube-maps. *ACM Transactions on Graphics* 2004; **23**(3): 853–860.
7. Dong S, Bremer P-T, Garland M, Pascucci V, Hart JC. Spectral surface quadrangulation. *ACM Transactions on Graphics* 2006; **25**(3): 1057–1066.
8. Lin C-H, Lee T-Y. Metamorphosis of 3d polyhedral models using progressive connectivity transformations. *IEEE Transactions on Visualization and Computer Graphics* 2005; **10**: 2–12.
9. Lin C-H, Lee T-Y, Chu H-K, Yao C-Y. Progressive mesh metamorphosis: animating geometrical models. *Computer Animation and Virtual Worlds* 2005; **16**(3–4): 487–498.
10. Lee T-Y, Yao C-Y, Chu H-K, Tai M-J, Chen C-C. Generating genus-n-to-m mesh morphing using spherical parameterization: research articles. *Computer Animation and Virtual Worlds* 2006; **17**(3–4): 433–443.
11. Fan Z, Jin X, Feng J, Sun H. Mesh morphing using polycube-based cross-parameterization: animating geometrical models. *Computer Animation and Virtual Worlds* 2005; **16**(3–4): 499–508.
12. Praun E, Sweldens W, Schröder P. Consistent mesh parameterizations. In *Proceedings of ACM SIGGRAPH 2001*, 2001; 179–184.
13. Kraevoy V, Sheffer A. Cross-parameterization and compatible remeshing of 3D models. *ACM Transactions on Graphics* 2004; **23**(3): 861–869.
14. Schreiner J, Asirvatham A, Praun E, Hoppe H. Inter-surface mapping. *ACM Transactions on Graphics* 2004; **23**(3): 870–877.
15. Ju T, Zhou Q-Y, Hu S-M. Editing the topology of 3D models by sketching. *ACM Transactions on Graphics* 2007; **26**(3): 42–51.
16. Hyun D-E, Yoon S-H, Chang J-W, Seong J-K, Kim M-S, Jüttler B. Sweep-based human deformation. *The Visual Computer* 2005; **21**(8–10): 542–550.
17. Yoon S-H, Kim M-S. Sweep-based freeform deformations. *Computer Graphics Forum* 2006; **25**(3): 487–496.
18. Zhang Y, Bazilevs Y, Goswami S, Bajaj CL, Hughes TJR. Patient-specific vascular nurbs modeling for isogeometric analysis of blood flow. *Computer Methods in Applied Mechanics and Engineering* 2007; **196**(29–30): 2943–2959.
19. Khodakovsky A, Litke N, Schröder P. Globally smooth parameterizations with low distortion. *ACM Transactions on Graphics* 2003; **22**(3): 350–357.
20. Sander PV, Snyder J, Gortler SJ, Hoppe H. Texture mapping progressive meshes. In *Proceedings of ACM SIGGRAPH 2001*, 2001; 409–416.
21. Kin-Chung Au O, Tai C-L, Chu H-K, Cohen-Or D, Lee T-Y. Skeleton extraction by mesh contraction. *ACM Transactions on Graphics* 2008; **27**(3): 1–10.
22. Wang Y-S, Lee T-Y. Curve-skeleton extraction using iterative least squares optimization. *IEEE Transactions on Visualization and Computer Graphics* 2008; **14**(4): 926–936.
23. Pascucci V, Scorzelli G, Bremer P-T, Mascarenhas A. Robust on-line computation of Reeb graphs: simplicity and speed. *ACM Transactions on Graphics* 2007; **26**(3): 58–67.

Authors' biographies:

Chih-Yuan Yao was born in Tainan, Taiwan in 1980. He received his BS and MS degree in computer engineering from National Cheng-Kung University, Taiwan, in 2002 and 2003, respectively. He is currently working toward his PhD degree at the Department of Computer Science and Information Engineering, National Cheng-Kung University. His research interests is computer graphics.

Hung-Kuo Chu received the BS degree in computer science/engineering from National Cheng-Kung University, Tainan, Taiwan, in 2003. Now, he is pursuing his PhD degree at Department of Computer Science and Information Engineering, National Cheng-Kung University. His research interests include geometry modeling, mesh deformation and perception analysis.

Tao Ju is an Assistant Professor in the Department of Computer Science and Engineering at the Washington University in St. Louis (USA). He obtained his MS and PhD degrees in Computer Science at Rice University in 2005. He conducts research in computer graphics and bio-medical applications, and is particularly interested in geometric modeling and processing.

DOI: 10.1002/cav

Tong-Yee Lee received the PhD degree in computer engineering from Washington State University, Pullman, in May 1995. He is currently a Distinguished Professor in the Department of Computer Science and Information Engineering, National Cheng-Kung University, Tainan, Taiwan, ROC. He leads the Computer Graphics Group, Visual System Laboratory, National Cheng-Kung University (http://graphics.csie.ncku.edu.tw/). His current research interests include computer graphics, non-photorealistic rendering, image- based rendering, visualization, virtual reality, surgical simulation, medical visualization and medical system, and distributed and collaborative virtual environments.

DOI: 10.1002/cav