SPINE-BASED DEFORMATION WITH LOCAL VOLUME PRESERVATION

A Dissertation
Presented to
The Academic Faculty

by

Wei Zhuo

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in Computer Science in the
School of Interactive Computing

Georgia Institute of Technology
December, 2014

SPINE-BASED DEFORMATION WITH LOCAL VOLUME PRESERVATION

Approved by:

Professor Jarek Rossignac,
Committee Chair
School of Interactive Computing
Georgia Institute of Technology

Professor Jarek Rossignac, Advisor
School of Interactive Computing
Georgia Institute of Technology

Professor Stefanie Hahmann
Applied Mathematics
Grenoble Institute of Technology, INRIA Research

Professor Karen Liu
School of Interactive Computing
Georgia Institute of Technology

Professor Greg Turk
School of Interactive Computing
Georgia Institute of Technology

Professor George M. Turkiyyah
Department of Computer Science
American University of Beirut, KAUST

Date Approved: July 30, 2014

DEDICATION

To my husband,

my parents,

and my grandparents.

PREFACE

This dissertation is an original intellectual product of the author's doctoral work.

Two images, Figure 3 and Figure 4 in Chapter 1, are intended to illustrate potential applications. Their contents have been selected from results returned by public image search tools with relevant keywords. Chapter 2 is a literature survey which presents images from related published work with proper references and citations. The rest of the images in this dissertation are produced by programs written for research projects that fund the author's doctoral study.

Portions of the dissertation, or preliminary mathematical models and experiments on deformation driven with non-stretchable spine curve, have been published as *Wei Zhuo and Jarek Rossignac, "Fleshing: Spine-driven Bending with Local Volume Preservation", Computer Graphics Forum (CGF), VOL. 32, NO. 2, 2013* and *Wei Zhuo and Jarek Rossignac, "Curvature-based Offset Distance: Implementation and Applications", Computer & Graphics, VOL. 36, NO. 5, 2012.* These two publications are also presented at Eurographics (EG 2013) and Shape Modeling International (SMI 2012).

This work was supported in part by NSF Grant Number 0811485. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of NSF.

ACKNOWLEDGEMENTS

I gratefully acknowledge the support and encouragement provided by my advisor, Professor Jarek Rossignac, during the time I spent at Georgia Tech. I feel extremely fortunate to have crossed path with him and to have the opportunity to work with him so closely for many years. He has been a most valued mentor and friend during tough times. He has cultivated me with his fantastic taste of research while allowing me the freedom of pursuing problems I found most important and interesting; at the same time, in critical moments, he has always been available to provide sincere and determined opinions, helping me make the right decisions.

I would like to acknowledge the feedback received from members of my thesis committee: Professor Karen Liu, Professor Stefanie Hahmann, Professor Greg Turk and Professor George M. Turkiyyah. I would like to thank them for interesting discussions and inspiring suggestions on my research which helped me look at my work in a broader context.

The friendship, companionship and support of my colleagues from the geometry research group in the School of Interactive Computing would be hard to replace. A big thanks to all my collaborators at Lawrence Berkeley National Lab, IBM T.J. Waston Research Center, and the School of Computational Science Engineering at Georgia Tech.

Finally, I would like to dedicate this work to my husband and my parents, whose love and patience have been my constant source of inspiration, without which this work would have been impossible. My achievement is also theirs.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

In shape modeling applications, *deformation* is the process of applying a continuous, non-affine transformation to a shape. The definition of the deformation should be independent of the representation of the shape. In practice, the shape is often represented by its boundary, which is defined by a set of vertices and by connectivity information. The transformation is often applied to these points.

A deformation algorithm takes the original shape and designer's choices as inputs, and outputs the deformed shape. This dissertation dedicates to introducing *spine-based deformation*: Any distortion to the shape is controlled by a low dimensional proxy, which is a spine curve or surface. Considering a sometimes important constraint to preserve the shape's volume during deformation, this thesis addresses a suite of problems in spine-based deformation with local volume preservation, meaning that the volume of any subset of the shape is preserved. Although our deformation model may be applied to the control points or vertices of a surface model that is not a water tight boundary of a solid, in this thesis, the term shape will refer to a solid model which has a clearly defined interior and volume. Previously proposed local or global volume compensation techniques are typically based on iterations that introduce a complexity bilinear in the numbers of vertices and iterations. we present a family of closed-form solutions for shape deformation with mathematically exact local volume preservation, and demonstrate their power in the context of interactive bending, rotating, sliding or stretching a 2D or 3D shape. The overall complexity is linear in the number of vertices.

Proposed spine-based deformation framework adopts the following assumptions in geometric modeling:

- When the spine is a curve, a plane normal to the spine curve remains normal to the spine curve after deformation. The parameter associated with the point at which the plane intersects the curve is unchanged.

- When the spine is a surface, a line normal to the spine surface remains normal to the spine surface after deformation. The parameters associated with the point at which the line intersecting the plane remain unchanged.

With these assumptions, we compute the closed-form formulation for the deformation that guarantees local volume preservation and is expressed using real roots of low degree polynomials and simple point and vector expressions. Due to its simplicity, our solution may be used to deform complex models in realtime during interactive manipulation or animation, where the behavior of the spine has been designed or is computed in realtime through simulation.

CHAPTER I

INTRODUCTION

Before discussing related work and introducing theoretical or technical challenges, we first describe what is the spine-based deformation. We have mentioned that the change of the shape is due to manipulating operations such as bending or stretching of a lower dimensional proxy which we called the *spine*. However, in order to let the proximal spine's movements determine an object's deformation, the object needs to be *registered* to the spine at first. Also, after deforming the spine we need to decide how to *reconstruct* the object from the proximal spine and registration parameters. This chapter provides the fundamental framework and describes a list of assumptions in spine-based deformations.

1.1 What is spine-based deformation?

In shape modeling, the term 'deformation' is used to describe the process of applying a continuous, spatially-varying transformation to an object. Our focus is on a specific type of deformations defined by bending or stretching a spine curve or surface, that pierces or lies near the shape. For instance, one may want to interactively stretch or bend a shape by stretching or bending a proximal curve through control point manipulation. Likewise, one may use a spine surface as the proxy to control the deformation. More details of the application scenarios are given in Section 1.4.

As shown in Figure 1 which provides an overview of the steps in spine-based deformation, the designer starts with a shape S_0, and specifies an initial version C_0 of the spine, which needs to be a smooth curve that may pierce S_0 or not. Then the designer specifies the new positions or time-evolution of the control points of the spine, such that the spine C changes from C_0 to C_1. Then spine-based deformation

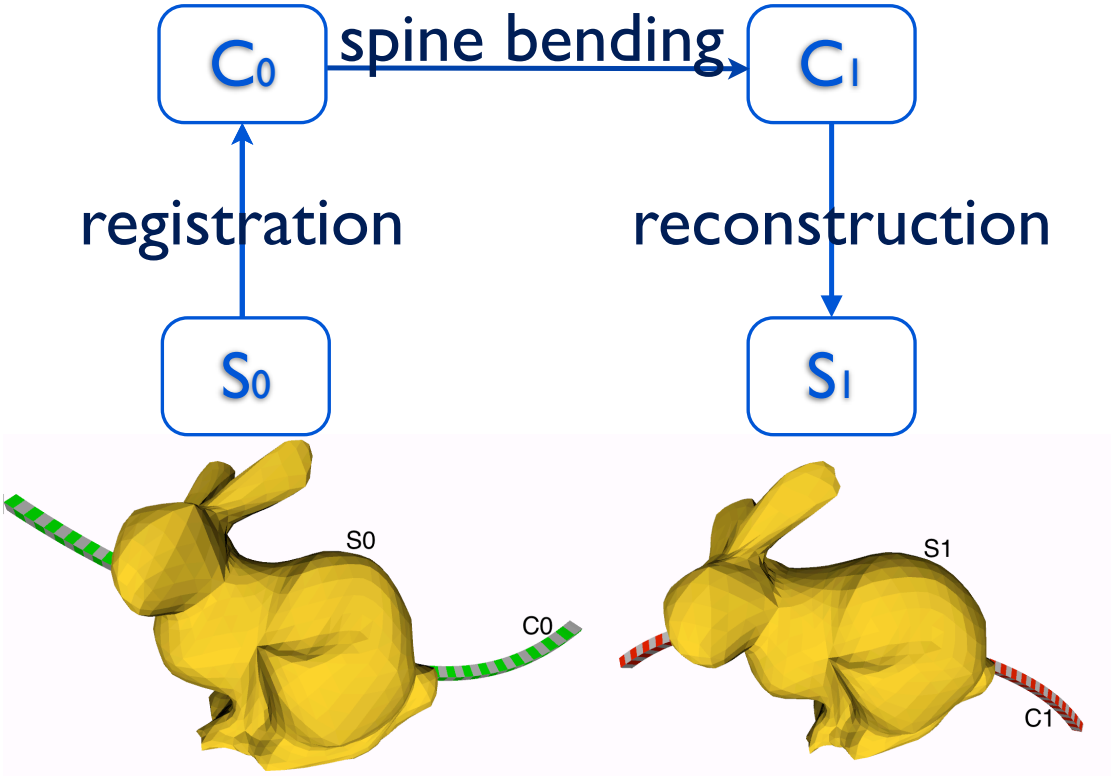


Figure 1: Overview of the steps in spine-based deformation.

algorithm computes the current position of each vertex of the shape and displays the resulting deformed shape S_1. Notice that the deformation calculation can be easily parallelized for each vertex of the shape, which makes it possible to accelerate using GPU.

1.2 Assumptions

Before describing the contributions made for this type of deformation, we introduce the most important assumptions in order for this framework to work for spine curves and spine surfaces respectively.

Assumption I: Cross-section preserving

In the deformation driven by a spine curve, a *cross-section* is a collection of points of the shape registered to the same point on the spine curve. Assume that $C_0(s)$ is a point on the initial version C_0 of the spine curve. Here s is the parameter along the spine curve. The registration step associates the parameter s with a point P_0 of S_0, such that $C_0(s)$ is the closest projection of P_0 onto on to the spine curve C_0. All points of S_0 associated with a particular parameter s of are called a cross-section, which is formally defined as the point set $\{P_0, P_0 \in S_0, \arg\min dis(C_0(s), P_0) = s\}$. Note that each cross-section is planar.

One important assumption in deformation driven by a spine curve is cross-section preserving. During the deformation, points on the solid of the same cross-section will remain in the cross-section associated with the same parameter s. Note that the shapes of the initial and deformed cross-sections may be different, but they are both planar. Therefore, the deformation driven by a spine curve has the following assumptions: First, planes normal to the curve, or cross-sections remain normal to the curve after deformation. Second, the parameter s of any cross-section remain the same during bending.

Often, the spine curve may represent the central axis of an elongated object. The

designer defines the initial spine maybe by placing a few control points for it. Then she defines its deformation over time, maybe by specifying a few key positions for each control point that will be interpolated by the motion of that control point. For each time t, the deformation algorithm computes the current position of each vertex of the solid and displays the resulting triangulated surface.

Assumption II: Normal segment preserving

Similarly, in deformation controlled by a spine surface, the spine may represent the medial surface of a flat object. The designer defines the initial spine surface by placing a few control points for it. And then she specifies its deformation over time, either by control point manipulation or scripted animation. At each time frame, the deformation algorithm computes the current position of each vertex of the shape and displays the result.

One important assumption in deformation driven by a spine surface is normal segment preserving. Specifically, assume that $C_0(u, v)$ is a point on the initial spine surface. Here we need two parameters u, v for surface parameterization. We associate the parameters u, v with a point P_0 of S_0 such that $C_0(u, v)$ is the closest projection of P_0 onto C_0. All points of S_0 associated with a set particular parameters u, v form a line segment normal to C_0 at $C_0(u, v)$. During the deformation, points of a line segment will remain in the line associated with the same surface parameters.

Formally speaking, all points of S_0 associated with a particular pair of parameters u, v on C_0 are called a normal segment, which is formally defined as the point set $\{P_0, P_0 \in S_0, \arg\min dis(C_0(u, v), P_0) = (u, v)\}$. Note that each normal segment is one-dimensional. All points of a normal segment remain within one normal segment after the deformation with the same pair of parameters.

Implications

With Assumption I or II, the registration parameter s, or (u, v) remains invariant during transformation. This implicates that only the offset value from the spine is

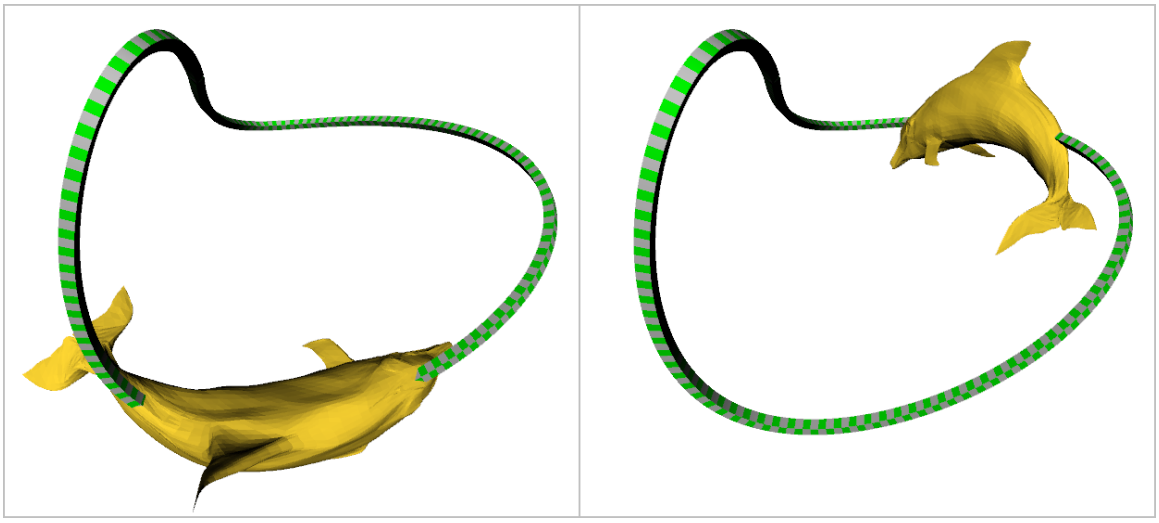


Figure 2: Sliding a dolphin along a 3D curve by adjusting the registration parameter globally.

allowed to change during the deformation. Note that Assumption II is a reasonable assumption in classical mechanics on thin plate bending [43]: straight lines normal to the base surface remain straight and normal to the base surface after deformation. We extend this assumption to the deformation driven by a 3D curve: cross sections normal to the spine curve remain planar and normal to the spine curve after the deformation, and arrive at Assumption I. Note that these assumptions are not strict restrictions on the possible effects in spine-driven deformation: the registration parameter s or (u, v) is considered invariant only when deriving the offset value for local volume preservation. While effects such as sliding along the spine is still possible to create by programming the global change (See Figure).

1.3 Key contributions

With the constraints formulated in Section 1.2, it is now possible to define spine-based deformation formally, in terms of a mathematical formulation of the mapping from the initial to the deformed position for each point of the input shape. This is one of the key contributions of this dissertation. Another key contribution reported in this thesis is a family of such mappings, while satisfying the requirements of preserving

cross-sections and normal segments, that preserve the local volume everywhere during the deformation. Details of the problem on local volume preservation is discussed in Section 1.5. Section 1.6 summarizes a full list of properties of our solutions presented in this dissertation.

1.4 Applications

When considering the applications related to spine-based deformation, we should first consider shapes or models that are suitable for deformation with a proxy curve or surface. In fact, in an interactive session, designer efforts such as shape modeling, spine specification and control point manipulation are critical aspects of the application's utility. In this section we mainly focus on the motivation and application scenarios for deformations driven by spine curves and surfaces. Sample models and shape suitable for this type of deformation are also briefly discussed.

1.4.1 Applications for deformation driven by a spine curve

Deformations driven by a spine curve are motivated by applications in modeling and animating tube-like structures, such as hoses, wires, ducts, and also the animations of a trunk, snake, or a tongue, as illustrated in Figure 3. In these models, it is intuitive to specify a curve, which may represent the central axis of an elongated part.

In one interactive shape-editing application developed by Llamas et. al. [35], the designer manipulates the spine using two frames, each controlled by a tracker in a different hand. The orientations of the trackers define the end-tangent directions to the spine. The total torsion along the spine is controlled by the rotations of the trackers around the corresponding tangents.

To help with simulation or animation applications, the designer should define the initial spine together with its evolution over time, as previously mentioned in Section 1.1. To do so, she would specify only a few control points of the spine and a few key positions for each control point that will be interpolated by the motion of

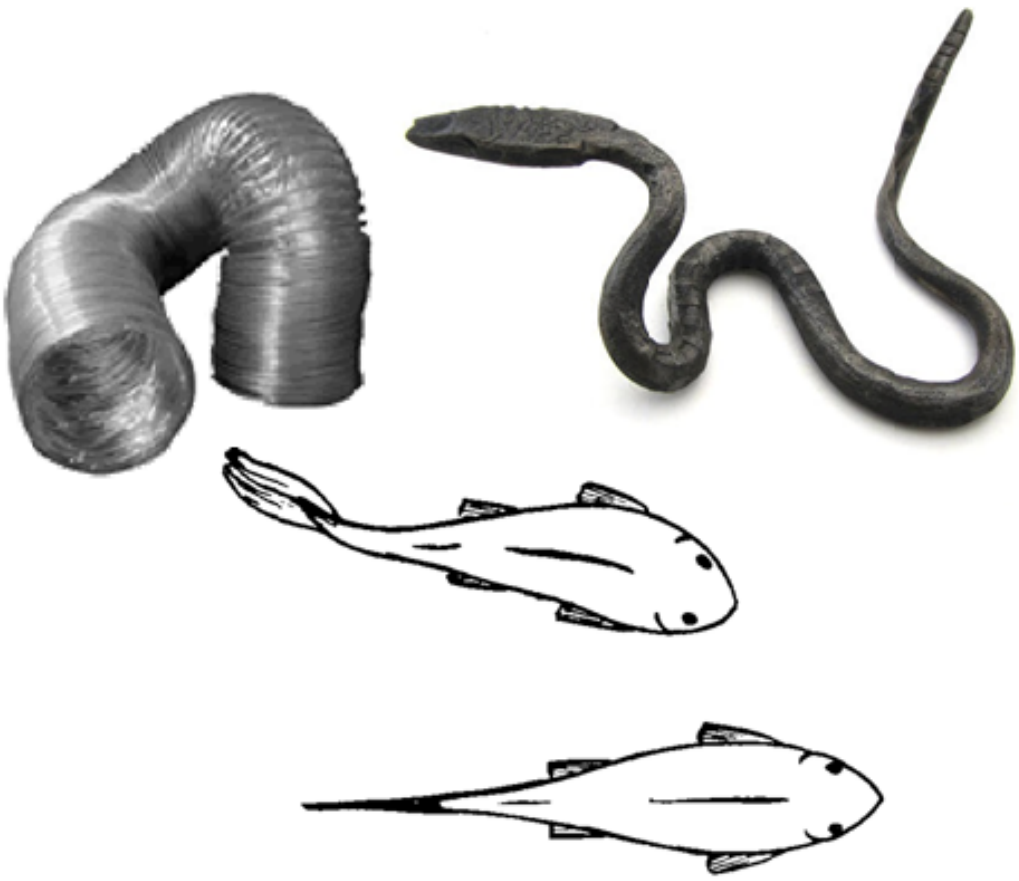


Figure 3: Models suitable for deformation driven by a spine curve.

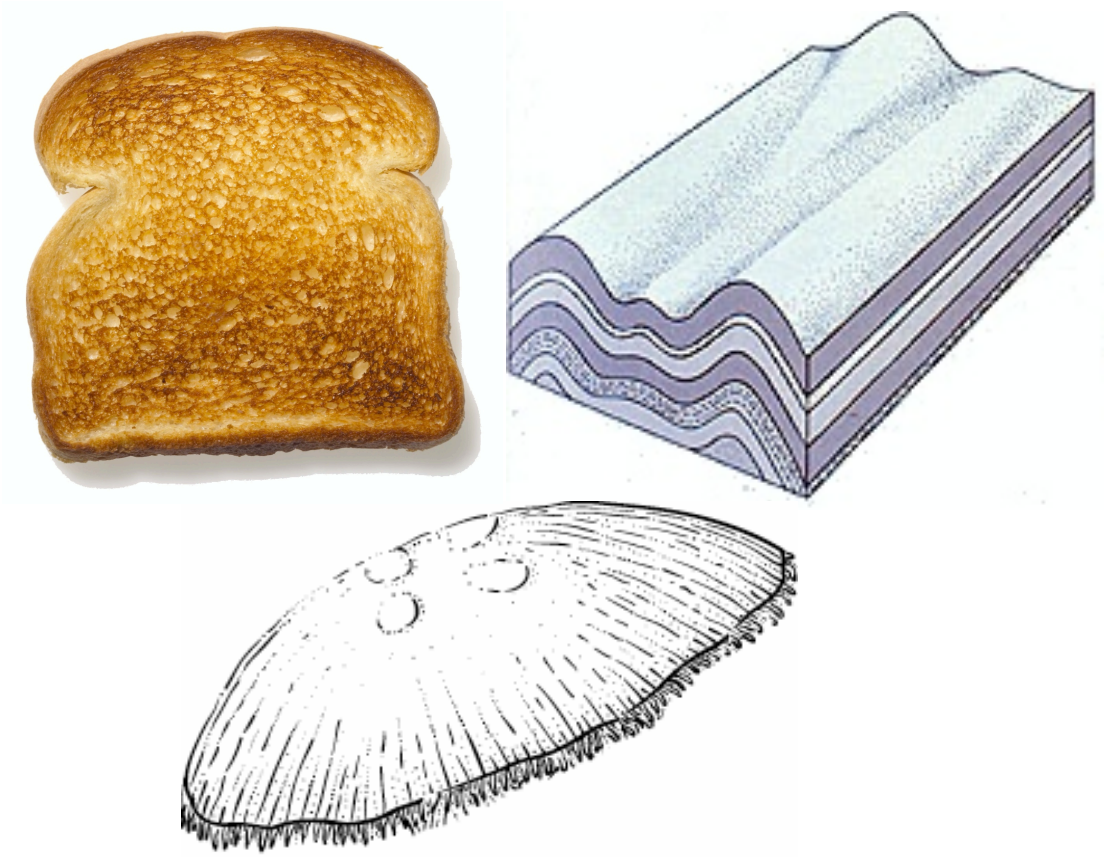


Figure 4: Models suitable for deformation driven by a spine surface.

that control point. Then for each time t, the deformation algorithm computes the current position of each vertex of the solid and displays the resulting deformed solid.

In those applications, the spine curve is the proxy which the designer uses to control the shape change. The deformation of the proxy curve itself is not the focus of this dissertation. Nevertheless, it is essential to compute changes of the thickness of the "meat" attached to the curve, due to the proxy curve's bending or stretching. For example, if the "meat" is a volume-preserving finite element mesh, how should the deformation algorithm compute the position of each cell vertex while the cell remains incompressible?

1.4.2 Applications for deformation driven by a spine surface

Deformations driven by a spine surface are motivated by applications in modeling plate-like structures, such as sheets, mattresses, smoothly bendable boards, and also the animations of creatures with soft shells or deformable membranes. We also anticipate potential applications in mechanics on thin plate bending, sheet-stamping and machining processes. Figure 4 shows a few examples suitable for deformation driven by a spine surface. In this models, it is intuitive to specify a surface, which may fit a part resembling to a thin plate.

Typically in machining or deposition process, the spine surface is static and is usually referred as the base or backbone surface. The material removal rate at a point on the base surface is characterized by the ratio of the removed volume to the local surface area at the point. For constant material removal rate, the milling depth is not a constant and should adapt to the local curvature of the base surface. Hence, if the base surface is developable, the milling depth can be computed by a locally volume-preserving bending of a plane into the base surface. Similarly, the deposition amount is characterized by the increased volume to the local surface area. We show that the thickness of the deposited layer can be computed by a special form of the

curvature-sensitive formula in our bending framework.

In modeling the deformation or animations of plate-like structures, the spine surface is the proxy which the designer uses to control the shape change. The deformation of the shape is completely determined by the change of the proxy surface as mentioned in Section 1.1. Instead of focusing on the deformation of the thin plate or the proxy surface itself, we are interested in the answer to this question: How does the thickness of the "meat" attached to the proxy surface change due to the bending and stretching of the proxy surface? For example, if the "meat" is a volume-preserving finite element mesh. Assume that the volume of each cell in the mesh is incompressible. Then our deformation algorithm should compute the exact shape of each cell while each cell volume remains a constant everywhere during the transformation.

1.5 Challenges with local volume preservation

Physically plausible simulations that involve biological creatures or deformable shapes made of incompressible materials require that the volume be preserved. Spine-based deformation has a wide range of applications in tissue modeling, surgical planning or even image editing. In these solutions, the area or the volume of the deformed region needs to be preserved for correctness or control quality. For example, during an animation where no external forces or torques are exerted on an incompressible body, the momentum and kinetic energy are preserved. Both depend on the mass and hence of the volume (if one assumes constant density). Hence, changes of volume during an animation will result in surprising changes of velocity.

It is much simpler to preserve global volume than local volume. For example, one may dilate the entire solid by a specific amount to compensate for the undesired volume gain or loss [17]. In fact, one of the contribution of our deformation algorithm is to provide a simple formula for computing that dilation amount for arbitrary (not necessarily convex) solids [63]. Unfortunately, preserving the global volume is not

sufficient for a physically plausible behavior.

1.5.1 Local volume preservation through thickness correction

Local volume preservation means to preserve the volume of "any chunk" in space, not just the summed volume of "all chunks" of a solid during transformation. Here we provide a 2D example to illustrate the effect of local volume preservation. Strictly speaking, the requirement becomes local area preservation in 2D as we want to preserve the area of any region in space during spine-based deformation.

Let us consider the two-dimensional version of the local volume preserving (or local area preserving) problem. As shown in Figure 5, the blue curve represents the spine that stabs a piece of incompressible material which has a layered structure along the spine curve. Assume that initially the layers are of the same thickness. Then the designer bends the spine curve downward and the layers along it are deformed accordingly. Here the question is how does each layer deform?

We show two versions of the deformed result in Figure 5. In the center of the figure, the thickness of each layer of the piece of material remains the same. At the bottom of the figure, the layers on the convex side of the spine becomes thinner, and the layers on the concave side of the spine becomes thicker. Which scenario is correct if we want the area of each layer to be preserved?

The answer is that the bottom of Figure 5 shows the correct deformed result if we want the area of each layer remain unchanged. Due to that a layer on the convex side of the spine is stretched, its thickness should decrease as the layer's length increases in order to compensate the area gain. On the contrary, a layer on the concave side of the spine is compressed, its thickness should increase as the layer's length decreases in order to compensate the area loss.

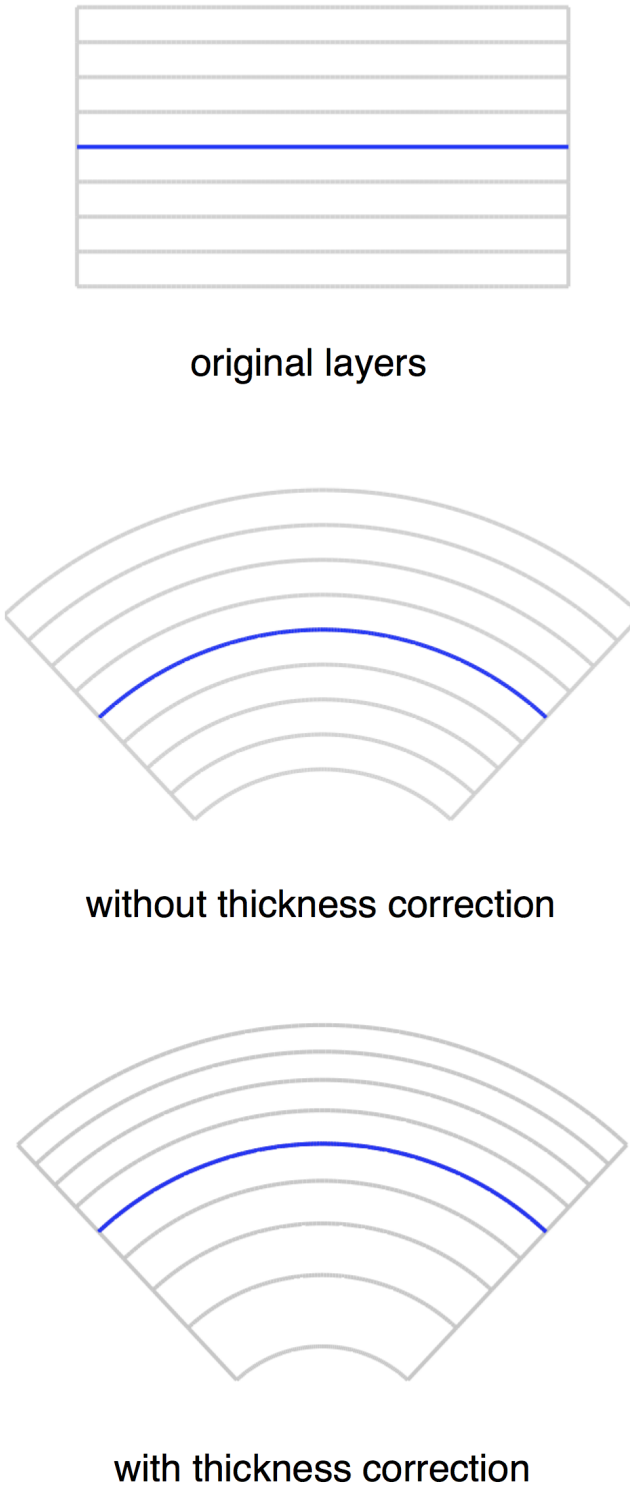


Figure 5: Local area preservation through thickness correction.

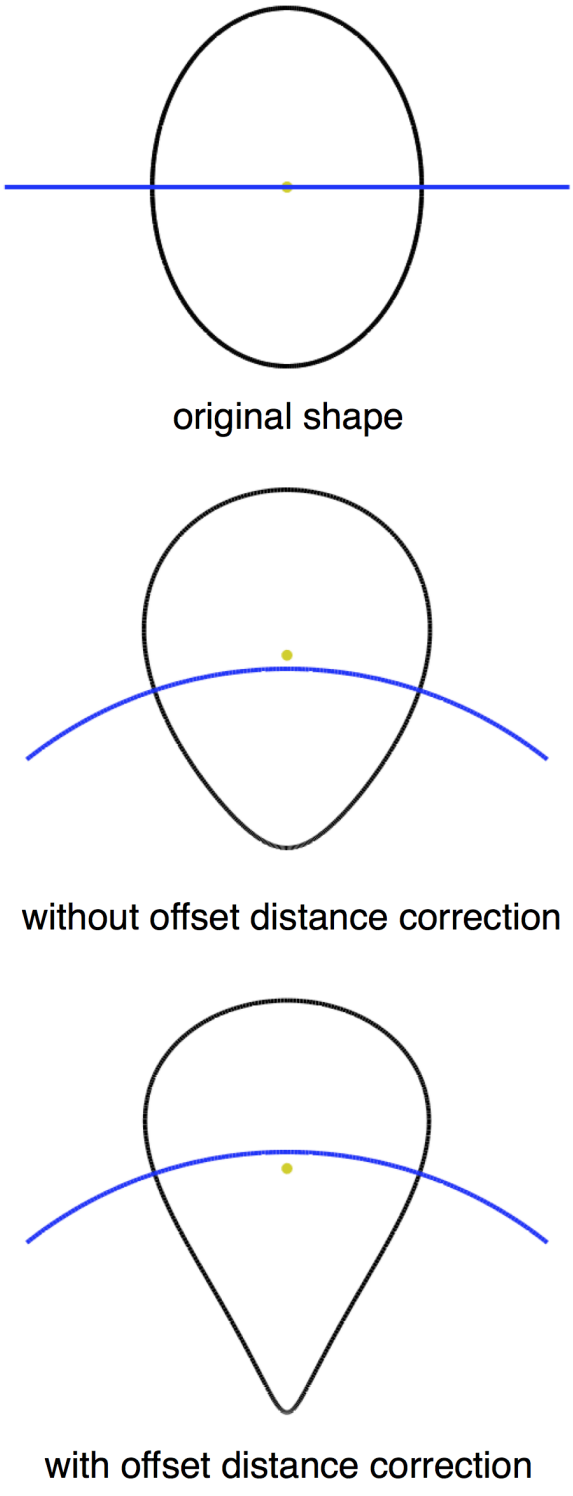


Figure 6: Local area preservation through offset distance correction. The yellow dot represents the centroid.

1.5.2 Local volume preservation through offset distance correction

Section 1.5.1 introduces an example of local volume preservation through thickness correction. To support more general planar shape deformation with local volume preservation, we extend the solution from correcting the thickness to correcting the offset distance of each point on the solid to the spine curve.

As shown in Figure 6, the blue curve represents the spine which stabs the center of a piece incompressible material which has the boundary of a ellipse. Note that the shape does not need to be an ellipse as shown later by an example in Figure 17. Assume that the designer bends the spine downward without stretching. This causes the shape to deform accordingly. The question is how does the shape look like after the deformation?

We show two versions of the deformed result underneath. The first version is shown in the center of Figure 6, where offset distance on each side of the spine remains the same after bending. The other version is shown at the bottom of the figure, where the offset distance increases on the concave side, and decreases on the convex side of the spine. Which scenario is the correct one if we want the area of the shape to be preserved locally?

The answer is not obvious but the bottom of Figure 6 shows the correct deformed result if we want local area preservation. Due to that the region on the convex side of the spine is stretched, the offset distance should decrease so as to compensate the area gain. On the contrary, the region on the concave side of the spine is compressed, and the offset distance should increase in order to compensate the area loss on the concave side. Naive deformation result without offset distance correction is shown in the center of Figure 6, where the area of the region above the spine is enlarged while the area of the region below is reduced. It appears that some area initially below the spine has magically transferred through and moved above the spine.

When the local area is preserved, the portion above the spine are stretched along

the spine and therefore become narrower, closer to the spine. Portions below are subject to the inverse effect: they are pushed away from the spine. This also affects the movement of the center of mass, which is visualized as a yellow dot in Figure 6. The center of mass is initially sits on the spine curve. After spine bending, the center of mass may change its relative position to the spine. In the correct deformation scenario where the "meat" are pushed away from the spine on the concave side, the center of mass tends to move below the spine.

Limiting or existence condition

When reconstruct the deformed point as the offset from the spine curve or surface, we need to prevent the condition that the offset distance exceeds the local radius of curvature. Otherwise, there is a local self intersection, hence no valid solution for the result to be locally volume preserving. To avoid this condition, the original point needs to be within the valid region defined by C_0 and C_1. For example in Figure 6 the updated offset distance $h_1 = f(h_0)$ should satisfy $h_1 < \frac{1}{\kappa_1}$, where κ_1 is the local curvature of C_1. Therefore the original offset distance should satisfy $h_0 < f^{-1}(\frac{1}{\kappa_1})$, where f denotes solver that updates the offset distance. In the following chapters, we will define and derive the limiting condition for the valid solution to exist for every type of spine-driven deformation.

Topological limitations on the spine

Kälberer and et. al. [29] studied projective field-based methods for parameterization of tube-like surfaces offset from spines with bifurcations. They have shown that a branch with Y-junction causes a singularity in the projective field and to the parameterization. (A branch with T-junction can cause an additional singularities.) Our framework relies on the assumption that the closest projection of a point on the spine should be unique. In the existence of a junction, the valid space for a point

to have a unique projection is reduced, degenerating to the lower-dimensional spine at the junction. Compared with work on skeleton-based deformation that allows junctions (e.g. [55, 34, 60]), our approach does not compute the deformed point as a distance weighted combination of points offset from multiple anchor points. Instead, we focus on computing the exact offset distance from one anchor point that leads to local area or volume preservation. Note that our approach may still be used to deformation driven by a spine with bifurcations. For instance, one may use our approach to efficiently compute the offset distances for points in the valid region to avoid expensive iterations for preserving the local volume while switch to distance-weighted interpolation for points outside the valid region.

1.5.3 Theoretical framework

Section 1.5.1 and 1.5.2 introduces planar deformation driven by a spine curve via thickness correction and offset distance correction in order to preserve the volume locally. Nevertheless, there are several assumptions need to be relaxed to support more general spine-based deformation with local volume preservation. First, the designer should be able to stretch the spine in addition to bending. In this case, the shape around the spine curve should be compressed or stretched axially as well as in directions normal to the spine, as discussed in Chapter 3. Second, the deformation should be in 3D, which has an additional degree of freedom compared to 2D. For the deformation driven by a 3D spine curve, there is an extra degree of freedom in the binormal direction, as discussed in Chapter 4. Finally, if the spine is a surface, an additional parameter is required for anchoring a point in space to the proxy surface, as discussed in Chapter 6.

To support spine-based deformation with local volume preservation as well as with various spine proxies, we introduce the mathematical framework. In order to formalize the notion of local volume preservation, one must introduce the measure of the

local volume change everywhere in the solid during its deformation. Let P_0 denote a point in space of the solid before deformation and P_1 the corresponding point after deformation. The transformation from P_0 to P_1 satisfies the requirements specified in Section 1.1. Let M represent the transformation, $P_1 = M(P_0)$. The Jacobian matrix of the transformation is $J_M = \frac{\partial P_1}{\partial P_0}$, where P_1 and P_0 are also regarded as mappings from local parameters to points in space. Given particular local parameters for P_0 and P_1, J_M is affine, meaning M is locally affine. However the overall spine-based deformation M is not an affine transformation, so J_M is spatially varying. Interestingly, the determinant of the Jacobian of the transformation, $det(J_M)$, is a good measure of the local volume change. There is a local expansion if the corresponding $det(J_M)$ is larger than 1. The reverse is local contraction if the corresponding $det(J_M)$ is smaller than 1. For local volume preservation, the determinant $det(J_M)$ should equal to 1 exactly everywhere.

1.6 Precise problem formulation

After having introduced the framework of spine-based deformation, we now turn to the problem formulation. Recall that the designer starts with a shape S_0 and specifies an initial spine C_0, which is a smooth curve that may pierce the solid S_0 or not. Then the designer deforms C_0 to C_1. The solution for any shape to maintain its original volume during deformation is to obtain a mapping $M : S_0 \rightarrow S_1$, such that M preserves volume locally (i.e., $vol(U) = vol(M(U))$, for any subset U of S_0). A list of requirements for M to be valid and producing plausible deformation results is the following.

1. *Topology-independent*: M should operate on any shape topology and independent of S_0. The input solid can be represented by either a triangle or a quad surface, or even a point cloud. M is a space transformation irrelevant to the representation of S. M is fully defined by C_0 and C_1. This allows the designer

to deform meshes without changing their connectivities.

2. *Homeomorphism*: M should be a homeomorphism between S_0 and S_1. This is important because we want the mapping to be invertible: $M^{-1}(P_1) = P_0$, where M^{-1} is defined by the initial spine as C_1 and the final spine as C_0. The homeomorphsim requirement is met as long as both S_0 and S_1 are valid without self-intersections caused by sharp bending.

3. *Crossection/Normal-preserving*: M should preserve the local parameter on the spine. In deformation driven by a spine curve, the local parameter s of the closest projection $C(s)$ of the point should remain the same. In deformation driven by a spine surface, the local parameters u, v of the closest projection $C(u, v)$ of the point should remain the same.

4. *Locally volume preserving*: Last but most important, $det(J_M) = 1$, meaning that M is local volume preserving everywhere. In another words, $vol(U) = vol(M(U)$ for any subset U of S_0. This is important for the physical plausibility of digital simulations, especially when they involve interactions between evolving, incompressible solids and surrounding fluids.

CHAPTER II

LITERATURE SURVEY

We present here an expository account of work related to spine-based deformation. Section 2.1 introduces models and tools for deformation driven by spine curve. Section 2.2 describes a intuitive problem and related solutions in global volume compensation. Section 2.3 discusses divergence-free deformation for preserving the volume locally in skeletal and surface-driven deformation. Finally, Section 2.4 presents variations of spine-based modeling in medical modeling and the key towards twist-minimization.

2.1 Deformation driven by spine curve

This section presents several planar spine-based deformation models and a tool towards 3D bending. We refer to the type of deformation driven by non-stretchable spine as *bending*. Section 2.1.1 discusses four planar bending models in a chronological order of their emergence, which also happens to reflect the degree of complexity of each model. Section 2.1.2 discusses Bender tool that support the deformation of 3D objects in detail.

2.1.1 Planar shape and image deformation

In 1984, Barr [7] presents a bending model in which the spine is the X-axis. The purpose is to simulate global linear bending where the length of the spine does not change, while the bending angle changes linearly in the bent region as shown in Figure 7. The spine is an arc with constant curvature k in the bent region. The offset distance from the spine does not change in this bending. This leads to the scenario shown in the center of Figure 5 or Figure 6 as discussed in Section 1.5.

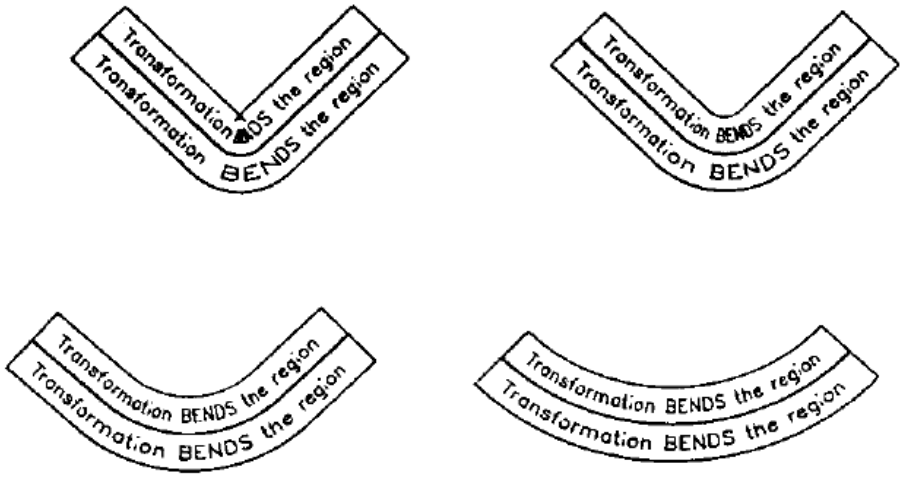


Figure 7: Barr 1984, Global and local deformation of solid primitives

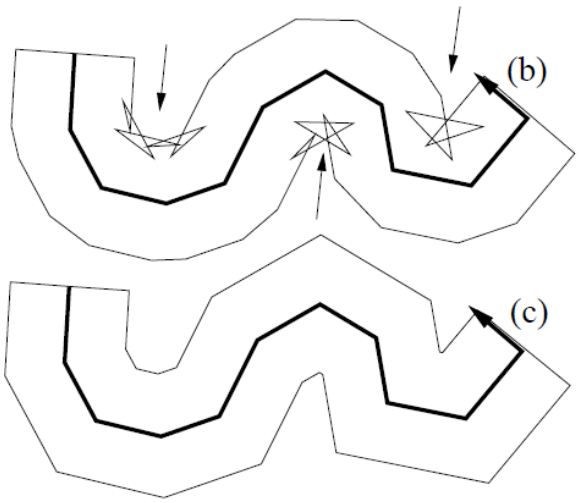
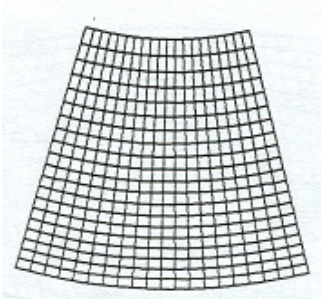
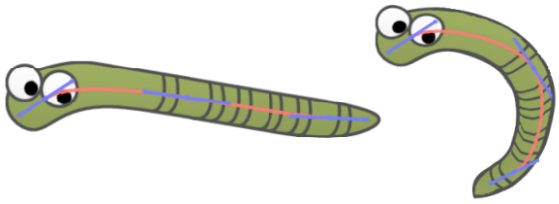


Figure 8: Hsu, Lee and Wiseman 1984, Skeletal strokes



Variable-offset bending by Chirikjian



Variable-offset bending applied to image deformation
by Zhuo and Rossignac

Figure 9: Variable offset bending

Later, Hsu, Lee and Wiseman apply a bending model similar to the one proposed by Barr to graphics design [27]. In this work, the spine is a user-specified planar curve, representing an artistic brush stroke, rendered with textures. They draw the textures using the normal to the spine curve as the local y-axis. The work supports more general bending in which the spine curve does not need to change linearly, or have constant curvature, in the bent region. They also deal with sharp bending, by trimming local self-intersections as shown in Figure 8.

The two approaches mentioned above do not preserve area of the shape locally. A drawback of the bending models described so far is that it can not produce the correct result for simulating the deformation of incompressible material. Usually on bending a physical object, the material on the concave side of the spine is compressed while stretched on the convex side. The way that the planar shapes deform, or the amount by which the material shrinks or expands should preserve the area of any subset in the material.

For the reason mentioned above, Chirikjian [16] proposed a mathematically precise approach for 2D bending with local area preservation. The solution is to update the offset distance based on the curvature of the spine as follows:

$$h_1 - \frac{kh_1^2}{2} = h_0,$$

where h_1 and h_0 are the updated and the original offset distances; k is the curvature after spine bending. Initially the spine is straight. The result applied onto an originally uniform grid is shown at the top of Figure 9, which is also the same as the scenario shown at the bottom of Figure 5.

In our work on curvature-based offset distance [63], we use this variable offset formula for bending an image with local area preservation. To alleviate the drawback of insufficient sampling, we use the spine-aligned grid. The deformed image is a texture mapping of the original image with the deformed grid, as shown at the bottom of Figure 9.

2.1.2 Bender tool

To support not just planar bending but the deformation of 3D objects and surfaces, Llamas, Powell, Rossignac and Shaw present a Bender tool [35], which allows the user manipulates the spine using two frames, each controlled by a tracker in a different hand. The orientations of the trackers define the end-tangent directions to the spine. The spine is modeled as a bi-arc curve [48]. The designer presses buttons that have been engineered on the trackers to indicate the moment where the current shape of the spine and of the torsion should be registered as the grab ribbon. Then, as the designer manipulates the two trackers, the current ribbon is computed at each frame.

In Bender, the mapping of the vertices of the solid is performed as follows. For each vertex P_0 of the solid, Bender computes the parameter s of the closest projection $C_0(s)$ of P_0 onto the spine of the grab ribbon. P_0 is expressed in the local frame at $C_0(s)$, $P_0 = C_0(s) + xT_0(s) + yN_0(s) + zB_0(s)$. Bender also computes the distance d between

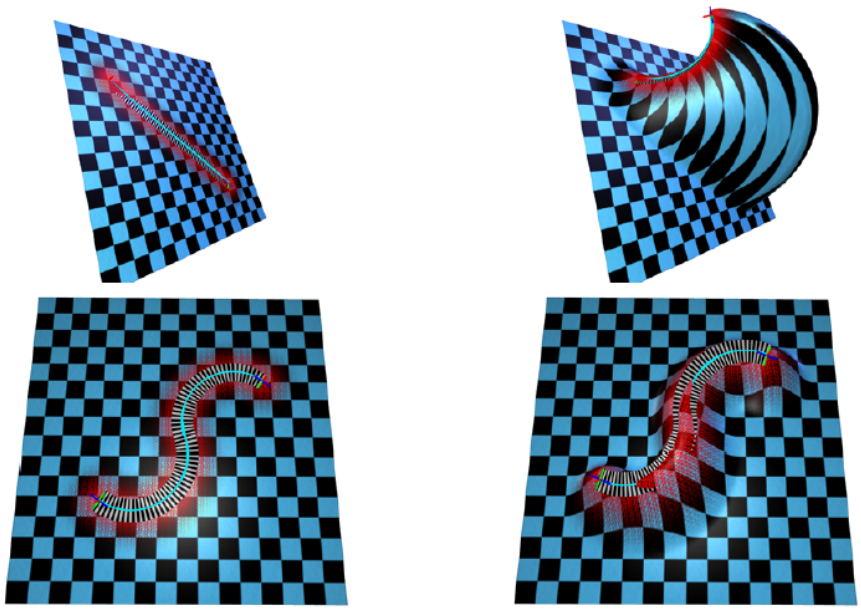


Figure 10: Llamas et al. 2005, Bender: A virtual ribbon for deforming 3D shapes

P_0 and $C_0(s)$. To compute the mapping P_1, Bender identifies the corresponding frame $T_1(s), N_1(s), B_1(s)$ on the current ribbon. However, instead of mapping P_0 to $\bar{P}_1 = C_1(s) + xT_1(s) + yN_1(s) + zB_1(s)$, they compute the screw motion M such that $M(0)$ is identity and $M(1)$ maps P_0 to $\bar{P}_1$. Then, it applies a fraction $M(f(d))$ of that screw motion to P_0 and obtain $P_1 = M(f(d))P_0$, where $f(d)$ is a decay function modeled using a cosine square expression of the distance from P_0 to $C_0(s)$.

The Bender approach is designed to support local tweaks, where the effect of the tweak blends smoothly with the unchanged surrounding, as shown in Figure 10. Specifically, to produce useful bending of tubular parts, the authors propose to change the f function to give it a plateau region. In this case, there is no attenuation and the effect of their mapping is similar to the one proposed here with two differences: (1) They can support an unnatural twist designed by the operator and distributed uniformly along the spine. (2) Even within the plateau region, their bending does not preserve the local volume.

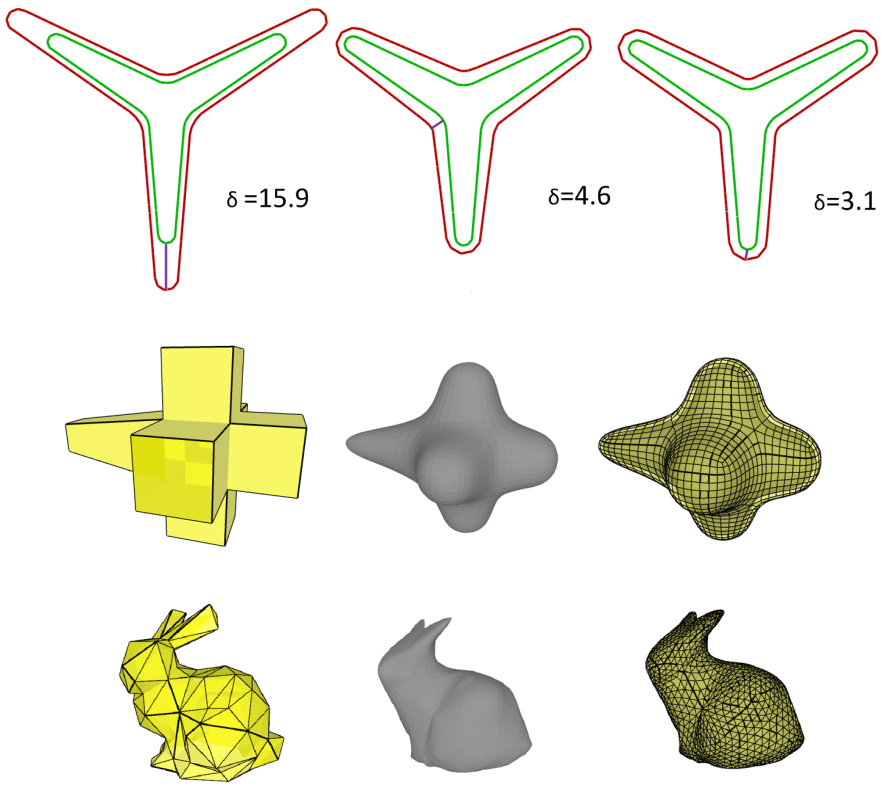


Figure 11: Zhuo and Rossignac 2012, global volume compensation with minimized Hausdorff error [63]

2.2 Existing techniques in global volume compensation

Maintaining the volume is important for modeling deformations where the volume occupied by the shape remains constant, and in physics-based simulations where material incompressibility matters [31]. In general, volume can be efficiently corrected by uniform scaling the shape by $s = \sqrt[3]{\frac{V_t}{V_0}}$, where V_0, V_t are current and target volumes [17]. However, uniform scaling may produce unbounded Hausdorff error between the original shape and the scaled shape, especially when the shape contains parts that are long and thin, as shown in Figure 11.

2.2.1 Volume compensation in freeform deformation

As a post-processing step, area or volume preservation has been studied for multi-level shape editing. Hahmann, Sauvage and Bonneau [23] present multiresolution

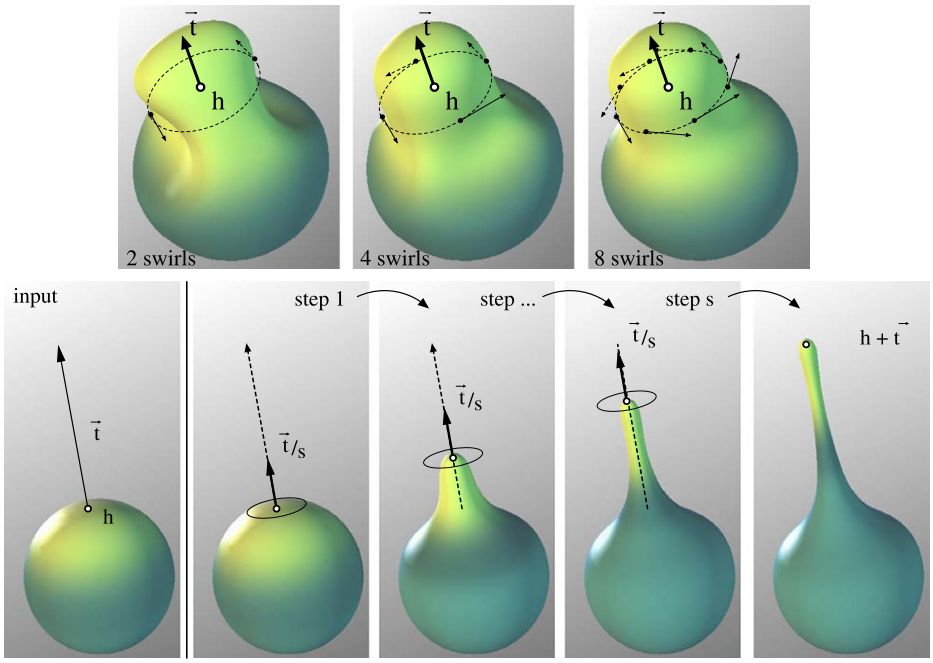


Figure 12: Angelidis, Cani, Wyvill and King 2006, Swirling sweepers.

deformation of curves which satisfy the bilinear constraint of constant enclosed area. In order for the volume to converge to the target value, they evaluate the current area at each iteration, and adjust the control vertices. The cost of volume evaluation is proportional to the number of vertices. The work of Hirota, Maheshwari and Lin [25] computes and corrects the volume of a shape at multiple subdivision levels so that the volume does not need to be evaluated at the highest subdivision level at each iteration.

Angelidis, Cani, Wyvill and King [2] define the basic operation, called a swirl, that locally twists the space around an axis. By arranging multiple swirls in a circle such that the twist axes of these swirls are coplanar and radially outward, they can achieve the effect of pulling along the direction normal to the twist axes. This simulates a stretching along the spine as shown in Figure 12. The overall transformation also preserves the volume locally.

2.2.2 Machining with equivolumetric offset

In machining with constant material removal rate, Moon [36] identifies a quadratic formula for offsetting backbone curves with uniform flux: The increased area is evenly distributed along the boundary. The formula is $h - \frac{kh^2}{2} = r$, where h is the milling depth and r is the material removal rate. In [63], we show how to generate a series of contours of this curvature-aware offsetting. Directly offsetting according to the formula exhibits an increasing amount of discontinuities where the curvature of the previous offset changes rapidly. We propose to use the combination of curvature-aware offsetting and selective smoothing to produce concentric offset contours that are smooth and approach a constant area-to-length ratio.

In differential geometry, a classical theorem due to Steiner [54] establishes the differential relationship between the surface properties and the volume enclosed: The amount of increased volume is a closed-form expression of the offset distance, surface area, Gaussian and mean curvatures. To preserve the total volume, one can grow or shrink the shape uniformly (via constant distance normal offsetting rather than global scaling) based on global curvatures in one step without iteration. For example, we compute the constant offset distance from the base surface that regain the target global volume: Instead of evaluating the curvatures everywhere on the surface, we define and evaluate the global curvatures to compute the constant offset distance[63]. We demonstrate the accuracy of our single step computation on triangle and quad meshes of various shapes.

Alternatively, Moon [37]'s solution is to compute the variable offset distance from the base surface that makes the deposition amount locally proportional to the surface area. Computational results were verified on cylindrical, ellipsoidal and catenoid surfaces.

2.3 Approaches to local volume preservation

The above subsection was focused on solutions that changed the boundary of a shape to adjust its global volume by a prescribed amount. Here we discuss prior art on the more delicate problem of ensuring that a continuous deformation process (such as one simulating the physical behavior of shapes, plastics, or fluids) remains divergence free at all points and at all times. This type of deformation process preserves the local volume, and is essential to a more natural, physically plausible behavior of the deformation.

2.3.1 Divergence-free displacement field

In deformation driven by a base surface, local volume preservation aims at preserving the local volume distribution between the base surface and the offset surface. Botsch and Kobbelt [10] explore the degrees of freedom in the position for a offset point to satisfy the local volume preserving constraint: They do not require the offset direction to be normal to the base surface. This is a departure from Reddy's classical model of bending thin plates [43], where the assumption is that lines remain normal to the base surface after bending. Moon's approach to equivolumetric offset [37] has the assumption of requiring the offset direction to be normal to the base surface.

However, it does not applicable to surface bending with local volume preservation as it assumes a static base surface. The formula may not be directly applicable to regular surface deformation as it does not take any local surface stretching or compression factor into account.

Local volume-preserving deformation of a object aims at obtaining a divergence-free displacement field for all points of the object: $\nabla \cdot V = 0$, where V is the vector-valued function denoting the displacement vector defined everywhere within the object. In finite element simulations [8], the displacement field is computed by time integration. At each time step, the processing consists of: (1) Evaluating the strain

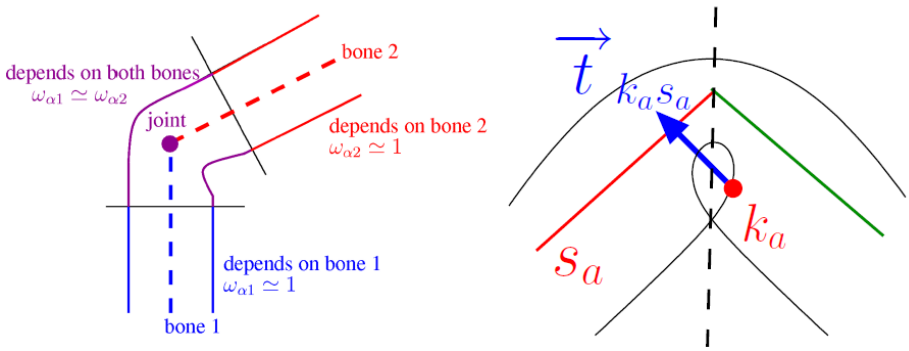


Figure 13: Rohmer, Hahmann and Cani 2008, Local volume preservation for skinned characters

and stiffness tensors from the object geometry and material property. (2) Computing the force field everywhere within and on the object from the evaluated strain and stiffness. (3) use the force field to update the velocity field. (4) correcting the velocity field to ensure that it has zero divergence.

2.3.2 Iterative normal displacement

In skeleton-driven deformation with local volume preservation, Rohmer, Hahmann and Cani [45] localize the volume correction on different regions of the shape's boundary. They use a constant, spatially-varying correction map specified according to the material property associated with each region. To correct the volume, they offset each point by a amount proportional to the correction map at each point. To avoid local self-intersections, they detect if an offset point is within its region determined by an automatic segmentation of the space around the skeleton. If a point is not within its region, they translate the point until it reaches the border of its associated region, as illustrated in Figure 13. Their subsequent work [46] further shows that a stylized deformation, such as isotropic inflation, bulging, or rippling effects, is possible by using 1D profile curves to control the correction map.

2.3.3 Local volume preservation in fluid simulations

Volume-preservation is essential in fluid simulations, solid-fluid coupling, and the interaction between fluid and bubbles or other media. The incompressibility is usually obtained by linearly solving for the implicit pressure values that make the velocity field divergence-free. This procedure is also called 'projection' [53], and follows the 'diffusion' and 'advection' processes.

Raveendran, Thuerey, Wojtan and Turk [41, 42] use volume-preserving morphing to interpolate liquids between control shapes. Their morphing algorithm produces a motion that is smooth while conserving the total volume. This effect is achieved by minimizing the objective function that is the sum of squares of local differences while subject to the total volume constraint requiring that the oriented interior volume equals a constant. Taking the derivative of this total volume constraint and applying Green's theorem result into a simplified form of the total volume constraint: The constraint that the sum of the volume associated with each chunk inside the boundary mesh should be a constant is equivalent to the constraint that the area weighted sum of the out-flux of the velocity field on the boundary mesh should be zero.

2.4 Variations of Spine-based modeling

After discussed prior work and techniques on spine-driven deformation with local volume preservation, we review here some of the variations of spine-based models in other contexts without volume preservation concerns. We have already mentioned that stroke design can be automated by curve bending [32]. In addition, Hsu and Lee extend the bending model to animating 2.5D cartoons [26]. They anchor different parts (which may have overlaps) of a image to a spine. The user can twist, bend or stretch the spine for deforming the parts and generate an animation.

Figure 14: Wang, Jüttler, Zheng and Liu 2008, Computation of Rotation Minimizing Frames

2.4.1 Medical modeling

Spine-based models in object recovery proves useful in vision research [39]. Various types of objects contain parts that may be formulated as generalized cylinders, which are the results of a possibly varying cross section along a path specified by a spine that may be an arbitrary space curve. The cross section needs not be connected so as to have bifurcations. In [4], Antiga, Ene-Iordache and Remuzzi extract such structures for blood vessels reconstruction and meshing from MR angiography. They also compute central paths and maximal inscribed balls in the vessel for blood vessel surface analysis, and use that paths as spine to generate deformed vessel surface. Variations of deformation driven by a spine surface are suitable for simulating or controlling a layered structure, such human tissue modeling [40].

2.4.2 Twist compensated frame

We first discuss spine curve registration that defines a point P with respect to the spine by the spine's parameter and also by the relative position of the point on the cross-section at s with regard to the spine. To register the point with respect to that cross-section, we need a frame consisting of two orthogonal vectors. That frame is defined at each point of the spine by a normal direction. A natural candidate is the Frenet normal which provides a convenient local frame along the curve. However, using the Frenet frame introduces unwanted twists as show in the top row of Figure 14.

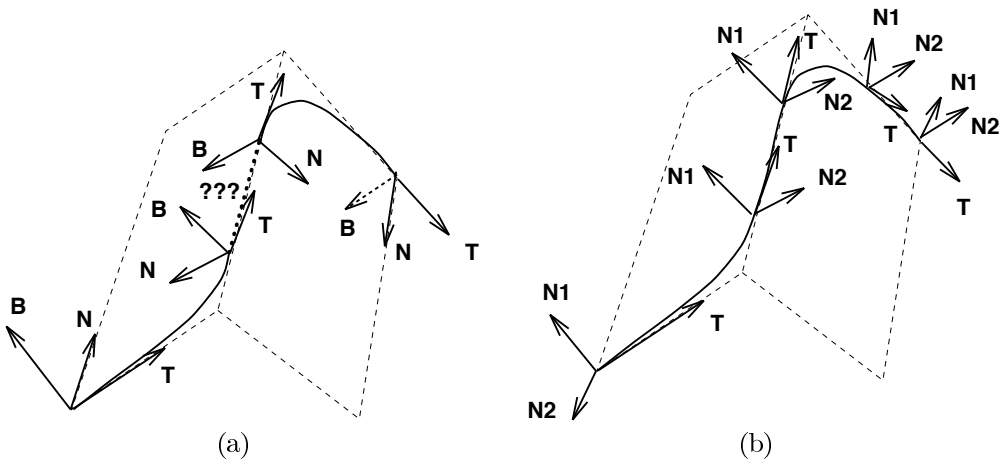


Figure 15: Hanson and Ma 1995, Roof-top analogy for approximating twist-minimized frame [24]

Wang, Jüttler, Zheng and Liu compute the rotation minimized frame as the better alternative to the Frenet frame. Hanson and Ma [24] define the solution in terms of parallel transport. They present the parallel transport algorithm that computes a smoothly varying frame consisting of a pair of parallel vectors orthogonal to each other. Their algorithm makes use of the rotation matrix [21] for generating the parallel transport frame along a piecewise linear approximation of a 3D curve.

CHAPTER III

DEFORMATION WITH 2D SPINE CURVE

Spine based deformation has a wide range of applications in tissue modeling, surgical planning or image design and editing. Often in these applications, the area or the volume of the deformed region needs to be preserved for correctness and control quality. This chapter formally introduce this problem in 2D. Section 3.1 gives the formulation and derivation for the deformation driven by a planar non-stretchable spine curve with local area preservation. Section 3.2 extends the solution to planar stretchable spine curve. Section 3.3 discusses the implementation and Section 3.5 presents experimental results and analysis.

3.1 Planar Non-stretchable Spine Curve

Assume that a planar curve stabs a piece of incompressible material. If the curve is bent downwards, how will the shape deform? We ask this question and provide two solutions in Section 1.5: In the first solution, the offset distance of any point on the planar shape remain the same. In the second solution, the offset distance decreases on the convex side of the curve and increases on the concave side. The following mathematical formulation explains the correct scenario in which the area of any subset of the planar shape is preserved locally during the deformation.

3.1.1 Formulation and derivation

Let $C(s)$ represent a planar curve in space. We assume that the spine curve is non-stretchable and s is the arc-length parameter. By non-stretchable we mean that, the length of the spine curve is preserved after bending. Also, let $T(s)$, $N(s)$ and k denote vectors representing the unit tangent, normal, and curvature at $C(s)$. Let P denote

a point near C, that is defined by the parameters s and h as follows,

$$P(s,h) = C(s) + hN(s).$$

Take the derivatives on both sides with respect to s and h,

$$\frac{dP}{ds} = C'(s) + h\frac{dN(s)}{ds} = T(s)(1+hk),$$

$$\frac{dP}{dh} = N(s).$$

In bending defined by a planar curve, the mapping $P_0 \to P_1$ is determined by $C_0 \to C_1$. In order to preserve the local area, we allow the offset distance h to change from h_0 to the computed value h_1. Following is the derivation of the closed-form solution of h_1 in term of k_1, k_0, h_0. The Jacobian determinant of the transformation is

$$det(\frac{\partial P_1}{\partial P_0}) = (1-k_1h_1)dh_1/(1-k_0h_0)dh_0.$$

In order to preserve the local area everywhere, we must set $det(\frac{\partial P_1}{\partial P_0}) = 1$. Therefore,

$$(1-k_1h_1)dh_1 = (1-k_0h_0)dh_0.$$

Integrating on both sides yields,

$$h_1 - \frac{1}{2}k_1h_1^2 = h_0 - \frac{1}{2}k_0h_0^2. \tag{1}$$

The value of h_1 is the root of the above quadratic equation,

$$h_1 = \frac{1-\sqrt{1-2k_1h_0+k_1k_0h_0^2}}{k_1}. \tag{2}$$

If the spine curve is initially a straight line, i.e. $k_0 = 0$, the solution of the offset distance can be simplified to

$$h_1 = \frac{1-\sqrt{1-2k_1h_0}}{k_1}. \tag{3}$$

3.1.2 Existence Condition

Note that h_1 is a variable offset distance depending on the local curvatures k_0, k_1 and the original offset distance h_0 according to Equation 1. In order to let h_1 have a valid solution, the right hand side of Equation 2 should be real. This means that k_0, k_1 and h_0 should satisfy the following inequality,

$$1 - 2k_1h_0 + k_0k_1h_0^2 > 0. \tag{4}$$

If the spine curve is initially straight, or $k_0 = 0$, then the constraint becomes

$$1 - 2k_1h_0 > 0. \tag{5}$$

Therefore the spine curve can not bend too much: the signed curvature k_1 of the spine curve after bending should satisfy $k_1 < \frac{1}{2h_0}$ everywhere on the curve.

3.2 Planar Stretchable Spine Curve

In some applications, the designer may want to not only bend the spine, but also compress or stretch the shape along the spine. In other words, the length of the spine curve is not preserved during deformation. Regions of the shape near portions of the spine that have been contracted become thicker so as to compensate for the area loss resulting from the spine contraction. Of course, by symmetry, regions where the spine is elongated become thinner. Local spine contraction or elongation may be modeled by changing the parameterization or, in simple situations, by moving the control points tangentially to the spine. The following mathematical formulation provides the solution of the variable offset distance for the planar shape's area to be preserved locally during a deformation defined by a contraction or elongation of the spine.

3.2.1 Formulation and derivation

Let $C(s)$ represent a planar curve in space, where s is not the arc-length parameter. The length of the curve may not be preserved during deformation. In this case we

say that $C(s)$ is a stretchable spine as the speed of the curve $|C'(s)|$ is not a unit everywhere. We can still use l to denote the arc-length parameter such that $l(s)$ is the arc-length of C from its origin to $C(s)$. The relationship between s and l is,

$$l(s) = \int_{s_0}^{s} |C'(s)| ds,$$

where $|C'(s)|$ is the magnitude of the speed of the curve. And the derivative relationship between them is

$$dl = |C'(s)| ds.$$

$T(s)$, $N(s)$ are vectors representing the unit tangent, normal at $C(s)$. Let P denote a offset point near C, we have

$$P(s,h) = C(s) + hN(s).$$

Therefore,

$$\frac{dP}{ds} = C'(s) + h\frac{dN(s)}{ds} = T(s)(1+hk)|C'(s)|,$$
$$\frac{dP}{dh} = N(s).$$

Similarly in the deformation $P_0 \to P_1$ driven by $C_0 \to C_1$ (where C is planar stretchable), we allow only one parameter, the offset distance h, to change from h_0 to h_1 in order to compensate the local area. Following is the derivation of the closed-form solution of h_1 in term of k_1, k_0, h_0, and the speed of curve $|C_0'(s)|$, $|C_1'(s)|$.

The Jacobian determinant of the transformation is

$$det(\frac{\partial P_1}{\partial P_0}) = (1 - k_1 x_1)|C_1'(s)|dh_1/(1 - k_0 x_0)|C_0'(s)|dh_0.$$

For locally volume-preserving transformation, $det(\frac{\partial P_1}{\partial P_0}) = 1$, therefore,

$$(1 - k_1 h_1)|C_1'(s)|dh_1 = (1 - k_0 h_0)|C_0'(s)|dh_0.$$

Integrating on both sides yields,

$$(h_1 - \frac{1}{2}k_1 h_1^2)|C_1'(s)| = (h_0 - \frac{1}{2}k_0 h_0^2)|C_0'(s)|$$

We define the local stretch of the curve as the ratio of the speed magnitudes:

$$\sigma = \frac{|C_0'(s)|}{|C_1'(s)|}$$

Then the closed-form solution to the offset distance h_1 is the root of the following quadratic equation,

$$h_1 - \frac{1}{2}k_1 h_1^2 = (h_0 - \frac{1}{2}k_0 h_0^2)\sigma \tag{6}$$

Note that the formula is similar to the non-stretchable spine (Equation 1) except that we need to scale the right hand side by the local stretch parameter, σ. The solution to the updated offset distance h_1 is,

$$h_1 = \frac{1 - \sqrt{1 - 2\sigma k_1 h_0 + \sigma k_0 k_1 h_0^2}}{k_1} \tag{7}$$

If the spine curve is initially a straight line, i.e. $k_0 = 0$, the solution to the offset distance can be simplified to

$$h_1 = \frac{1 - \sqrt{1 - 2\sigma k_1 h_0}}{k1}. \tag{8}$$

3.2.2 Existence Condition

h_1 is a variable offset distance that depends on the local curvatures k_0, k_1, the original offset distance h_0 and the local stretch parameter according to Equation 6. In order to let h_1 exists (as a real root), the right hand side of Equation 7 should be real. This means that k_0, k_1, h_0 and σ should satisfy the following inequality,

$$1 - 2\sigma k_1 h_0 + \sigma k_0 k_1 h_0^2 > 0. \tag{9}$$

If the spine curve is initially straight, or $k_0 = 0$, then the constraint becomes

$$1 - 2\sigma k_1 h_0 > 0. \tag{10}$$

And in that special case the spine curve can not bend or compress too much: the signed curvature k_1 of the spine curve after bending should satisfy $k_1 < \frac{1}{2\sigma h_0}$ everywhere on the curve. So in general h_0 should be less than $h_0 < \frac{1}{2k_1}$.

3.3 Discretization and Implementation

In this section, we report results of several approaches that we have explored for implementing 2D spine-based deformation with local area preservation. Section 3.3.1 explains how to implement planar spine-driven deformation by using spine-aligned grid with a family of offsets from one backbone curve. Section 3.3.2 introduces a variant where an offset curve is also the backbone curve for the following offset curve. This type of offsets can be applied to modeling machining tool path with constant material removal rate [63].

3.3.1 A family of curvature-based offsets

When the goal is to deform the portion of an image around the original spine C_0, such as a stylized stroke [26], in order to avoid the cost of registering the grid points to the original spine, we advocate using a spine-aligned grid, as shown in Figure 18. In this approach, the deformed image is texture mapped onto the deformed grid. We generate the initial grid by estimating the normal at each vertex P_i of the initial spine and by generating points offset in both directions by jr, with j being an integer in some valid range. At each such grid-point, we record its coordinates in the image as texture coordinates. To display the deformed image, we use the same process to establish the normal at each vertex of the bent spine, and generate the corresponding grid points, but instead of offsetting them by jr, we offset them by a curvature-based distance h_1 computed by Equation 2 with jr as h_0, where k_0, k_1 and σ are estimated from the discrete spine curve before and after bending or stretching.

3.3.2 A series of successive curvature-based offsets

We recall the quadratic formula proposed in the context of machining:

$$\frac{1}{2}kh^2 + h - r = 0$$

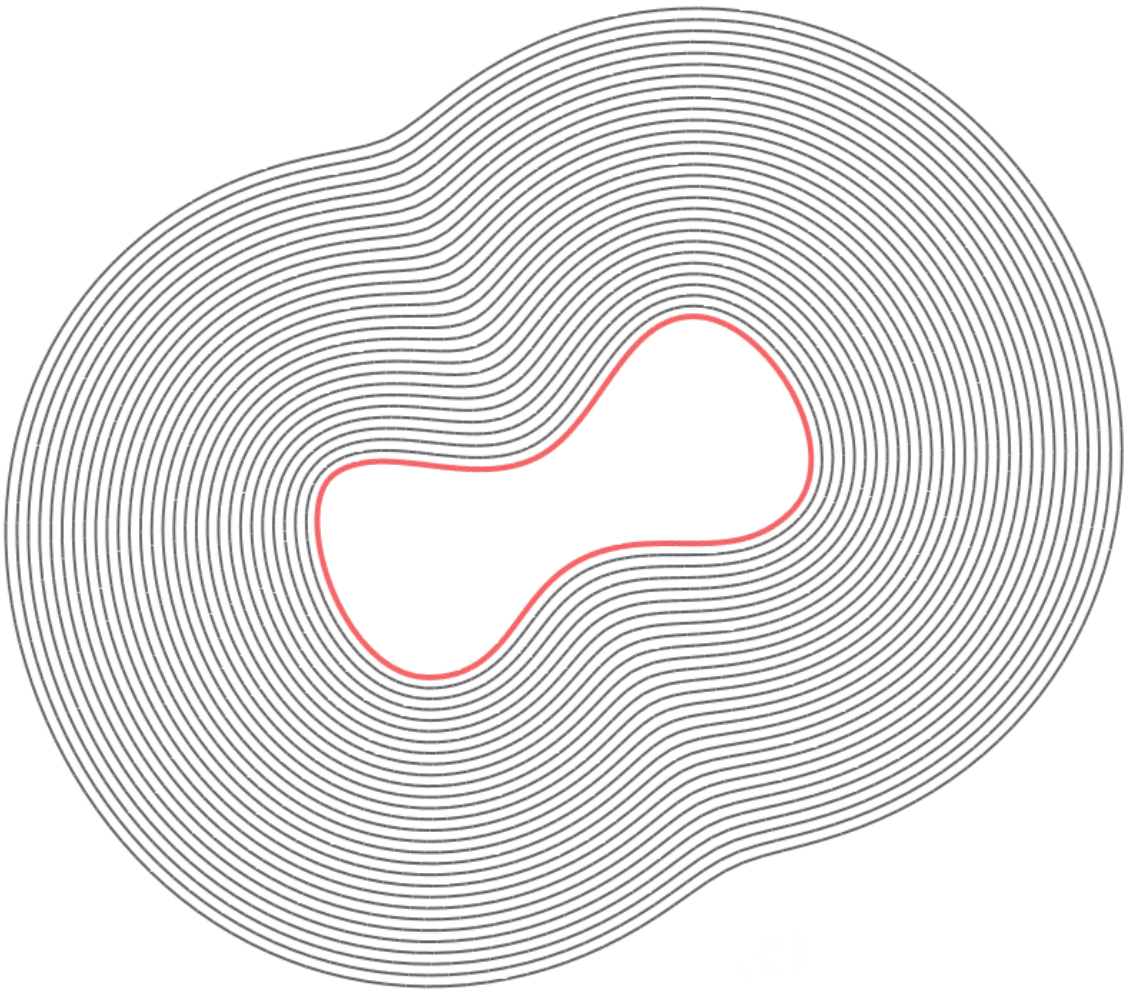


Figure 16: A series of successive curvature-based offsets.

where k is the local curvature of the progenitor curve P, h is the depth of cut, and r is the material removal rate to feed rate ratio. General milling tools have sufficient degrees of freedom which allow them to follow arbitrary planar paths. One of the challenges is to define a tool path that leads to constant material removal rate in milling for a target shape modeled by P. Since we want to keep the translational speed of the milling tool as constant as possible, the removed area per unit length should also be constant in order to achieve stable power consumption. Let this constant be r, solving the above equation gives the offset distance that defines the tool path with removed area per unit length equal to r.

In practice, the tool path consists of a set of concentric offsets from P. They form a set of *successive* offsets $\{O^j\}$, $j = 1, 2, \ldots$ from P:

$$\begin{aligned} O^1(s) &= P(s) + h_1(k_1 = k_P, h_0 = r)N_{P(s)} \\ O^{j+1}(s) &= O^j(s) + h_1(k_1 = k_{O^j}, h_0 = r)N_{O^j(s)} \end{aligned}$$

And $h_1(k_1, h_0)$ is computed with the corresponding parameters according to Equation 3.

3.3.3 Selective smoothing

However, it is known in differential geometry that the curvature transformation is a second-order operator on the base curve. Naturally, the curvature-based distance function $h_1(k_1, h_0)$ is second order as well. Hence only C^{d-2} continuity is observed in the offset when $P(s)$ is C^d continuous. This loss of smoothness is undesirable in generating a smooth offset curve. Hence, our smoothing strategy focuses on producing a curvature-compatible offset curve, where a point with non-negative curvature is mapped to a offset point with non-negative curvature, and the same for non-positive curvature. Selective Smoothing is similar to the Laplacian smoothing except that only points with non-compatible curvatures are subject to the operation. It consists of two steps in each iteration: *Select* and *Smoothen*.

Let k_i^o denote the discrete curvature at the i-th vertex on the offset curve O; k_i and N_i denote the signed curvature and the unit normal at P.

Select : Check each vertex O_i in O and put i into a smoothing list L if k_i and k_i^o are of different signs.

Smoothen : Compute a list of Laplacian vectors V_i at vertices of L; Move each vertex of L along the unit normal N_i by the dot product of V_i and N_i, and then empty L.

In Figure 16, we are able to generate a large series of consecutive curvature-based offsets using selective smoothing.

3.4 Projection, normal, curvature and stretch parameters for parametric and polygonal curve

Given $C(s)$ the parametric representation of a spine curve, the parameter s of the projection of a point P onto a spine curve $C(s)$ is formulated as $argmin_s dist(P, C(s))$, where $dist(P, C(s))$ is the distance from $C(s)$ to P. The normal $N(s)$ is defined by the normalized derivative of the unit tangent $norm(\frac{dT(s)}{ds})$, where $norm(V)$ is the normalized version of a vector V. The curvature is then the magnitude of the derivative of the unit tangent, $|\frac{dT(s)}{ds}|$. Also, as mentioned earlier the local stretch of the curve is defined as the ratio of the speed magnitudes, $\sigma = \frac{|C_0'(s)|}{|C_1'(s)|}$.

Next we discuss implementation problems when the spine C is a polygonal curve with a list of vertices C^j. The parameter s of the projection of a point P onto C is computed by $argmin_j dist(P, C^j)$. The normal N^j at vertex C^j is computed as the normalized vector $C^{j-1}C^{j+1}$ turned left. The curvature k^j at C^j is approximated by the inverse of the radius of the circle interpolating the three vertices, C^{j-1}, C^j and C^{j+1}. Note that we consider the resulting curvature is positive if the chain of the three vertices $C^{j-1}C^jC^{j+1}$ is a left-turn, and negative if it is a right-turn. Finally, to obtain the local stretch parameter, we compute the local speed at C^j as $|C^{j-1}C^j| + |C^jC^{j+1}|$

and take the ratio of the speed values obtained before and after a deformation of the spine.

For brevity, we assume that these parameters are only evaluated at vertices of the polygonal curve. While in the more accurate implementation (see Chapter 7), we need to evaluate these parameters on edges of the polygonal curve as interpolations of these parameters at nearby vertices.

3.5 Results and analysis

We start results with a trivial example, Figure 17, that shows bending an arbitrary planar shape with an arc. The rectangular region around the shape is striped to show how the normal offset distance changes in two different bending schemes, both of which are without stretching where the length of the arc remains as a constant. In the center, we offset each point from the spine without correction based on curvature: The offset distance remain the same, $h_1 = h_0$. Therefore, the thickness of each layer does not change. This leads to area loss for the layers below the spine, and area gain above the spine, despite that the total area of all shown layers remain unchanged. At the bottom, we offset each point by distance h_1, adjusted from h_0 based on curvature according to Equation 1. Therefore the thickness of each layer changes: it becomes thicker for the layers on the concave side of the spine, and thinner for the layers on the convex side. This leads to the effect that the area of each layer remains the same after bending.

Figure 18 shows an example of deformation of a spine-aligned grid driven by bending and stretching a spine curve. The spine is modeled by a polynomial curve interpolating four control points. We used Neville's algorithm to compute a series of points on the curve. The spine is initially straight. The center image shows bending the spine into an arc without stretching. The bottom image shows stretching and compressing the spine without bending by moving two center control points closer

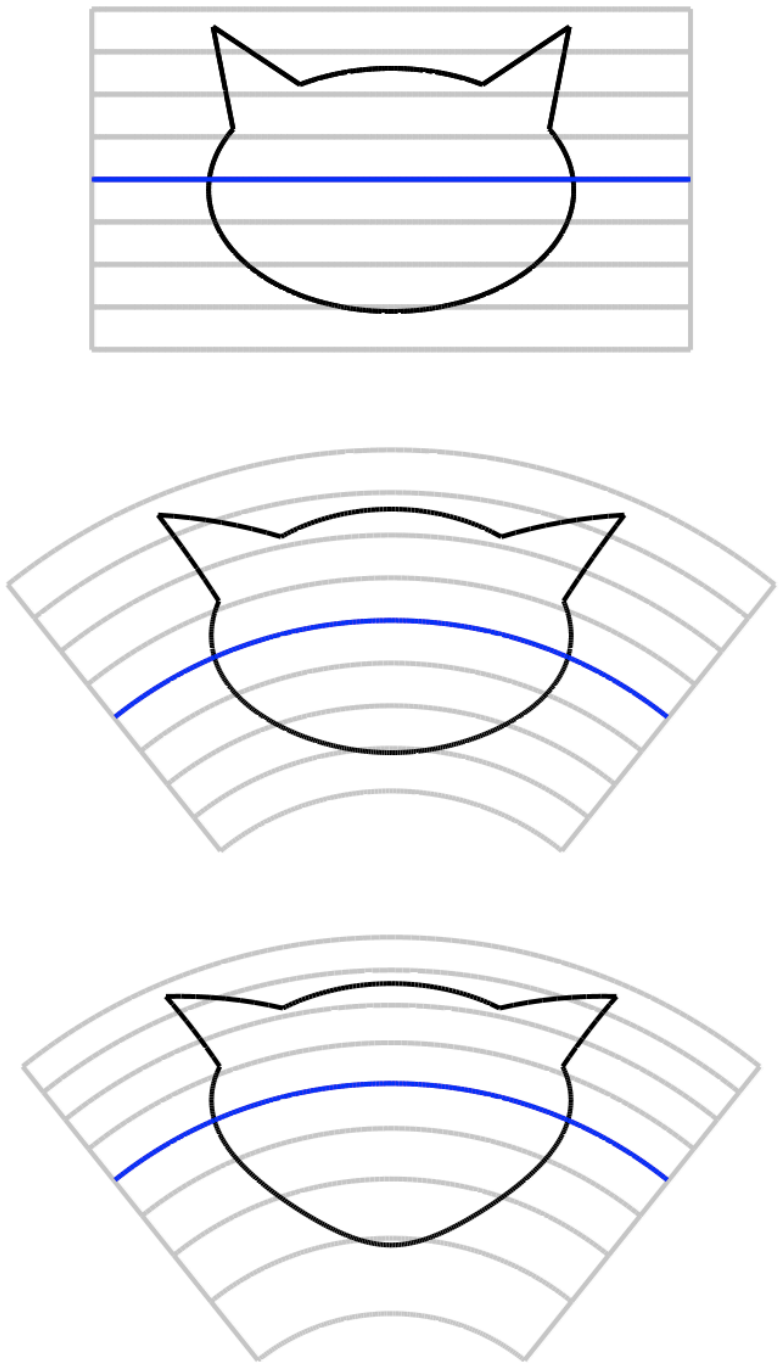


Figure 17: Bending incompressible shape and layers with an arc.

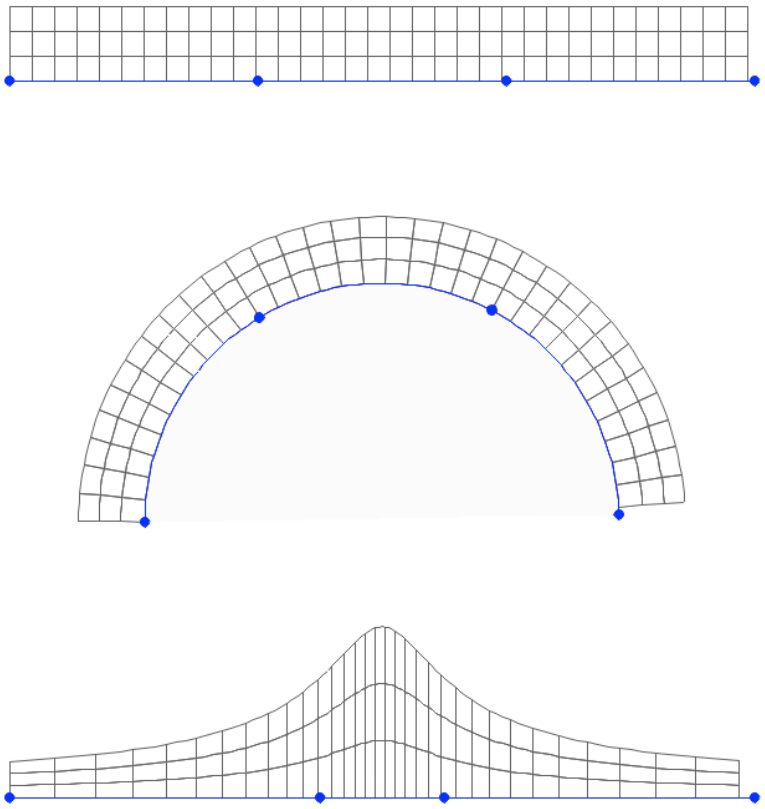


Figure 18: Deformation of a spine-aligned grid driven by bending and stretching a spine. Every rectangular cell preserves its area.

along the spine. After bending, stretching or compressing the spine curve, the offset distance for each grid point is updated according to Equation 8. The offset distance increases when the spine bends concavely or compresses axially, and decreases when it bends convexly or stretches axially. In this way, the area of each cell in the grid is preserved after deformation. Note that the cells are not exactly quads, but each have two straight sides.

To better reflect the area change in each quadratic cell in the spine-aligned grid, Figure 19 uses color mapping to map area loss to blue and area gain to red in a color ramp. The top image is the original spine with grid. The center of Figure 19 is without offset distance correction, and shows significant area change in most of the quad cells. At the bottom, the area of each quad cell is roughly preserved with the offset distance corrected according to the local curvature and stretch parameter.

The area is not perfectly preserved due to sampling and round-off errors. One way to reduce the error is to use more vertices to define each cell as we have discussed in one of our previous work [63]. Note that the boundary of each cell is represented as a polygon with 10 vertices and that the local area preserving deformation is applied to each one of them.

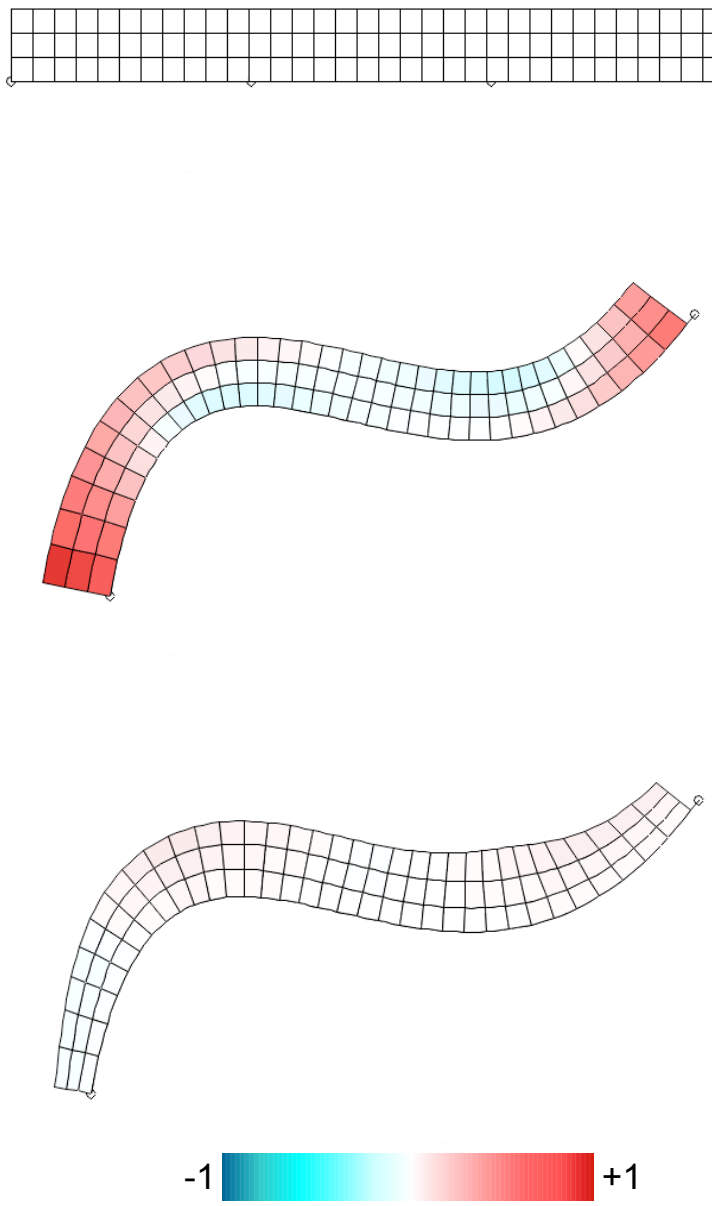


Figure 19: Using color mapping, we see how the area of each cell in the grid changes. The spine is the bottom curve.

CHAPTER IV

DEFORMATION WITH NON-STRETCHABLE 3D SPINE CURVE

In 2D, we compute the curve parameter s and distance h of a point to its closest projection $C_0(s)$ on the spine before deformation. We reconstruct the point after deformation by using the same s and update h according to the local curvature and stretch parameters. In 3D, we compute the curve parameter s and the projection vector $C_0(s)P$ for a point P in the 3D space. The projection vector can be updated by any displacement that remains orthogonal to the spine curve's tangent. Therefore, there is an additional degree of freedom to reconstruct the point, compared with the 2D case, as shown in Figure 20. In order to preserve the local volume of any chunk in space, we want to adjust the offset vector by adding a displacement. Even though we assume that this displacement must remain in the cross section, there are still two degrees of freedom to move the point. This chapter discusses the closed-form solutions for spine-driven deformation with local volume preservation in 3D.

Here we assume that the 3D spine curve is non-stretchable. The deformation of the curve is length-preserving. The curve parameter s itself is also the arc-length parameter. The length-preserving deformation of the spine curve can be implemented, for example, by uniformly sampling a polynomial curve interpolating a few control points. The polynomial curve is then adjusted by control point manipulation. The length of the spine curve is preserved by resampling the polynomial curve while keeping the number of samples and the distance between consecutive samples as constants. Other methods for length preservation, such as [50], can also be viable implementations as long as they provide the arc-length parameter for any point on the curve during the

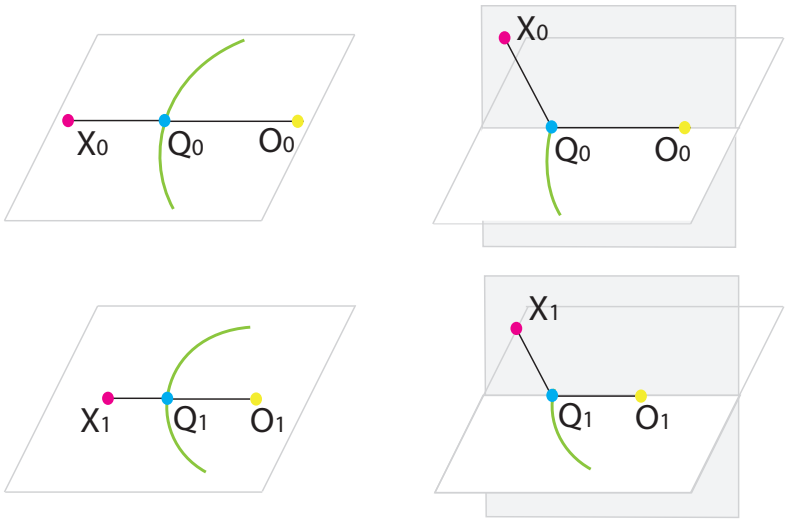


Figure 20: Depending on the direction to move the point in cross section, there is an additional degree of freedom for 3D spine curve (right) compared with the 2D spine curve (left).

deformation. Nevertheless, uniform resampling is a convenient way to achieve length preservation and to obtain the arc-length parameter without reparameterization.

4.1 Formulation and derivation

Let $C(s)$ represent a 3D curve in space. Let $T(s)$, $N(s)$ and $B(s)$ be unit vectors representing the Frenet tangent, normal and binormal at $C(s)$. Let P denote an offset point near C, such that,

$$P = C(s) + xN(s) + yB(s).$$

Therefore,

$$\frac{dP}{ds} = T(s) + x\frac{dN(s)}{ds} + y\frac{dB(s)}{ds}.$$

Using Frenet Serret formulae [18] yields,

$$\frac{dN}{ds} = -\kappa T + \tau B, \frac{dB}{ds} = -\tau N.$$

Therefore,

$$\frac{dP}{ds} = (1 - kx)T - \tau yN + \tau xB$$

$$\frac{\partial P}{\partial(s,x,y)} = \begin{bmatrix} (1-kx) & -\tau y & \tau x \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} T \\ N \\ B \end{bmatrix}$$

In the following solutions, we first assume that the bending occurs in the osculating plane spanned by T and B. We will relax this assumption in Section 4.2 that allows the solutions to extend to the general case. Specifically, Section 4.1.1 and Section 4.1.2 presents two intuitive solutions that explain how to compute values of x_1 and y_1 after bending. Section 4.1.3 proposes another solution, as a compromise between the previous two, to the family.

4.1.1 Normal solution

In the normal solution, we allow the parameter x to change from x_0 to x_1. Hence the Jacobian determinant of the transformation is

$$det(\frac{\partial P_1}{\partial P_0}) = (1-\kappa_1 x_1)dx_1/(1-\kappa_0 x_0)dx_0$$

Let $det(\frac{\partial P_1}{\partial P_0}) = 1$, we have,

$$(1-\kappa_1 x_1)dx_1 = (1-\kappa_0 x_0)dx_0$$

Integrating on both sides yields,

$$x_1 - \kappa_1 x_1^2/2 = x_0 - \kappa_0 x_0^2/2 \tag{11}$$

The solution x_1 is the root of this quadratic equation. The other coordinate remains unchanged ($y_1 = y_0$). Hence, the point P moves along N relative to the Frenet frame.

4.1.2 Binormal solution

In the binormal solution, we allow the parameter y to change from y_0 to y_1 but keep x constant. The Jacobian determinant of the transformation is

$$det(\frac{\partial P_1}{\partial P_0}) = (1-\kappa_1 x)dy_1/(1-\kappa_0 x)dy_0$$

$det(\frac{\partial P_1}{\partial P_0}) = 1$ gives,

$$(1 - \kappa_1 x_1)|C_1'(s)|dy_1 = (1 - \kappa_0 x_0)|C_0'(s)|dy_0$$

Integrating on both sides yield,

$$(1 - \kappa_1 x)y_1 = (1 - \kappa_0 x)y_0. \tag{12}$$

The solution y_1 is linearly related to y_0. The other coordinate remains unchanged ($x_1 = x_0$). The point P moves along B relative to the Frenet frame.

4.1.3 Radial solution

In addition to the normal and binormal direction, the point P can move in a direction radially outward within the cross section. In the radial solution, both x and y are updated, however their ratio $\tan\theta = \frac{y}{x}$, remains unchanged during deformation. So the radial solution can be seen as a compromise between the normal and binormal solutions. A point P near the spine is expressed as,

$$P = C(s) + r\cos\theta N(s) + r\sin\theta B(s)$$

In the radial solution, the offset distance r from the spine is adjusted from r_0 to r_1. The Jacobian determinant of the transformation is then expressed (in r_0 and r_1) as,

$$det(\frac{\partial P_1}{\partial P_0}) = \frac{r_1 dr_1(1 - \kappa_1 r_1\cos\theta)}{r_0 dr_0(1 - \kappa_0 r_0\cos\theta)}.$$

Let $det(\frac{\partial P_1}{\partial P_0}) = 1$. Then solve for r_1,

$$-\frac{2}{3}\kappa_1\cos\theta r_1^3 + r_1^2 = -\frac{2}{3}\kappa_0\cos\theta r_0^3 + r_0^2 \tag{13}$$

The solution r_1 is the root of the above cubic equation.

4.2 Implementation and Existence Condition

Before describing the implementation details, we first explain some of the experimental procedures. To demonstrate the correctness of the solutions, we initially use

relatively simple extrusion models, which are produced by sweeping a user specified planar cross-section along spine curves which are circular arcs. This allows us to better see the differences among the three solutions by showing the before and after images of the crossection of the extrusion model. After verifying the formulae, we implement them to general shapes and spine curves. However, the solutions provided by Equation 11, 12 and 13 assume that the bending (change of curvature) does not change the local Frenet frame. To support more general bending, Section 4.2.1 presents a technique that allows a change of basis in bending with general non-planar 3D curve.

4.2.1 Unbending-transfer-bending technique

To support more general bending, we split the computation of the offset distance into three general steps:

1. unbending: first solve for the updated value assuming that the initial spine $C_0(s)$ locally becomes a straight line. Specifically, for each point P_0, we first evaluate the local parameter (x_0, y_0 or r_0) of P_0 in the local Frenet frame. Then we compute the unbending image of the local parameter produced by assuming $\kappa_1 = 0$ in the formula.

2. transfer: We use the unbending image of the local parameter to produce a point, and compute its local coordinates in the original twist-compensated frame. Then we change the basis to the deformed twist-compensated frame, and evaluate the local parameter in the local Frenet frame of the deformed spine.

3. bending: After obtaining the local parameter in the new frame, we compute the bending image of the local parameter by assuming $\kappa_0 = 0$ in the formula. In this way, the spine is transformed from a straight line to its deformed version C_1.

We include in the following the formulae for steps 1 and 3. Step 2 will be motivated and discussed in Section 4.2.2. We provide three versions, that correspond to the three solutions, of unbending and of bending.

As for the Normal solution in Section 4.1.1, Equation 11 is limited to cases where the local curvature is changed, but the Frenet frame remains constant. To support non-planar spine curve bending, we provide below its decomposition into normal unbending and bending, which may be combined with the twist-compensation. To solve x_1, we break Equation 11 into two steps:

Normal Unbending: Assume that $C_0(s)$ is first straightened and we solve for the unbending image x_*,

$$x_* = x_0(1 - \frac{1}{2}\kappa_0 x_0). \tag{14}$$

As $\frac{x_*}{x_0} \geq 0$, the condition for a valid solution of r_* to exist is $|\kappa_0 x_0| \leq 2$.

Normal Bending: We then bend the straight spine into C_1 and solve for x_1 using x_*:

$$-\frac{1}{2}\kappa_1 x_1^2 + x_1 = x_*. \tag{15}$$

Hence, the closed-form solution for x_1 is

$$x_1 = \frac{1 - \sqrt{1 - 2\kappa_1 x_*}}{\kappa_1}.$$

In order for x_1 to be valid, the existence condition for κ_1:

$$\kappa_1 x_* \leq \frac{1}{2}, \tag{16}$$

and when κ_1 reaches this curvature limit, $x_1 = 2x_*$.

As for the Binormal solution in Section 4.1.2, Equation 12 is limited to cases where the local curvature is changed, but the Frenet frame remains constant. To support non-planar spine curve bending, we provide below its decomposition into normal unbending and bending. To solve y_1, we break Equation 12 into the following steps:

Binormal Unbending: (similar to normal unbending) In Equation 12, we set $\kappa_1 = 0$, $y_* = y_1$. This gives,

$$y_* = (1 - \kappa_0 x) y_0. \tag{17}$$

In order for y_* to be valid, we have $\kappa_0 x \leq 1$.

Binormal Bending: Let $\kappa_0 = 0$, $y_0 = y_*$ and we have,

$$y_1 = \frac{1}{1 - \kappa_1 x} y_*. \tag{18}$$

In order for y_1 to be valid, we have

$$\kappa_1 x < 1. \tag{19}$$

When κ_1 reaches this curvature limit, y_1 becomes unbounded.

As for the Radial Solution in Section 4.1.3, Equation 13 is limited to cases where the local curvature is changed, but the Frenet frame remains constant. To support non-planar spine curve bending, we provide below its decomposition into radial unbending and bending. To solve r_1, we break Equation 13 into two steps:

Radial Unbending: We first assume that $C_0(s)$ is straightened into a line (i.e. $\kappa_1 = 0$) and solve for the unbending image r_*:

$$r_* = r_0 \sqrt{1 - \frac{2}{3} \kappa_0 \cos\theta_0 r_0}. \tag{20}$$

In order for r_* to exist, $\frac{2}{3}\kappa_0 \cos\theta r_0 < 1$. As $\cos\theta$ varies in $[-1, 1]$, an sufficient condition for r_* to exist is $|\kappa_0 r_0| \leq \frac{3}{2}$.

Radial Bending: We then bend the straight spine into C_1 and solve for r_1 using r_*:

$$-\frac{2}{3} \kappa_1 \cos\theta_1 r_1^3 + r_1^2 = r_*^2. \tag{21}$$

In order to conclude the existence condition, we normalize the unknown and the coefficients in Eq. 36. Specifically, let $\lambda = \frac{r_1}{r_*}$ and $\alpha = -\frac{2}{3}\kappa_1 r_* \cos\theta$, then

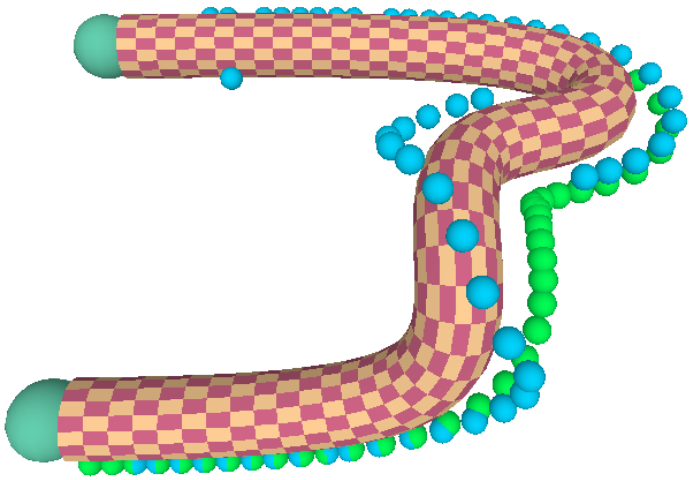


Figure 21: Showing the Frenet normal (blue) and the twist-compensated normal (green) along a twisted tube.

Eq. 36 becomes $\alpha\lambda^3 + \lambda^2 = 1$. Let $f(\lambda) = \alpha\lambda^3 + \lambda^2 - 1$, which has two local extrema (minimum at $\lambda_1 = 1$ and maximum at $\lambda_2 = -\frac{2}{3\alpha}$). If $\alpha > 0, \lambda_2 < 0$, then $f(0)f(1) < 0$ and $f' > 0 \in [0, 1]$, and hence there exists a valid solution in $[0, 1]$. If $\alpha > 0, \lambda_2 > 0$, then a valid solution exists only if $f(\lambda_2) > 0$, or equivalently $\alpha^2 < \frac{4}{27}$. Again since $\cos\theta$ varies in $[-1, 1]$, an sufficient condition for r_1 to exist is $|\alpha| < \frac{2}{3\sqrt{3}}$, or

$$|\kappa_1 r_*| \leq \frac{1}{\sqrt{3}}, \tag{22}$$

and when κ_1 reaches this curvature limit, $r_1 = \sqrt{3}r_*$.

4.2.2 Normal propagation

In order to perform the transfer step, one computes the local frame along the initial spine curve, registers the point to a frame at the closest projection on the spine. This amounts to computing the local coordinates. Then one computes the local frame along the deformed spine curve and constructs the deformed point using the local

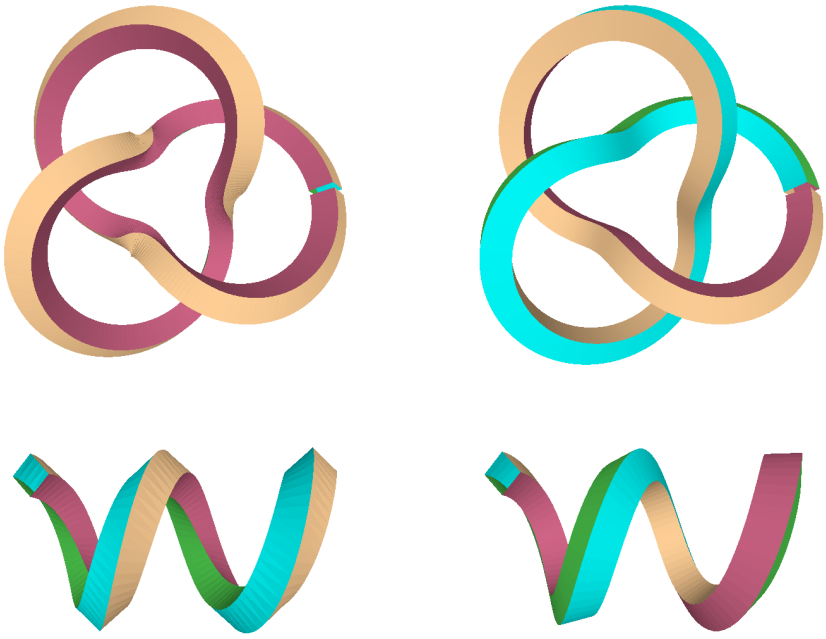


Figure 22: Reconstruction according to registrations with the Frenet frame (left) and the twist-compensated frame (right) on a trefoil knot and a helix.

coordinates. The local frame is usually aligned with the local tangent of the spine. The remaining issue is how to determine the other two basis vectors around the tangent. A natural candidate for the local frame is the Frenet frame $\{T(s), N(s), B(s)\}$ at $C(s)$ where $N(s)$ and $B(s)$ are the normal and binormal. We have used the Frenet frame to derive the three solutions in Section 4.1.

Although the Frenet frame provides a convenient local frame along the curve, it is not appropriate for registration and reconstruction, because it contains undesired twists and sudden changes of the normal direction, as shown in Figure 21 and Figure 22. To address this problem, we use a *twist-compensated* local frame. Its rotation with respect to the Frenet frame is defined by the integral of the torsion [49, 19]. We construct the twist-compensated normal using *parallel transport* by projecting the normal at the current vertex to the normal plane of the next vertex on the spine. Therefore, given an initial normal, the twist-compensated normal is obtained by propagation along the spine curve.

	Unbending	Bending	Bending with Rotation
Radial	1.27E-5	3.77E-5	3.97E-5
Normal	5.33E-5	9.93E-5	1.03E-4
Binormal	9.39E-5	1.85E-4	2.14E-4

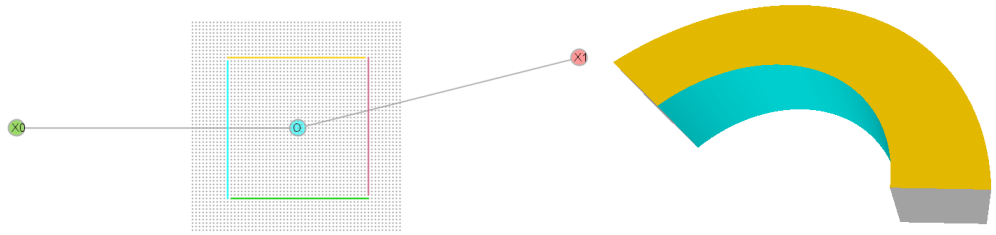
Table 1: The relative volumetric errors for mixed types of bend and unbend mappings.

4.2.3 Summary

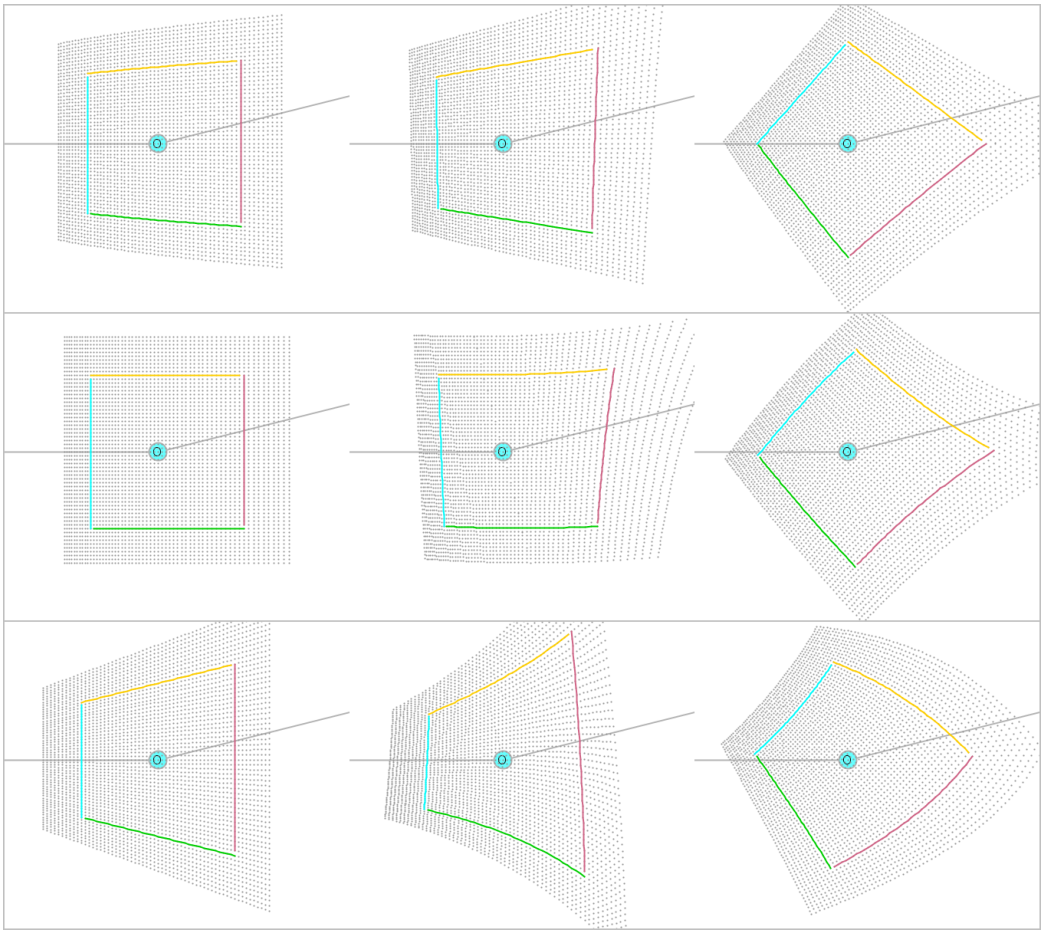
The implementation employs unbending-transfer-bending technique and normal propagation for registration and reconstruction with twist-compensated frames. While it is clear that the unbending and the bending step preserves the volume locally, the transfer step involves changing the basis vectors from the initial frame to the frame on the spine after deformation. This amounts to a rotation around the spine. Figure 23 shows the result of unbending, bending without and with a rotation on the extrusion model shown in Figure 23(a) with its cross-section embedded in a grid. The rotation step changes the deformation results as shown by the crossectional plots in Figure 23(b). However, the volume of the extrusion model is not altered by the rotation step, as shown in Table 1, which reports the volume of the extrusion model by multiplying the area of the cross-sectional shape with the distance travelled by its centroid [22]. Notice that the radial method nearly preserves straight lines even though it is not an affine transformation.

4.3 Results and Analysis

Figure 24 shows the deformation of two layers of tube surfaces driven by bending a straight spine into a circular arc. We show three types of correction to the original tube surfaces. Each shows the transverse and the frontal views of the bent tubes, and the cross section is dynamically plotted on the top-right. The red marks indicate how points move along the normal, binormal and radial direction. For example, the application of the normal solution to the cylindrical tube surfaces is shown by

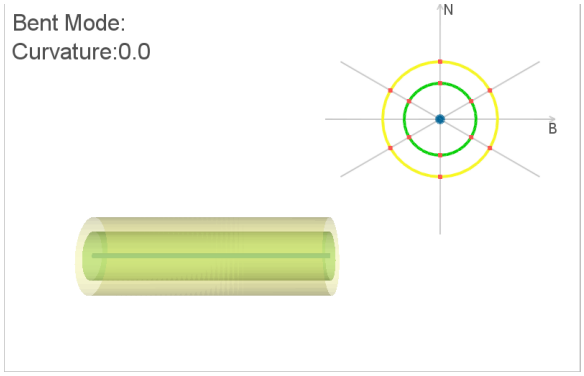


(a) Two specified axes of unbending and bending, and the rotation angle ($\pi/4$) for a initial extrusion model on the right

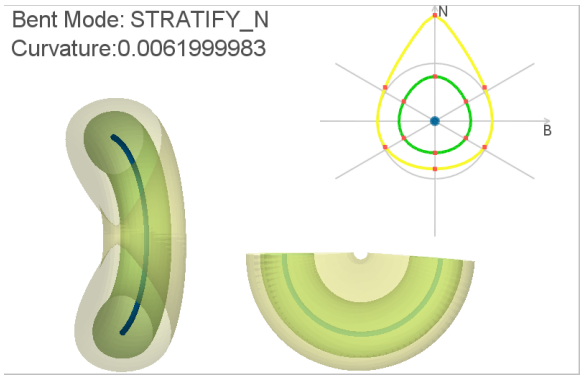


(b) From left to right we show the cross sections of the naive unbend and bend mappings and, in addition, the bend mappings with rotation. The top, center, and bottom rows are corresponding to the radial, normal and binormal solutions.

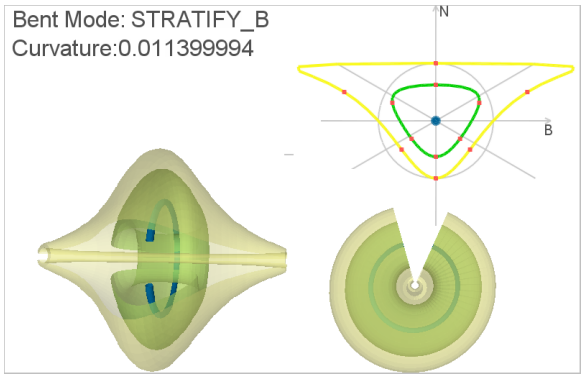
Figure 23: Showing the effects of rotation between the unbending and bending steps.



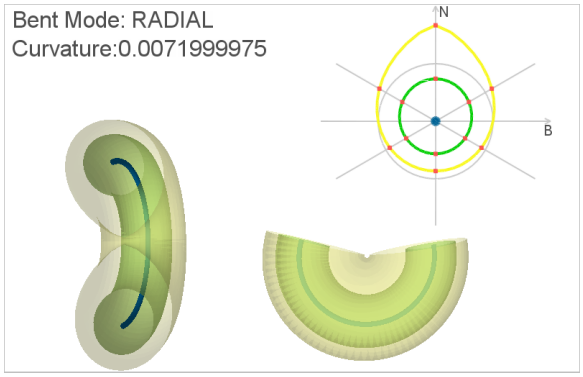
(a) original



(b) normal



(c) binormal



(d) radial

Figure 24: Frontal and crossectional views of a cylindrical model after bending.

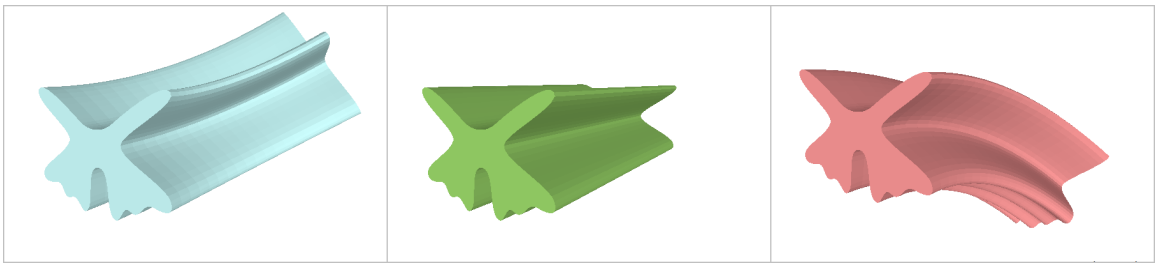


Figure 25: Unbending and bending a extrusion model.

Figure 24 (c). As shown in the cross-sectional plot, points are constrained to slide in the normal direction. Similar to the 2D solution, the tube surface stretches on the concave side and shrinks on the convex side of the circular spine in order to compensate for local compression and expansion. When reaching the curvature limit in Inequality 29, the tube surface starts to intersect itself. Note that the normal solution has a more stringent curvature limit than the radial one (Inequality 37) for the same initial tube surface. Figure 24 (d) shows the application of the binormal solution to the cylindrical tube surfaces. As shown in the cross-sectional plot, points can only move in the binormal direction: they expand or shrink bilaterally on the concave or the convex side of the circular spine. When reaching the curvature limit in Inequality 33, the tube surface becomes flat on the concave side. Note that the binormal solution has the least stringent curvature limit among the three solutions.

Figure 25 presents a example with unbending and bending a extrusion model. The extrusion model is produced by sweeping an arbitrary planar crossection along a smooth 3D spine curve. For example, the user draws a contour and indicates the point stabbed by the spine. The initial solid is in blue. The solid after unbending and bending are in green and red respectively. The crossectional views of radial, normal, binormal unbending and bending are shown in Figure 26. Notice that the radial fleshing nearly preserves straight lines even though it is not an affine map.

Figure 27 shows bending a triangle mesh representing a bunny, with the initial spine shown in green. The deformed the spine is shown in red. On the right, we shown

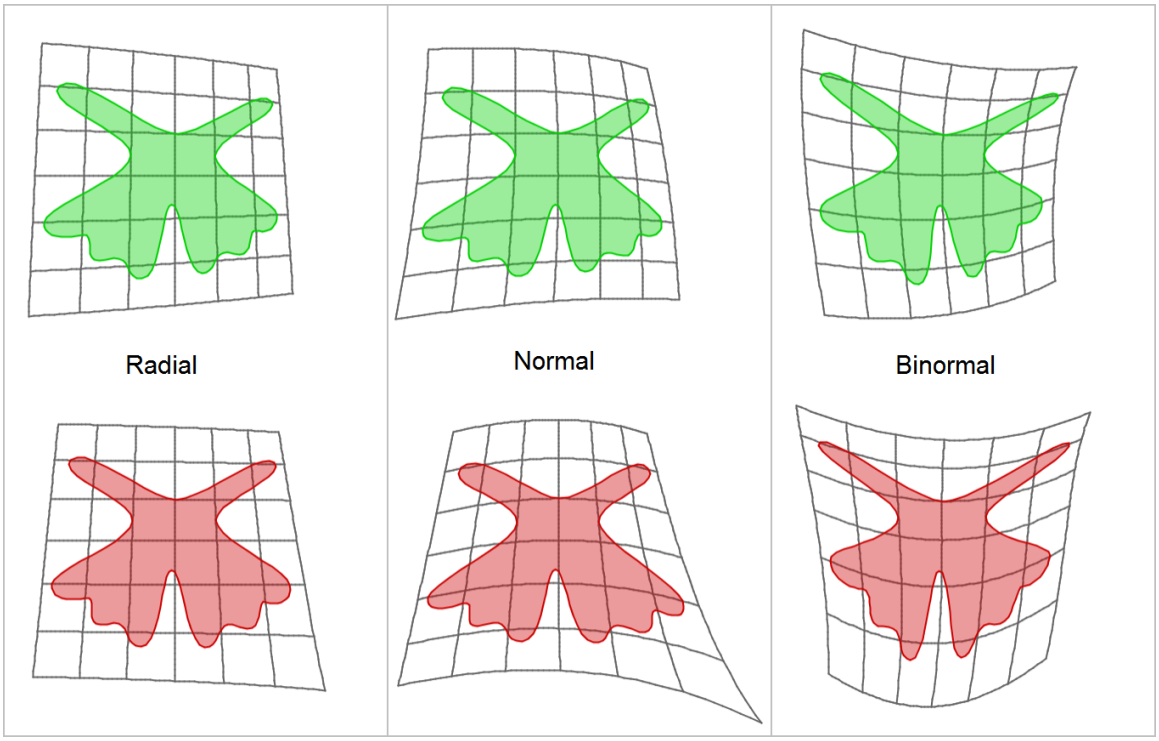
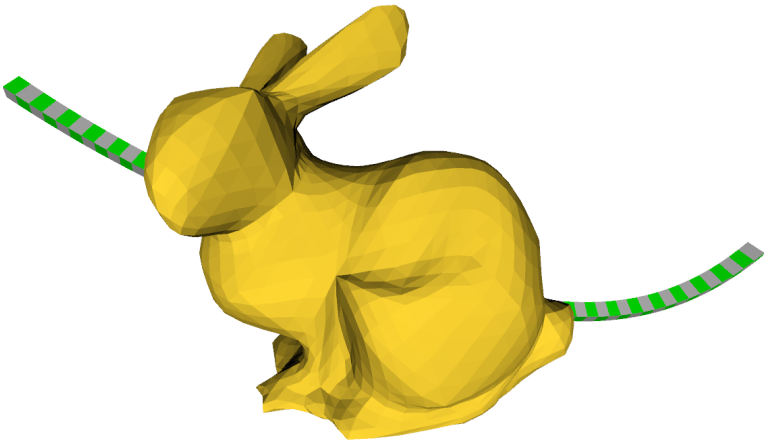
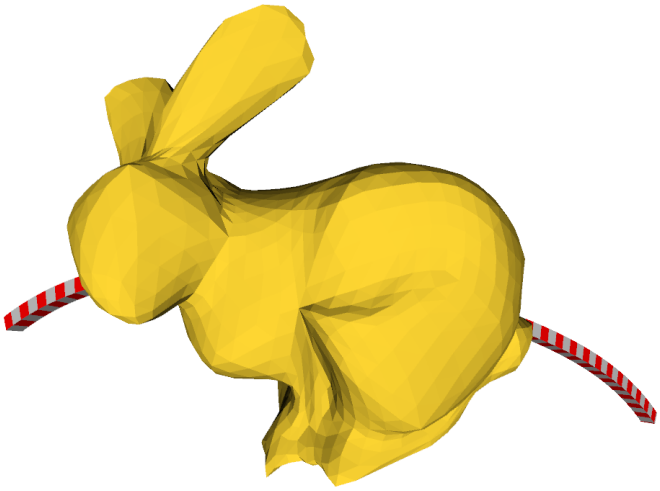


Figure 26: Crossectional views of the extrusion model after unbending and bending.

the result obtained using the standard reconstruction without volume-preserving correction, for which the total volume change is 9%. Figure 28 show the results produced by the three correction schemes: radial, normal, and binormal, for which the total volume change (due to sampling and round off errors) is less than 0.3%.



original bunny and initial spine



standard deformation without correction

Figure 27: The original bunny and the deformed bunny without correction.

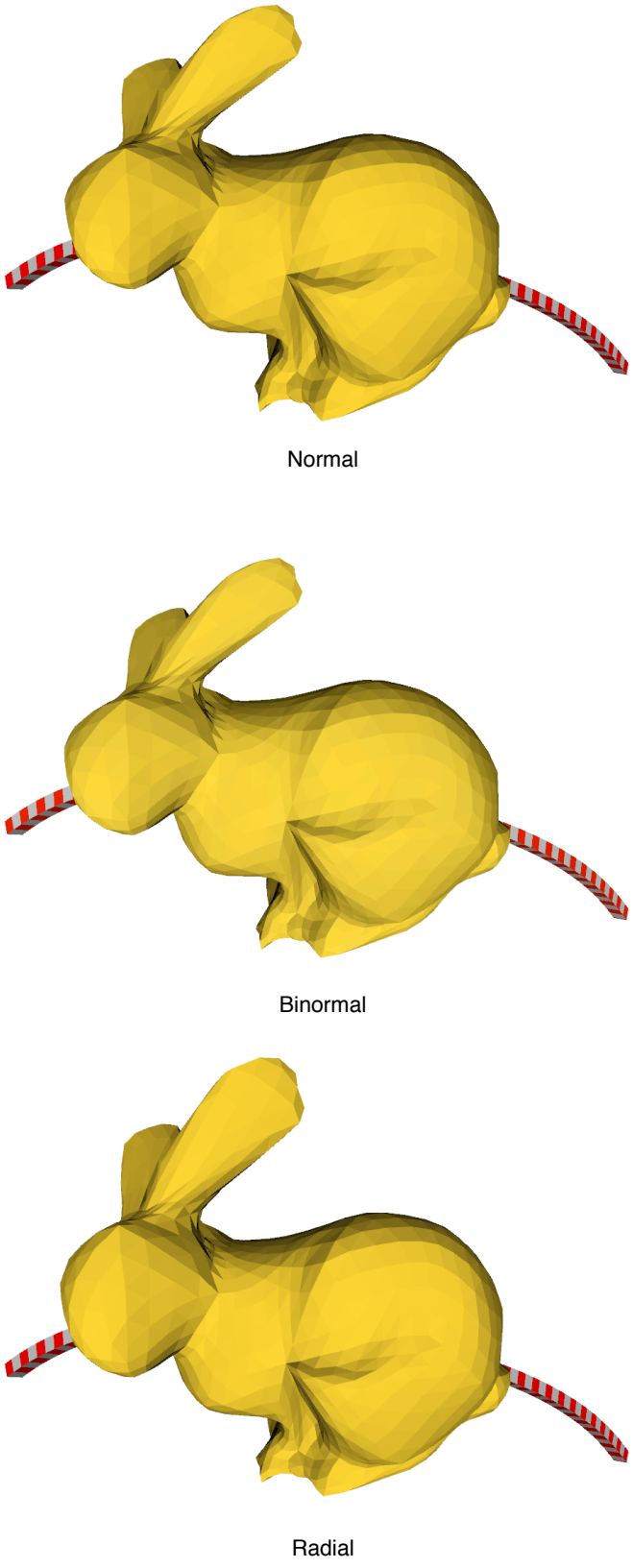


Figure 28: The deformed bunny with normal, binormal and radial correction.

CHAPTER V

DEFORMATION WITH STRETCHABLE 3D SPINE CURVE

In this chapter, we relax the assumption of non-stretchable 3D spine curve. The deformation of the spine curve does not need to be length-preserving. This allows the designer to stretch the spine curve and the model registered on it. We can use the natural parameter directly instead of the arc-length parameter. We include the derivation for local volume preserving deformation with stretchable 3D spine in Section **??** for completeness.

5.1 Formulation and derivation

Let $C(s)$ represent a 3D curve in space. For 3D stretchable spine, s is the natural parameter instead of the arc-length parameter. Same as the 2D case, the derivative relationship between s and the arclength l is

$$dl = |C'(s)|ds.$$

Let $T(s)$, $N(s)$ and $B(s)$ be unit vectors representing the Frenet tangent, normal and binormal at $C(s)$. Let P denote an offset point near C, such that,

$$P = C(s) + xN(s) + yB(s).$$

Therefore,

$$\frac{dP}{ds} = C'(s) + x\frac{dN(s)}{ds} + y\frac{dB(s)}{ds},$$

or,

$$\frac{dP}{dl} = T(s)|C'(s)| + x\frac{dN(s)}{dl}|C'(s)| + y\frac{dB(s)}{dl}|C'(s)|$$

according to the derivative relationship.

Using the Frenet Serret formulae [18] yields,

$$\frac{dN}{dl} = -\kappa T + \tau B, \frac{dB}{dl} = -\tau N.$$

Therefore,

$$\frac{dP}{ds} = (1 - kx)|C'(s)|T - \tau y|C'(s)|N + \tau x|C'(s)|B$$

$$\frac{\partial P}{\partial (s, x, y)} = \begin{bmatrix} (1 - kx)|C'(s)| & -\tau y|C'(s)| & \tau x|C'(s)| \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} T \\ N \\ B \end{bmatrix}$$

Notice how the derivation takes the magnitude of the speed of the curve $|C'(s)|$ into account: A variable parameterized by s is multiplied by $|C'(s)|$ when taking the derivative with respect to the arclength. This differentiates the derivation of stretchable formulae from non-stretchable ones.

5.1.1 Normal solution

In the normal solution, we allow the parameter x to change from x_0 to x_1. Hence the Jacobian determinant of the transformation is

$$det(\frac{\partial P_1}{\partial P_0}) = (1 - \kappa_1 x_1)|C_1'(s)|dx_1/(1 - \kappa_0 x_0)|C_0'(s)|dx_0$$

For locally volume-preserving transformation, $det(\frac{\partial P_1}{\partial P_0}) = 1$, therefore,

$$(1 - \kappa_1 x_1)|C_1'(s)|dx_1 = (1 - \kappa_0 x_0)|C_0'(s)|dx_0$$

Integrating on both sides yields

$$(x_1 - \kappa_1 x_1^2/2)|C_1'(s)| = (x_0 - \kappa_0 x_0^2/2)|C_0'(s)|.$$

Let σ represent the local stretch, or the ratio of the curve's speed before and after the deformation:

$$\sigma = \frac{|C_1'(s)|}{|C_0'(s)|}.$$

Then the solution x_1 is the quadratic root of the following equation,

$$(x_1 - \kappa_1 x_1^2/2)\sigma = x_0 - \kappa_0 x_0^2/2 \tag{23}$$

5.1.2 Binormal solution

In the binormal solution, we allow the parameter y to change from y_0 to y_1. The Jacobian determinant of the transformation is

$$det(\frac{\partial P_1}{\partial P_0}) = (1 - \kappa_1 x_1)|C_1'(s)|dy_1/(1 - \kappa_0 x_0)|C_0'(s)|dy_0$$

For locally volume-preserving transformation, $det(\frac{\partial P_1}{\partial P_0}) = 1$, therefore,

$$(1 - \kappa_1 x_1)|C_1'(s)|dy_1 = (1 - \kappa_0 x_0)|C_0'(s)|dy_0$$

Integrate on both sides, we have

$$(1 - \kappa_1 x_1)|C_1'(s)|y_1 = (1 - \kappa_0 x_0)|C_0'(s)|y_0$$

Let σ represent the local stretch, or the ratio of the curve's speed before and after the deformation:

$$\sigma = \frac{|C_1'(s)|}{|C_0'(s)|}.$$

The solution y_1 is linearly related to y_0 as shown in the following equation,

$$(1 - \kappa_1 x_1)y_1\sigma = (1 - \kappa_0 x_0)y_0 \tag{24}$$

5.1.3 Radial solution

The radial solution is a compromise between the normal and binormal solution. A point P near the spine is expressed as,

$$P = C(s) + r\cos\theta N(s) + r\sin\theta B(s)$$

In the radial solution, the offset distance from the spine is adjusted from r_0 to r_1. The Jacobian determinant of the transformation is then expressed (in r_0 and r_1) as,

$$det(\frac{\partial P_1}{\partial P_0}) = \frac{r_1 dr_1(1 - \kappa_1 r_1\cos\theta)|C_1'(t)|}{r_0 dr_0(1 - \kappa_0 r_0\cos\theta)|C_0'(t)|}.$$

Let $det(\frac{\partial P_1}{\partial P_0}) = 1$. Then solve for h_1, we have

$$|C_1'(t)|(-\frac{2}{3}\kappa_1 \cos\theta r_1^3 + r_1^2) = |C_0'(t)|(-\frac{2}{3}\kappa_0 \cos\theta r_0^3 + r_0^2).$$

Let σ represent the local stretch during deformation:

$$\sigma = \frac{|C_1'(s)|}{|C_0'(s)|}.$$

The solution r_1 is a cubic root of the following equation,

$$(-\frac{2}{3}\kappa_1 \cos\theta r_1^3 + r_1^2)\sigma = (-\frac{2}{3}\kappa_0 \cos\theta r_0^3 + r_0^2). \tag{25}$$

5.2 Implementation and Existence condition

The implementation procedure for 3D stretchable spine curve is very similar to the implementation for non-stretchable spine curve discussed in Section 4.2. However, there are two differences. First, the implementation of the stretchable spine curve is simplified from the requirement of length preservation. Second, the solutions are not only based on the local curvature, but also on the local stretch, which should be considered in an intermediate step as explained in the following section.

5.2.1 Unbending-transfer-bending technique for stretchable spine

Like the case for deformation with non-stretchable spine, the solution provided by Equation 23, 24 and 25 assume that the bending and stretching do not change the local Frenet frame. To support more general bending, we split the computation of the offset distance into three general steps:

1. unbending: first update the offset distance assuming that the initial spine $C_0(s)$ locally becomes a straight line.

2. transfer: rotate the Frenet frame or equivalently compute the local coordinates of P in the rotated Frenet frame and scale the offset distance according to the local stretch parameter.

3. bending: after transferring to the new frame, compute the offset distance assuming that the straight spine bends into $C_1(s)$.

Section 4.2 presents the formulae for steps 1 and 3, as the second step is the same change of basis for all three solutions. Here for stretchable spine, the transfer step is different among the three solutions and we include formulae for all steps.

As for the Normal solution in Section 4.1.1, Equation 11 is limited to cases where the local curvature is changed, but the Frenet frame remains constant. To support non-planar spine curve bending, we provide below its decomposition into normal unbending and bending, which may be combined with the twist-compensation. To solve x_1, we break Equation 11 into two steps:

Normal Unbending: Assume that $C_0(s)$ is first straightened and we solve for the unbending image x_*,

$$x_* = x_0(1 - \frac{1}{2}\kappa_0 x_0). \tag{26}$$

As $\frac{x_*}{x_0} \geq 0$, the condition for a valid solution of r_* to exist is $|\kappa_0 x_0| \leq 2$.

Normal Transfer: Given x_*, we solve for a transferred value x_t based on local stretch,

$$x_t = x_*/\sigma \tag{27}$$

The local normal offset distance is inversely proportional to the local stretch paramter.

Normal Bending: We then bend the straight spine into C_1 and solve for x_1 using x_t:

$$-\frac{1}{2}\kappa_1 x_1^2 + x_1 = x_t. \tag{28}$$

Hence, the closed-form solution for x_1 is

$$x_1 = \frac{1 - \sqrt{1 - 2\kappa_1 x_t}}{\kappa_1}.$$

In order for x_1 to be valid, the existence condition for κ_1:

$$\kappa_1 x_t \leq \frac{1}{2}, \tag{29}$$

and when κ_1 reaches this curvature limit, $x_1 = 2x_t$.

As for the Binormal solution in Section 4.1.2, Equation 12 is limited to cases where the local curvature is changed, but the Frenet frame remains constant. To support non-planar spine curve bending, we provide below its decomposition into normal unbending and bending. To solve y_1, we break Equation 12 into two steps:

Binormal Unbending: Similarly, in Equation 12, we set $\kappa_1 = 0$, $y_* = y_1$. This gives us

$$y_* = (1 - \kappa_0 x) y_0. \tag{30}$$

In order for y_* to be valid, we have $\kappa_0 x \leq 1$.

Binormal Transfer: Given y_*, we solve for a transferred value y_t based on local stretch,

$$y_t = y_*/\sigma \tag{31}$$

The local binormal offset distance is inversely proportional to the local stretch paramter.

Binormal Bending: Let $\kappa_0 = 0$, $y_0 = y_t$ and we have,

$$y_1 = \frac{1}{1 - \kappa_1 x} y_t. \tag{32}$$

In order for y_1 to be valid, we have

$$\kappa_1 x < 1. \tag{33}$$

When κ_1 reaches this curvature limit, y_1 becomes unbounded.

As for the Radial Solution in Section 4.1.3, Equation 13 is limited to cases where the local curvature is changed, but the Frenet frame remains constant. To support non-planar spine curve bending, we provide below its decomposition into radial unbending and bending. To solve r_1, we break Equation 13 into two steps:

Radial Unbending: We first assume that $C_0(s)$ is straightened into a line (i.e. $\kappa_1 = 0$) and solve for the unbending image r_*:

$$r_* = r_0\sqrt{1 - \frac{2}{3}\kappa_0 \cos\theta_0 r_0}. \tag{34}$$

In order for r_* to exist, $\frac{2}{3}\kappa_0 \cos\theta r_0 < 1$. As $\cos\theta$ varies in $[-1, 1]$, an sufficient condition for r_* to exist is $|\kappa_0 r_0| \leq \frac{3}{2}$.

Radial Transfer: Given r_*, we solve for a transferred value r_t based on local stretch,

$$r_t = r_*/\sqrt{\sigma} \tag{35}$$

Different from the normal or binormal transfer step, The local radial offset distance should be inversely proportional to the square root of the local stretch paramter. This is because that the area of the cross section is proportional to r^2 while linear in x or y if the other is unchanged.

Radial Bending: We then bend the straight spine into C_1 and solve for r_1 using r_t:

$$-\frac{2}{3}\kappa_1 \cos\theta r_1^3 + r_1^2 = r_t^2. \tag{36}$$

In order to conclude the existence condition, we normalize the unknown and the coefficients in Eq. 36. Specifically, let $\lambda = \frac{r_1}{r_t}$ and $\alpha = -\frac{2}{3}\kappa_1 r_t \cos\theta$, then Eq. 36 becomes $\alpha\lambda^3 + \lambda^2 = 1$. Let $f(\lambda) = \alpha\lambda^3 + \lambda^2 - 1$, which has two local extrema (minimum at $\lambda_1 = 1$ and maximum at $\lambda_2 = -\frac{2}{3\alpha}$). If $\alpha > 0, \lambda_2 < 0$, then $f(0)f(1) < 0$ and $f' > 0 \in [0, 1]$, and hence there exists a valid solution in $[0, 1]$. If $\alpha > 0, \lambda_2 > 0$, then a valid solution exists only if $f(\lambda_2) > 0$, or

equivalently $\alpha^2 < \frac{4}{27}$. Again since $\cos\theta$ varies in $[-1, 1]$, an sufficient condition for r_1 to exist is $|\alpha| < \frac{2}{3\sqrt{3}}$, or

$$|\kappa_1 r_t| \leq \frac{1}{\sqrt{3}}, \tag{37}$$

and when κ_1 reaches this curvature limit, $r_1 = \sqrt{3} r_t$.

Summary. Compare to non-stretchable spine, the formulae for stretchable spine take the local stretch parameter σ into consideration. In implementation, this stretch parameter scales the normal, binormal, or radial offset distances linearly or sublinearly in addition to the change of basis. As a result, the volume loss or gain due to spine compression or extension is compensated by taking both the local curvature and the local stretch into consideration.

5.2.2 Discretization of stretchable spine curve

Previously we have discussed length-preserving spine curve in Chapter 4, where we basically want a smooth spine and a uniform arc-length or natural parameterization of it. We must be able to compute the arc-length parameter of the closest projection of a point onto the spine. Stretchable spine curve relaxes this requirement. It can be a smooth spine with any parameterization that describes the location of all points on the spine. The choice of representation for the spine is orthogonal to our contribution. Nevertheless, we implement two formulations for the spine curve:

1. A low degree, interpolating polynomial, which we evaluate using Nevilles algorithm [33]. The degree of the spine curve evaluated with Neville algorithm interpolating n control points is at most $n - 1$.

2. A quintic NUBS, which we evaluate using de Casteljaus algorithm. The former one is interpolating and convenient for simple spines (up to 5 control points). The latter has more flexibility and local control: it can be used to model more complex curves and also closed loops.

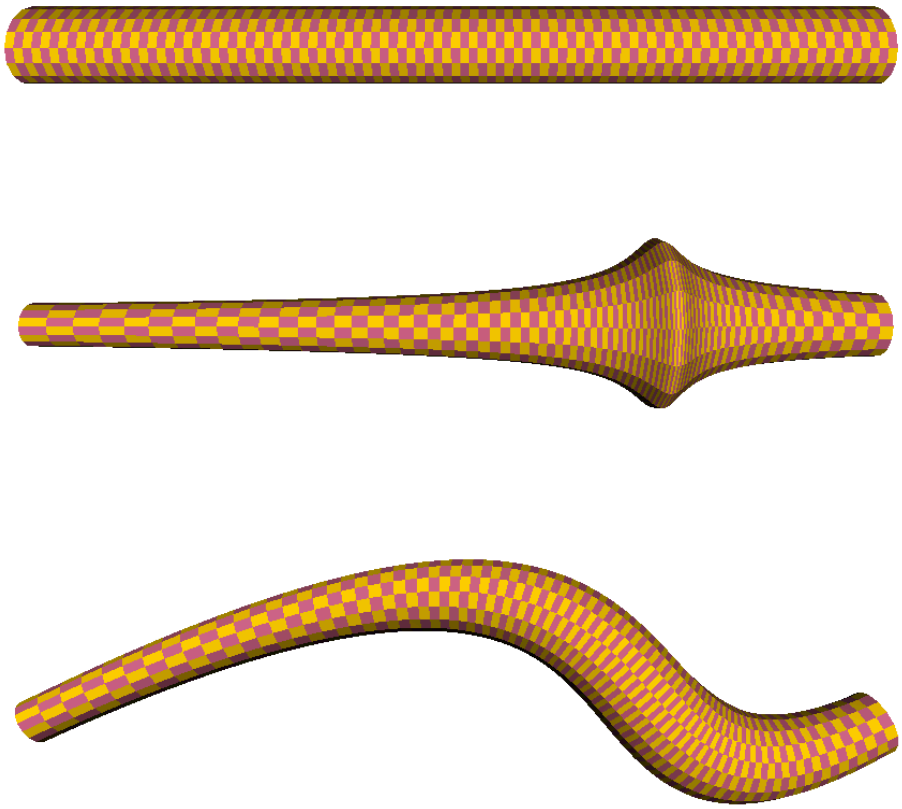


Figure 29: Stretching, compressing and bending a cylindrical surface. The spacing along the spine is shown on the cylinder.

5.3 Results and Analysis

Figure 29 shows the deformation of a cylindrical surface driven by a 3D spine curve. The curve is modeled by a polynomial using Neville algorithm that interpolates four control points. The user can drag any control point to bend or stretch the 3D curve.

As the user deforms the spine curve, the cylindrical surface aligned with the spine is deformed accordingly. The top of Figure 29 shows the original cylindrical surface which is a evenly spaced straight tube registered to a uniformly sampled line with

four control points. In the center, we keep the control points aligned, but move the two interior control points: the center-left one is moved apart from the leftmost one and towards the center-right one. As a result, the cylindrical surface is tapered on the left and compressed on the right to form a ring ridge as shown in the figure.

The user can also move the control points in space to bend the spine curve as desired. The bottom of Figure 29 shows the deformation result with local volume preservation: The volume of any wedge formed by a quad on the cylindrical surface and the corresponding line segment on the spine is preserved by our radial deformation algorithm presented in this section.

Figure 30 shows the results of applying the normal, binormal and radial solutions to the cylindrical surface model corresponding to the center image of Figure **??**, where the difference among the three solutions is the most obvious. The normal and binormal schemes stretch the cylindrical surface in directions orthogonal to each other, while the radial solution stretches the cylindrical surface isotropically within the cross section. Although the normal and the binormal solutions look othorgonal to each other, the binormal solution is not a 90^o rotation of the normal one if there is a bending in addition to stretching.

Figure 31 shows the results of thinning a bunny: We keep the control points aligned, but move the two interior control points apart from each other to stretch the belly of the bunny. The normal solution makes the bunny looks thinner in the normal direction while the binormal solution creates such an effect in the other direction. The radial solution is a compromise between the previous two.

Figure 32 shows the deformation of a bunny driven by a 3D spine curve. Again, the spine curve is modeled by a B-spline using Neville algorithm that interpolates four control points.

The bunny is registered to the spine curve. As the user deforms the spine curve, the bunny is deformed accordingly. The top of Figure 29 shows the original bunny

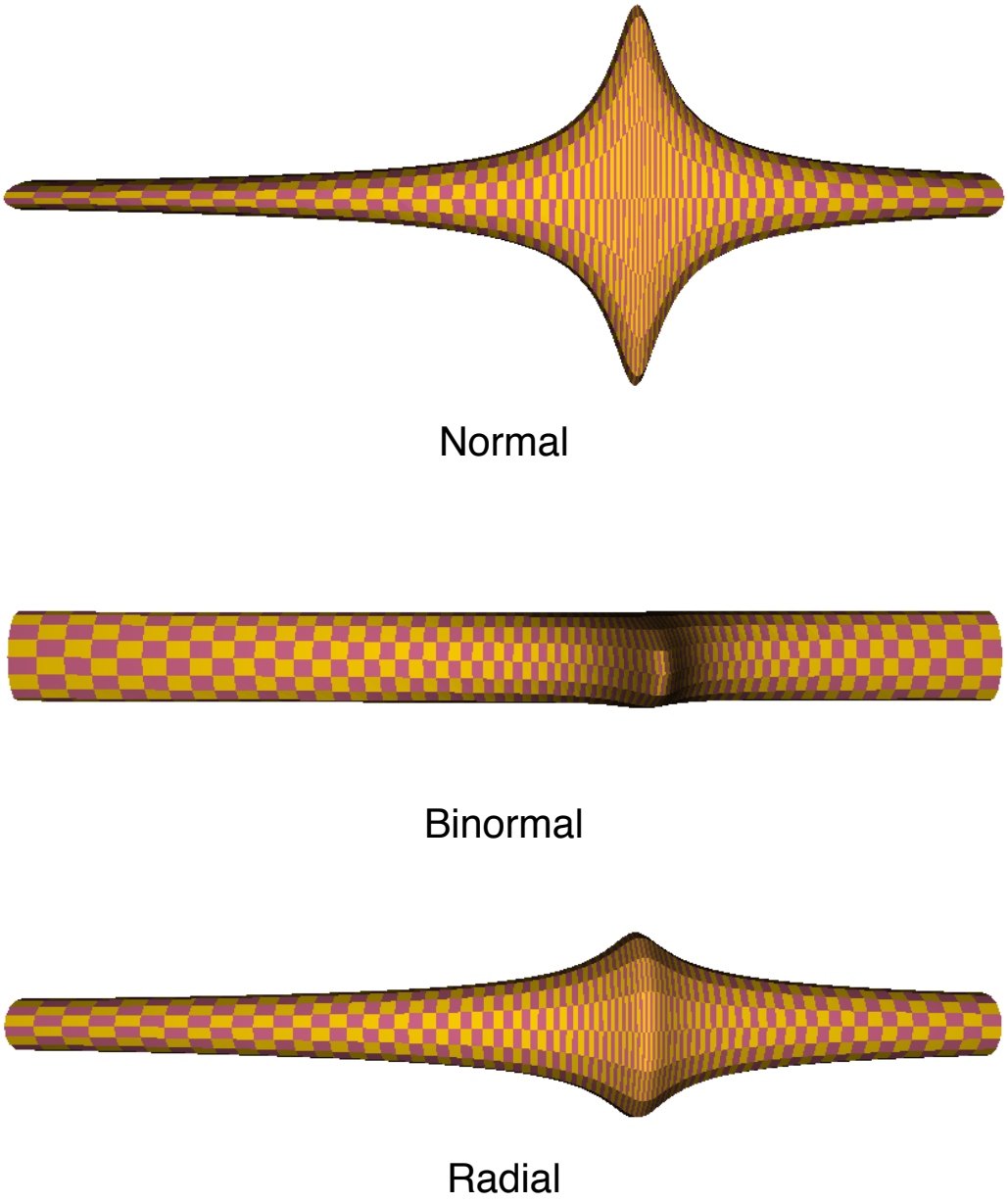


Figure 30: Results of applying the normal, binormal and radial schemes on cylindrical surface.

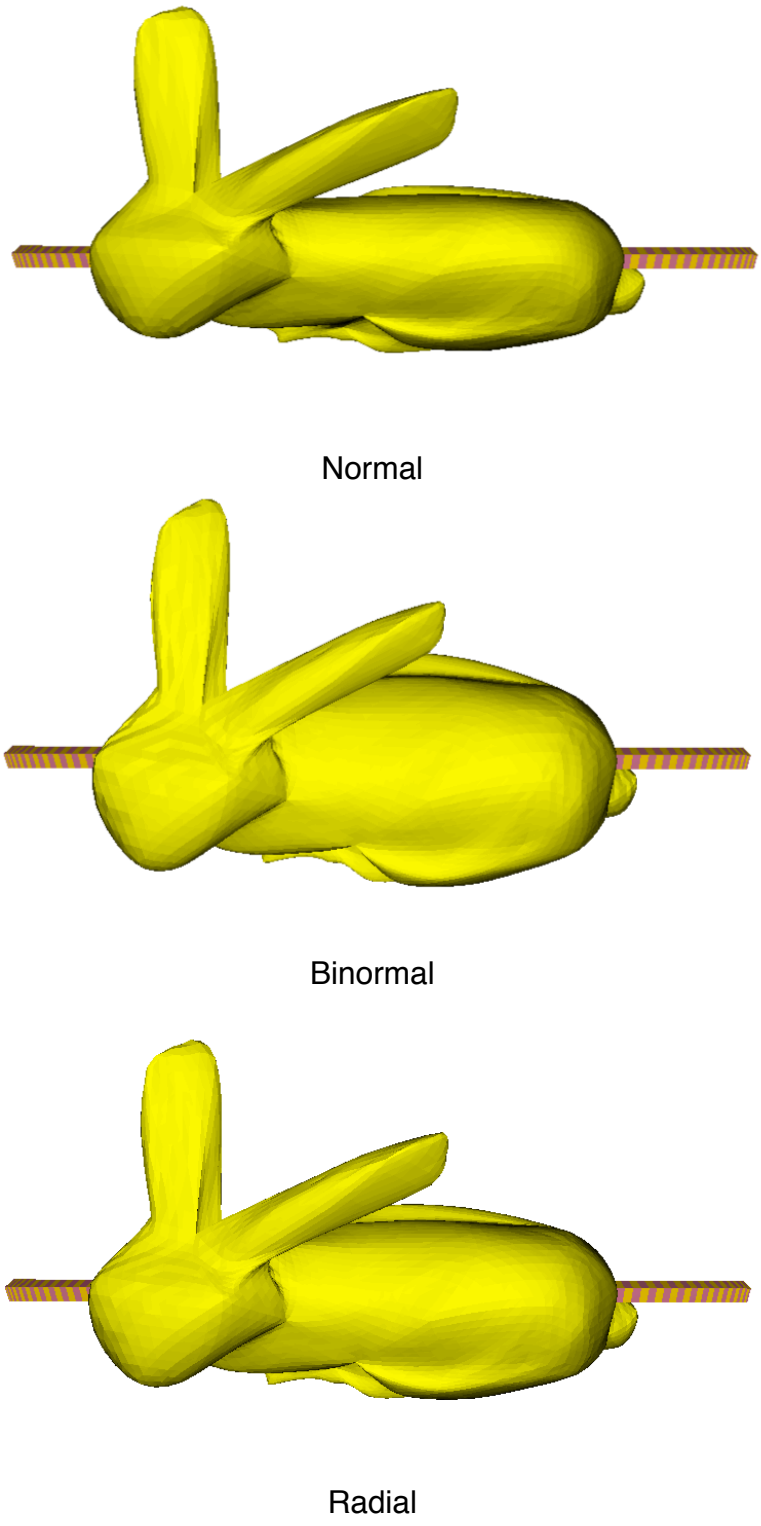


Figure 31: Results of applying the normal, binormal and radial schemes on thinning the bunny.

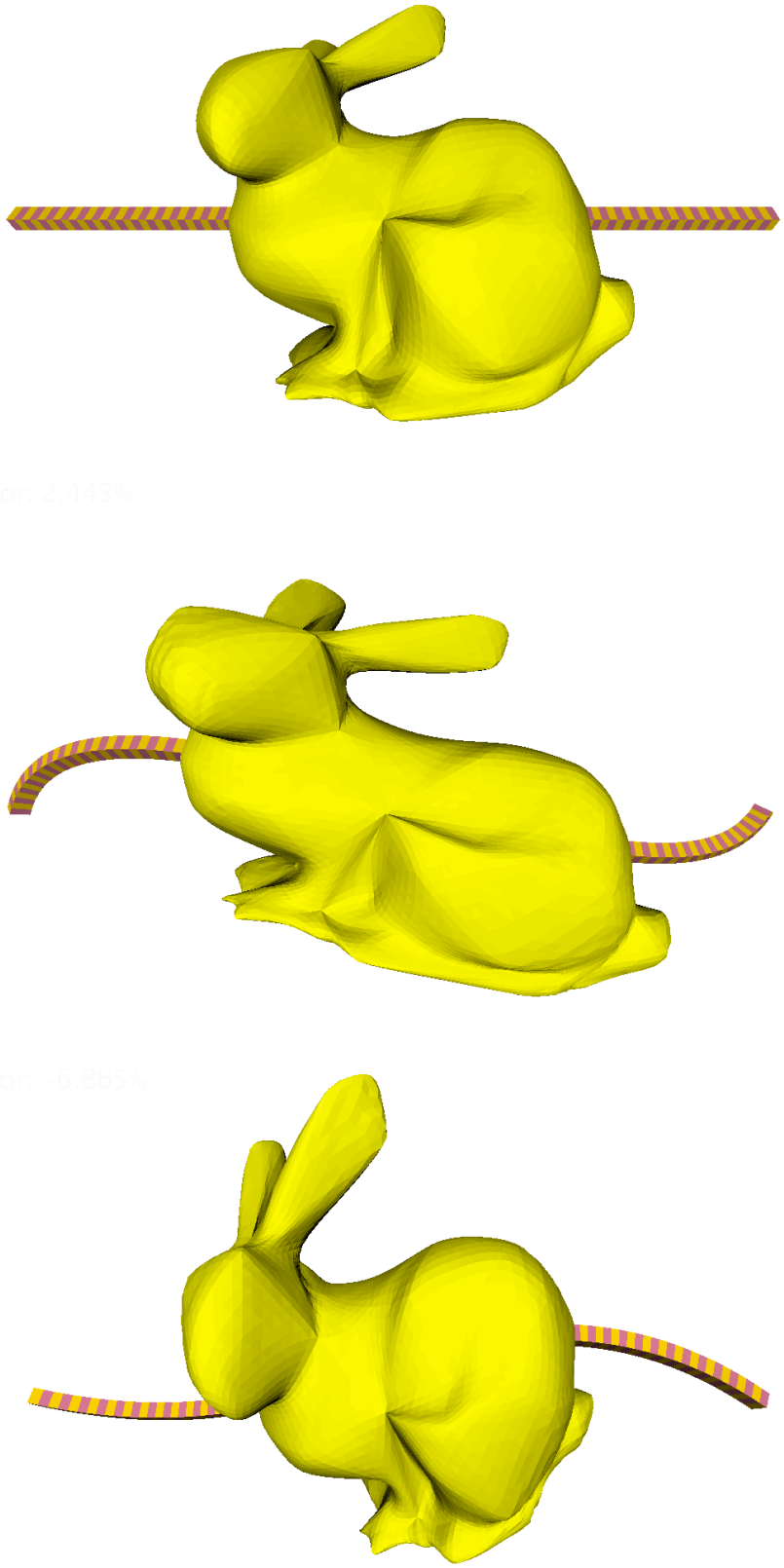


Figure 32: Deformation of a bunny driven by stretching, compressing and bending a 3D spine curve.

registered to a evenly spaced line with four control points. In the center of the figure, we move the two interior control points apart to create a stretch. At the bottom of Figure 32, we move two interior control points near each other to create a compression. As shown in the two figures, the bunny deforms while being stretched or compressed by the spine.

CHAPTER VI

FORMULATION FOR DEFORMATION WITH SPINE SURFACE

In spine-based deformation, the proxy spine can also be a surface. For spine surface, we need three parameters in the registration step in principle. This includes the distance h from the point in space to the closest projection on the surface, and the surface parameters u and v of the closest projection. Compared to the spine curve, there is an additional parameter to determine the projection on spine surface. However, there is only one normal at each projection and therefore one degree of freedom to move the point. In order to preserve the local volume of any chunk in space, we adjust the normal offset distance h based on the local curvatures of the surface.

This chapter extends the formulation of spine curve to spine surface. Section 6.1 discusses the formulation and derivation of local volume preserving deformation driven by a spine surface. Section 6.2 introduces the implementation and existence condition. Section 6.3 presents the results and analysis.

6.1 Spine Surface Deformation

Let $S(u, v)$ denote a two-dimensional sub-manifold, parameterized by u, v, of three-dimensional Euclidean space. Let P denote a offset point from S,

$$P(u, v, h) = S(u, v) + hN(u, v).$$

Therefore the 3×3 Jacobian matrix is,

$$\frac{\partial P}{\partial(u,v,h)} = \begin{bmatrix} S_u(u,v) + hN_u(u,v) \\ S_v(u,v) + hN_v(u,v) \\ N(u,v) \end{bmatrix}.$$

The determinant of the above Jacobian is,

$$det(\frac{\partial P}{\partial(u,v,h)}) = (S_u(u,v) + hN_u(u,v)) \times (S_v(u,v) + hN_v(u,v)).$$

Let $m = m(u,v)$ denote the local mean curvature and $g = g(u,v)$ denote the local Gaussian curvature. Due to that the mean curvature is the divergence of the normal field at $S(u,v)$, and the Gaussian curvature is the cross product of the Hessian of the normal, the above equation can be reduced to the following:

$$det(\frac{\partial P}{\partial(u,v,h)}) = (1 - 2hm + h^2 g)|S_u \times S_v|$$

Note that $(1 - 2hm + h^2 g)$ is equivalent to $(1 - hk_1)(1 - hk_2)$ where k_1 and k_2 are principle curvatures.

During the deformation driven by the spine surface, we allow the parameter h to change from h_0 to h_1. In order for the deformation to be locally volume preserving, we have,

$$det(\frac{\partial P_1}{\partial P_0}) = 1$$

Therefore,

$$(1 - 2h_1k_1 + h_1^2 g_1)|S_{1u} \times S_{1v}|dh_1 = (1 - 2h_0k_0 + h_0^2 g_0)|S_{0u} \times S_{0v}|dh_0$$

Integrate on both sides, we have

$$(h_1 - h_1^2 m_1 + \frac{h_1^3}{3} g_1)|S_{1u} \times S_{1v}| = (h_0 - h_0^2 m_0 + \frac{h_0^3}{3} g_0)|S_{0u} \times S_{0v}| \qquad (38)$$

Therefore, the updated offset distance h_1 is the solution of the above cubic equation.

Let σ denote the local stretch parameter for the spine surface,

$$\sigma = \frac{|S_{1u} \times S_{1v}|}{|S_{0u} \times S_{0v}|}. \tag{39}$$

Then Equation 38 becomes

$$h_1 - h_1^2 m_1 + \frac{h_1^3}{3} g_1 = (h_0 - h_0^2 m_0 + \frac{h_0^3}{3} g_0)/\sigma. \tag{40}$$

Therefore the solution of the updated offset distance h_1 is the root of the above cubic equation.

6.2 Implementation and Existence condition

The solution provided by Equation 40 specifies the relationship of the offset distance before and after the deformation: The offset distance changes according to the local gaussian curvature g, the local mean curvature k and the local stretch parameter σ. In the deformation driven by spine surface, the offset distance is always along the normal. Since the normal for registration and for updating the offset distance are the same, we do not need to have the transfer step to change the basis: Equation 40 can be used to compute the solution directly. However, we can still interpret the solver that solves Equation 40 into the following three steps:

1. unbending: first update the offset distance assuming that the initial spine $S_0(u, v)$ locally becomes a flat surface.

2. transfer: scale the unbending image according to the offset distance.

3. bending: after transferring to the new frame, compute the offset distance assuming that the flat surface bends into $S_1(u, v)$.

The design of the decomposition in the spine-surface deformation is similar to that presented by Section 5.2. However, the unbending, transferring and bending steps are different from the solutions in spine-curve deformation. First, there is only one

degree of freedom for offsetting the point in the normal direction. This is a important difference from the deformation driven by a spine curve where a family of solutions exists. Second in the transferring step, we need to consider the stretching in two directions on the spine surface instead of one on the spine curve.

Here we include the formulae for all three steps of deformation driven by a spine surface. We provide below its decomposition into unbending, transferring and bending, which may be combined with more accurate projection introduced in Chapter 7. To solve h_1, we break Equation 40 into the following three steps:

Unbending: Assume that $S_0(u, v)$ is first locally flattened and we solve for a temporary value h_*,

$$h_* = h_0(1 - m_0 h_0 + \frac{g_0}{3} h_0^2). \tag{41}$$

As $\frac{h_*}{h_0} \geq 0$, the condition for a valid solution of h_* to exist is

$$m_0 h_0 - \frac{g_0}{3} h_0^2 \leq 1.$$

Transferring: Given h_*, we solve for a transferred value h_t based on the local stretch parameter,

$$h_t = h_*/\sigma \tag{42}$$

The offset distance is inversely proportional to the local stretch paramter.

Bending: We then bend the flat spine surface into $S_1(u, v)$ and solve for h_1 using h_t:

$$h_1 - m_1 h_1^2 + \frac{g_1}{3} h_1^3 = h_t. \tag{43}$$

In order to solve h_1 in Equation 43, we use a change of variables. Let $x = \frac{h_1}{h_t}$, $m_* = m_1 h_t$, $g_* = g_1 h_t^2$ and we have

$$x - m_* x^2 + \frac{g_*}{3} x^3 = 1.$$

Take derivatives on both sides,

$$\frac{1}{g_*} - \frac{2m_*}{g_*} x + x^2 = 0.$$

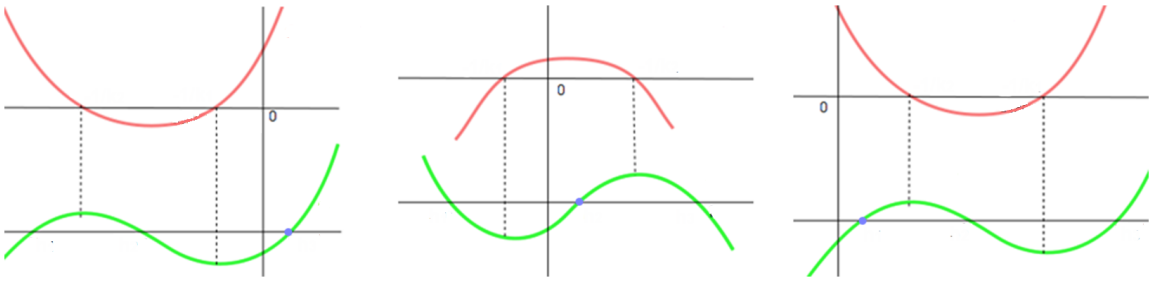


Figure 33: Identify the valid root in a cubic equation.

It is not difficult to see that the two inflection points of the cubic polynomial on the left hand side of the cubic equation are

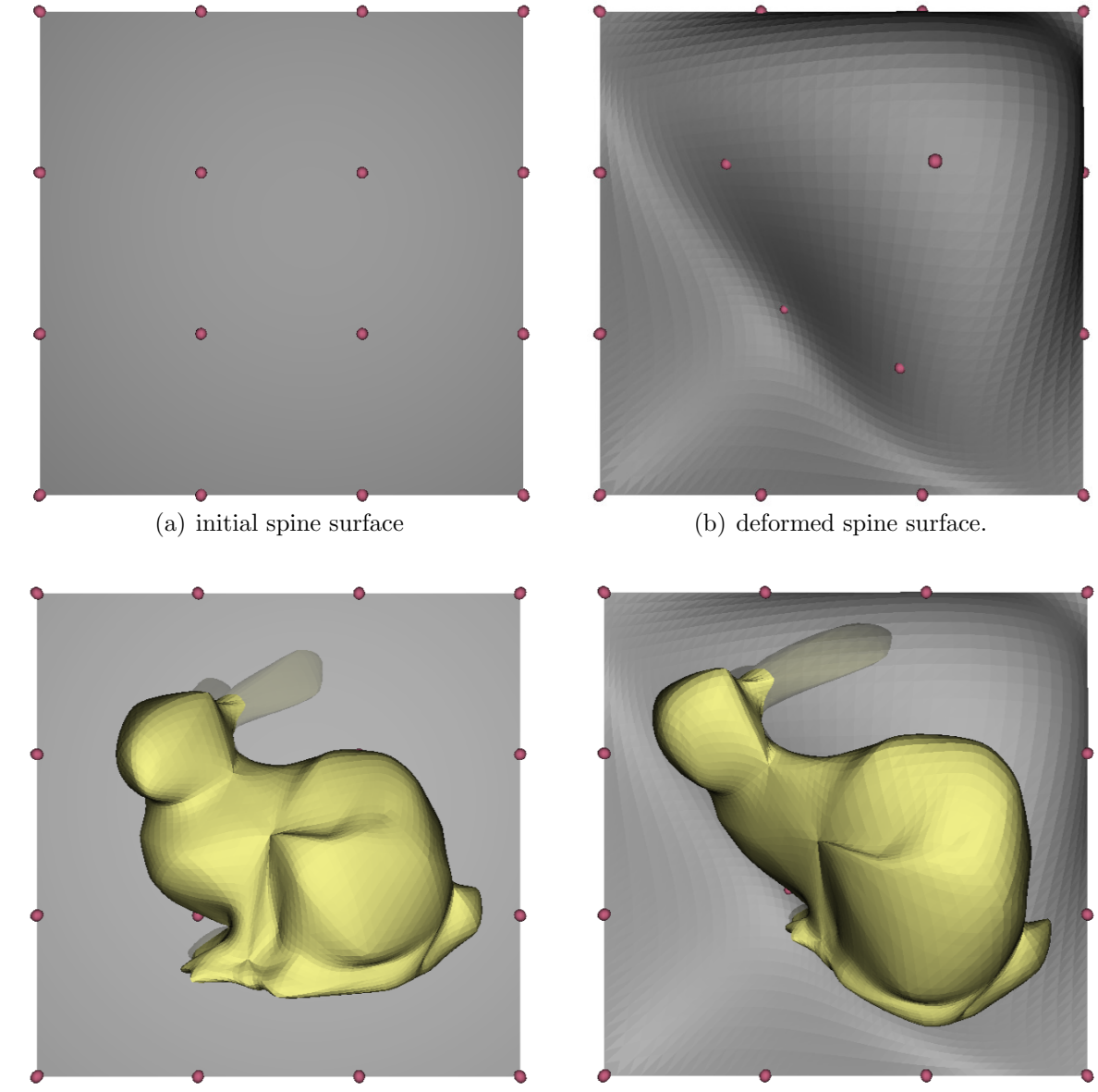
$$x_{i1} = \frac{1}{k_1 h_t}, \quad x_{i2} = \frac{1}{k_2 h_t},$$

where k_1 and k_2 are the principle curvatures. In order for the valid solution of h_1 or x to exist, Figure 33 highlights the valid root in blue in various curvature conditions. To summarize, if $2\sqrt{m_*^2 - g_*} - m_* > 3(m_* - \sqrt{m_*^2 - g_*})$, then there is a valid, unique real root in $[0, \frac{1}{\sqrt{m_*^2 - g_*} - m_*}]$. Otherwise, no valid real root exists and the solver should return the maximum offset distance that is free from a local self intersection.

6.3 Results and Analysis

Figure 34 shows the steps of deforming a bunny driven by a spine surface. The surface is modeled by a bi-variate polynomial surface patch using Neville algorithm [33] that interpolates 4×4 control points. The user can drag any control point to bend or stretch the surface. The bunny is first registered to the initial surface as shown by Figure 34(c), and reconstructed from the deformed spine surface as shown by Figure 34(d).

The top image in Figure 35 shows dragging the control points of the spine surface. The points are within the same plane. We move all the control points at the boundary and interior to create stretching and compression at different parts of the spine surface.



(a) initial spine surface

(b) deformed spine surface.

(c) original bunny registered to the initial spine surface

(d) deformed bunny driven by the spine surface

Figure 34: Overview of the deformation driven by a spine surface.

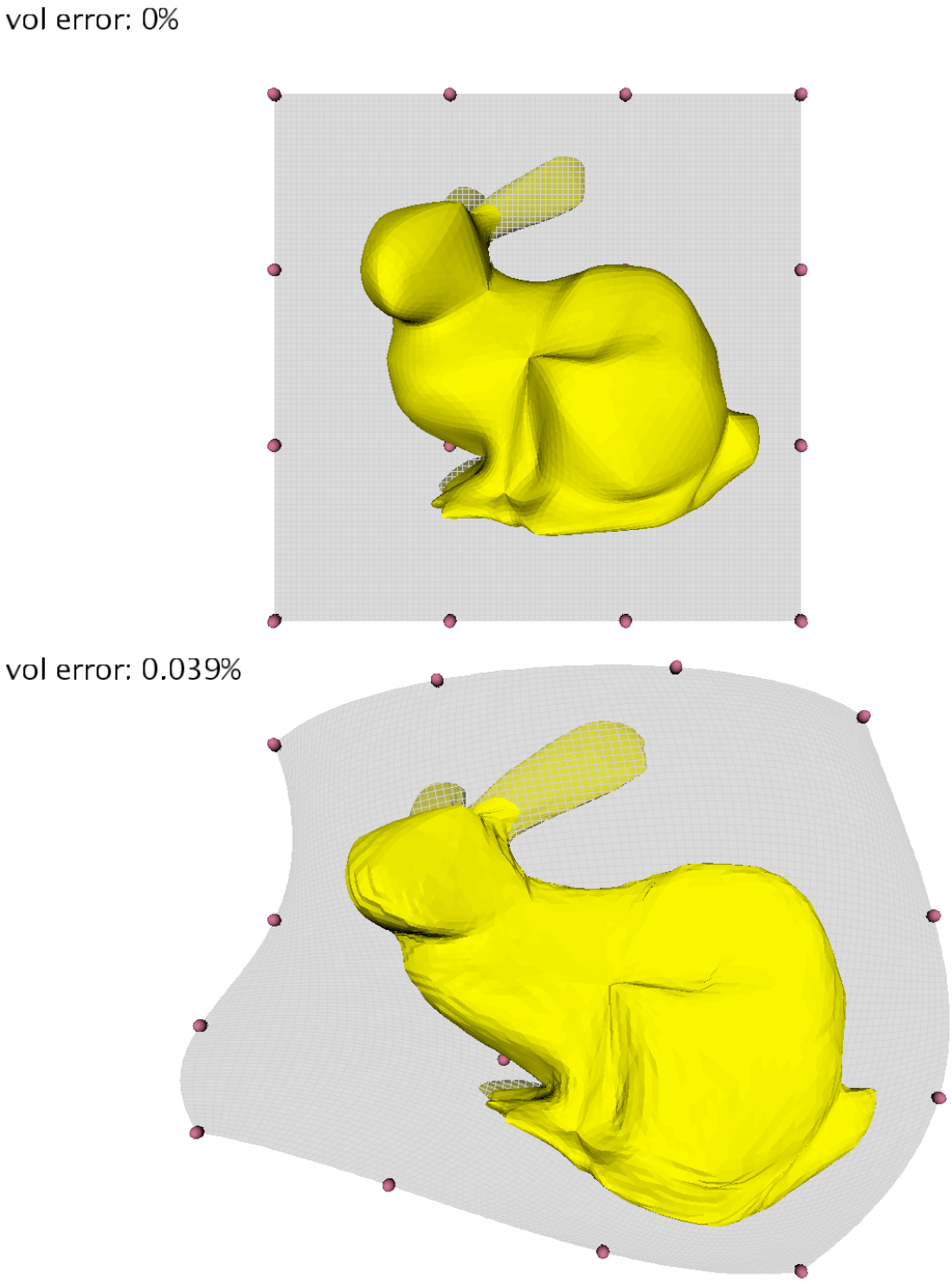


Figure 35: Stretching and compression of a bunny driven by a spine surface with volume preservation.

As a result, the bunny deforms while being stretched or compressed with its volume preserved locally and globally.

Figure 36 shows the deformation of a set of spheres. The top image in Figure 36 shows the original spheres and the spine surface. The spine surface is initially flat, evenly spaced, with the 3×3 spheres placed and registered onto it. As the user deforms the spine surface, the spheres are deformed accordingly, as shown at the center and the bottom of Figure 36. In addition to moving all the control points at the boundary and the interior to create stretching and compression at different parts of the spine surface, we also move control points in or out of the viewing plane to create bending. As a result, the set of spheres deform while being stretched, compressed and bent with their volumes preserved for individual spheres.

To better show the volumetric error for each sphere in the example shown by Figure 36, Figure 37 uses color mapping to map volume loss to blue and volume gain to red in a color ramp. The top of Figure 37 is without offset distance correction, and shows significant volume change in the set of spheres. At the bottom, the volume of each sphere is preserved with the offset distance corrected according to the local curvatures and the stretch parameter in Equation 40.

Figure 38 shows the deformation of a bunny driven by a spine surface which is not flat at the beginning. Different from the example in Figure 34, two control points of the spine surface are initially dragged out of the plane. The bunny is first registered to this curved surface as shown by Figure 38(c). Then the spine surface deforms as the two control points are pushed inside. The bunny is reconstructed from the deformed spine surface as shown by Figure 38(d). This creates an effect that the bunny sinks into the spine surface.

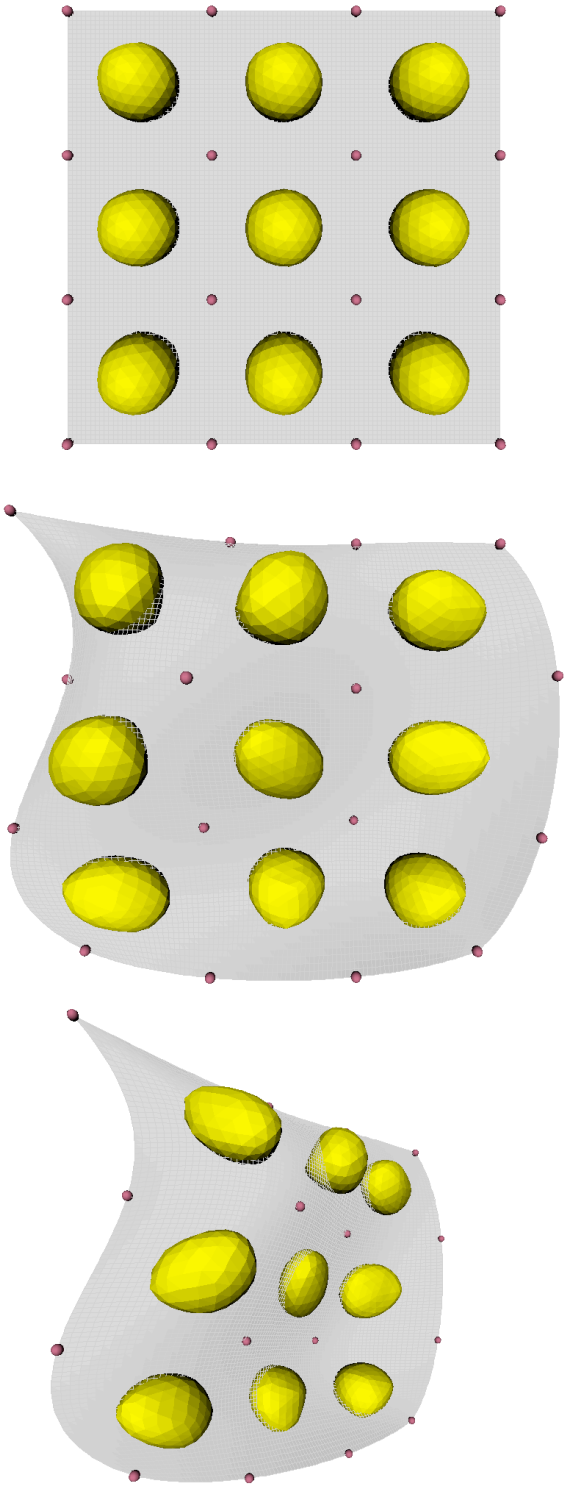


Figure 36: Deformation of spheres driven by a spine surface.

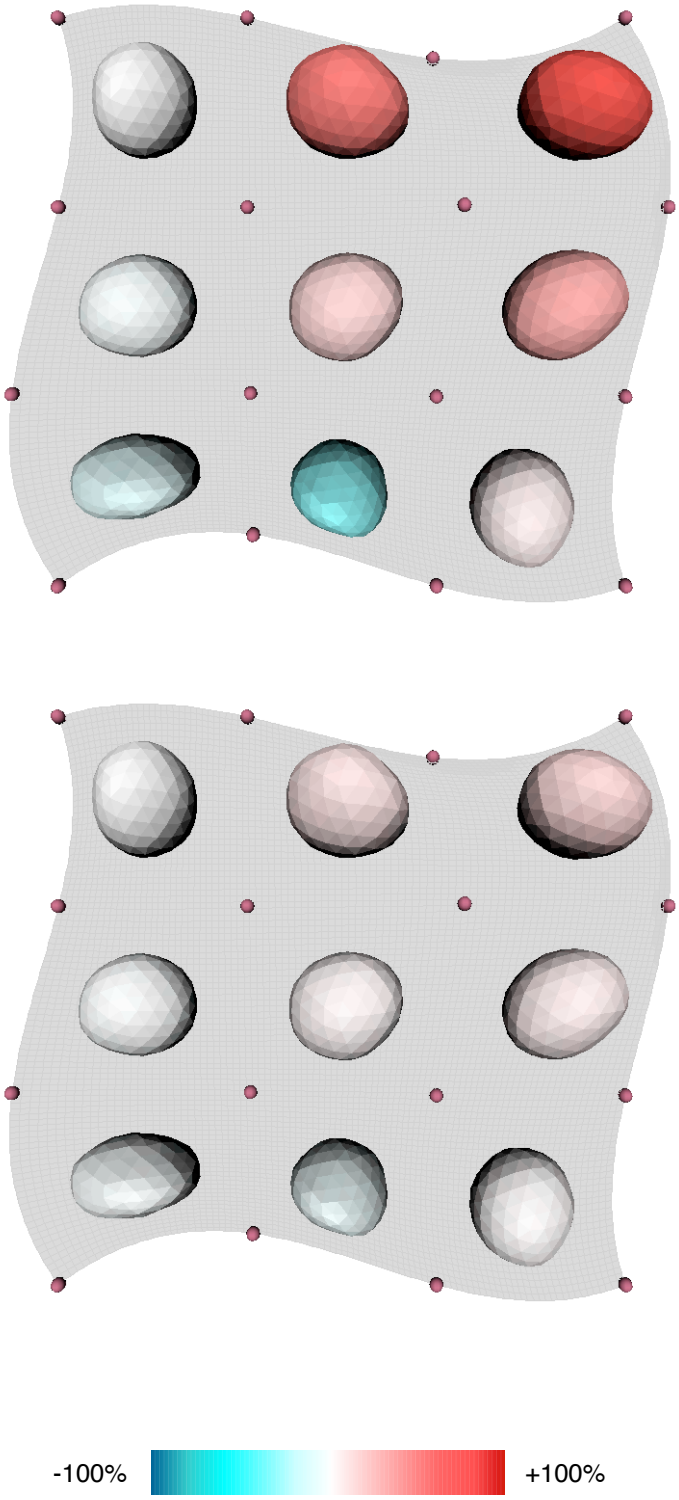
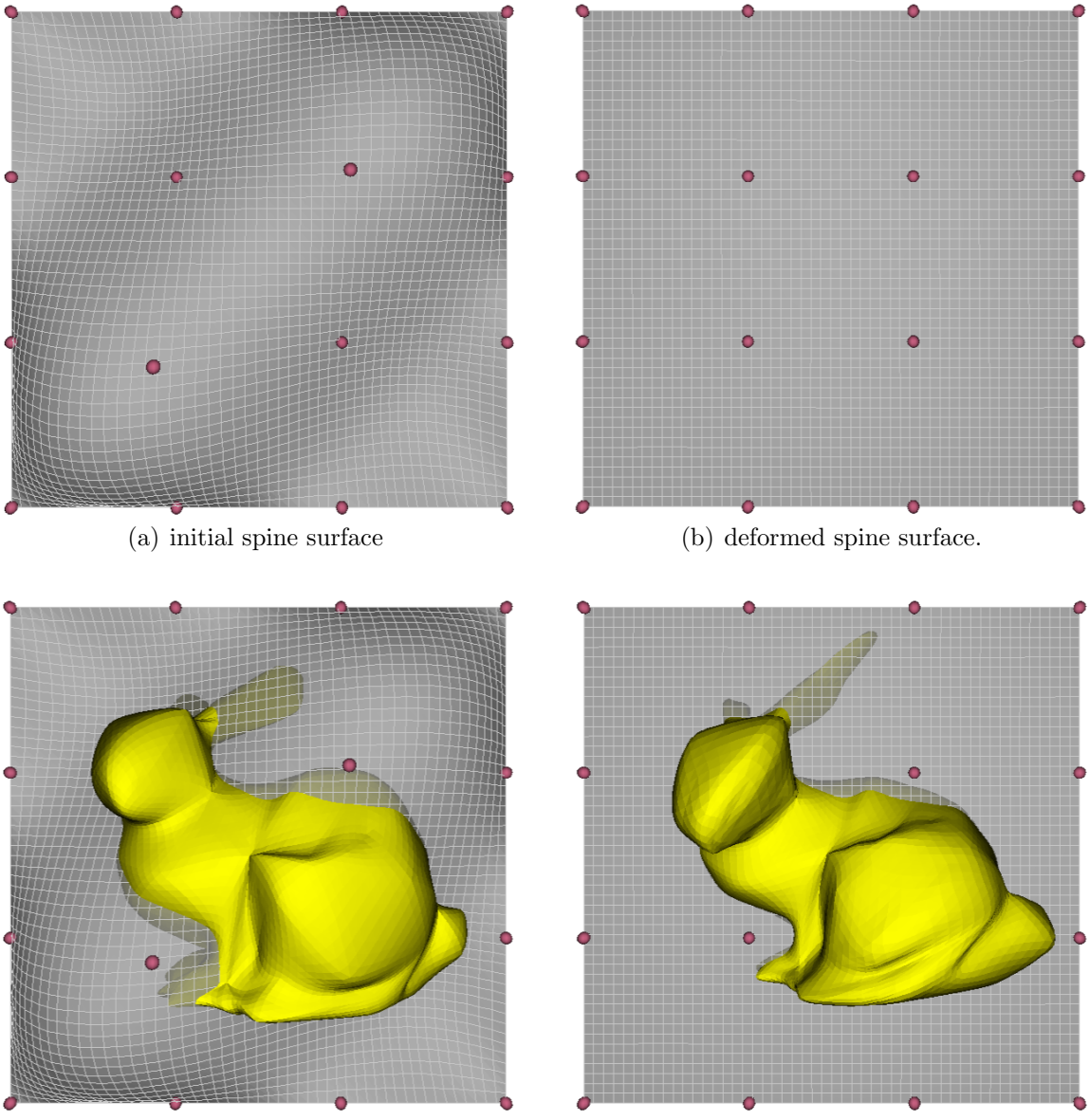


Figure 37: Using color mapping, we see how the volume of each sphere changes.



(a) initial spine surface

(b) deformed spine surface.

(c) original bunny registered to the initial spine surface

(d) deformed bunny driven by the spine surface

Figure 38: Deformation by a spine surface which is initially curved.

CHAPTER VII

ACCURACY AND SAMPLING

Chapter 5 and Chapter 6 present methods for correcting specific local parameters to prevent the local volume change. So far, we have not discussed much about issues related to the sampling of the input mesh and spine. Usually, sampling is the factor directly determine the accuracy of registration and reconstructed results, while using the same deformation algorithm. However, using unbounded number of samples could result into computational issues such as prolonged running time, exhausted memory, and insufficient digits to represent the change between consecutive samples. Therefore, it is considered worthwhile to design a interpolation method that improves the accuracy with limited number of samples available. Two related problems, the sampling of the spine and the sampling of the input object, are discussed in this chapter. The computation of projection in registration is also discussed with spine sampling.

7.1 Problem description

On one hand, we already know that the parameter s in the closest project $C(s)$ can denote the arc-length parameter in a non-stretchable spine, or any one-to-one parameter in a stretchable spine (For 3D spine surface, we need two parameters, u, v). A simple method to compute s is to search for a closest vertex on the discrete spine representation, and use the index of this vertex as the discretized parameter for reconstruction after deformation. However, this introduces a problem when different points in space that should have different closest projections registered to one vertex: In this case, the computed projection is not correct, hence reconstruction, even without bending produces a different model. This problem can cause serious errors and artifacts in the reconstructed result. Therefore, we propose a method to compute the

more accurate projection in Section 7.2.

On the other hand, our mathematical formulation to spine-based deformation is essentially a space transformation independent from the shape's topology. Note that this transformation is not an affine mapping as a triangle does not map into a triangle by this formulation. Therefore, insufficient sampling of the input object (e.g. a model represented by a coarse triangle mesh) can introduces much error. We address this problem by using a subdivision that produces a fine mesh, so that the error between the mapping of a triangle (by deformation) and the triangle spanning the mappings of its 3 vertices is small. To study the impact of sampling of the input shape on this error, Section 7.2 and Section 7.3 also present experimental methods and results with different level of subdivisions.

7.2 Proposed approaches

Recall that in the registration and reconstruction explained in Chapter 1: For each point P_0 on the original shape S_0, we compute its closest projection, $C_0(s)$, on the spine curve C_0, and the offset vector to this closest projection. Given s and h, we reconstruct the deformed point P_1 as an updated offset from $C_1(s)$.

Previously, we mentioned a method to sample the spine curve into a oriented set of n vertices $\{C_i^j; j = 0, 1, \cdots, n; i = 0, 1\}$ so as to search for the closest vertex among them. Let C_0^k be the closest vertex to P_0. The simple method uses C_0^k to approximate $C_0(s)$ in registration and C_1^k to approximate $C_1(s)$ in reconstruction.

This section introduces a more accurate projection method in which the closest vertex on the polygonal approximation of C_0 is used to approximate $C_0(s)$. Assume that Q_0 is the closest projection on the k^{th} edge $C_0^k C_0^{k+1}$ of the polygonal approximation of C_0:

$$Q_0 = C_0^k + aC_0^k C_0^{k+1}, 0 < a < 1.$$

Then the curve parameter is expressed in both k and a. We use linear interpolation

as it captures the accurate projection of a point onto the polygonal approximation of the spine. We use the same index and interpolation parameter to compute Q_1 for C_1:

$$Q_1 = C_1^k + aC_1^kC_1^{k+1},$$

which is used as the anchor point for computing P_1 in 3D spine-driven deformation.

Similarly for deformation driven by a spine surface S, we assume that Q_0 is the closest projection on the triangle mesh approximation of S_0. Let S_0^i, S_0^j, S_0^k denote vertices of the triangle containing Q_0 including the case that the projection is on the edge or vertex of the triangle. The true closest projection $S(u, v)$ is approximated by Q_0 as follows,

$$Q_0 = \lambda^1 S_0^i + \lambda^2 S_0^j + \lambda^3 S_0^k,$$

where

$$0 < \lambda^1 < 1, 0 < \lambda^2 < 1, 0 < \lambda^2 < 1, \lambda^1 + \lambda^2 + \lambda^3 = 1$$

are the barycentric coordinates of Q_0 with respect to the triangle $S_0^iS_0^jS_0^k$. Therefore the surface parameter is stored as $\{i, j, k, \lambda^1, \lambda^2, \lambda^3\}$, which are used to compute the anchor point Q_1 for the spine surface S_1 after deformation in constant time,

$$Q_1 = \lambda^1 S_1^i + \lambda^2 S_1^j + \lambda^3 S_1^k.$$

7.2.1 More accurate curvature, normal estimators

Section 7.2 computes the closest projection on the polygonal approximation, instead of the closest vertex among vertices of the polygonal approximation. Since we compute the more accurate projection, the normal and curvature at Q is interpolated as the normal and curvature at neighboring vertices of Q, after computing different normals at different vertices of the spine.

For the normal interpolation on 3D spine curve represented by point sequence, we compute the angle θ between N^k and N^{k+1}. Then, the interpolated normal should

have the angle $a\theta$ from N^k to itself. Details for normal interpolation are as follows:

$$\sin\theta = \|N^k \times N^{k+1}\|$$

$$\cos\theta = N^k \cdot N^{k+1}$$

$$\theta = \arctan(\sin\theta, \cos\theta)$$

$$N^Q = N^k \cos(a\theta) + (N^k \times N^{k+1}) \times N^k \frac{\sin(a\theta)}{\sin\theta}$$

The curvature at Q is linearly interpolated from the curvatures at C^k and C^{k+1}:

$$\kappa^Q = (1-a)\kappa^k + a\kappa^{k+1}$$

Though the spline may not have linearly varying curvature, it is possible to locally approximate any smooth curve with clothoid or Euler spirals [6].

For the normal interpolation on 3D spine surface represented by triangle mesh, we use the barycentric coordinates $\{\lambda^1, \lambda^2, \lambda^3\}$ to interpolate the normal and curvature at Q as follows:

$$N^Q = \lambda^1 N^i + \lambda^2 N^j + \lambda^3 N^k,$$

$$N^Q = N^Q / \|N^Q\|.$$

$$\kappa^Q = \lambda^1 \kappa^i + \lambda^2 \kappa^j + \lambda^3 \kappa^k.$$

Note that we need to interpolate normals in order to reconstruct the point after deformation. P_1 is reconstructed by offsetting a point at Q in the direction of N^Q with distance h_1 computed by Equation 3.

7.3 Results and analysis

We implement both the simple and the more accurate projection with corresponding estimators to demonstrate the benefit of using the more accurate projection. Overall experimental results show that the volume is more precisely preserved with the accurate projection, which also produces better visual results in the reconstructed meshes.

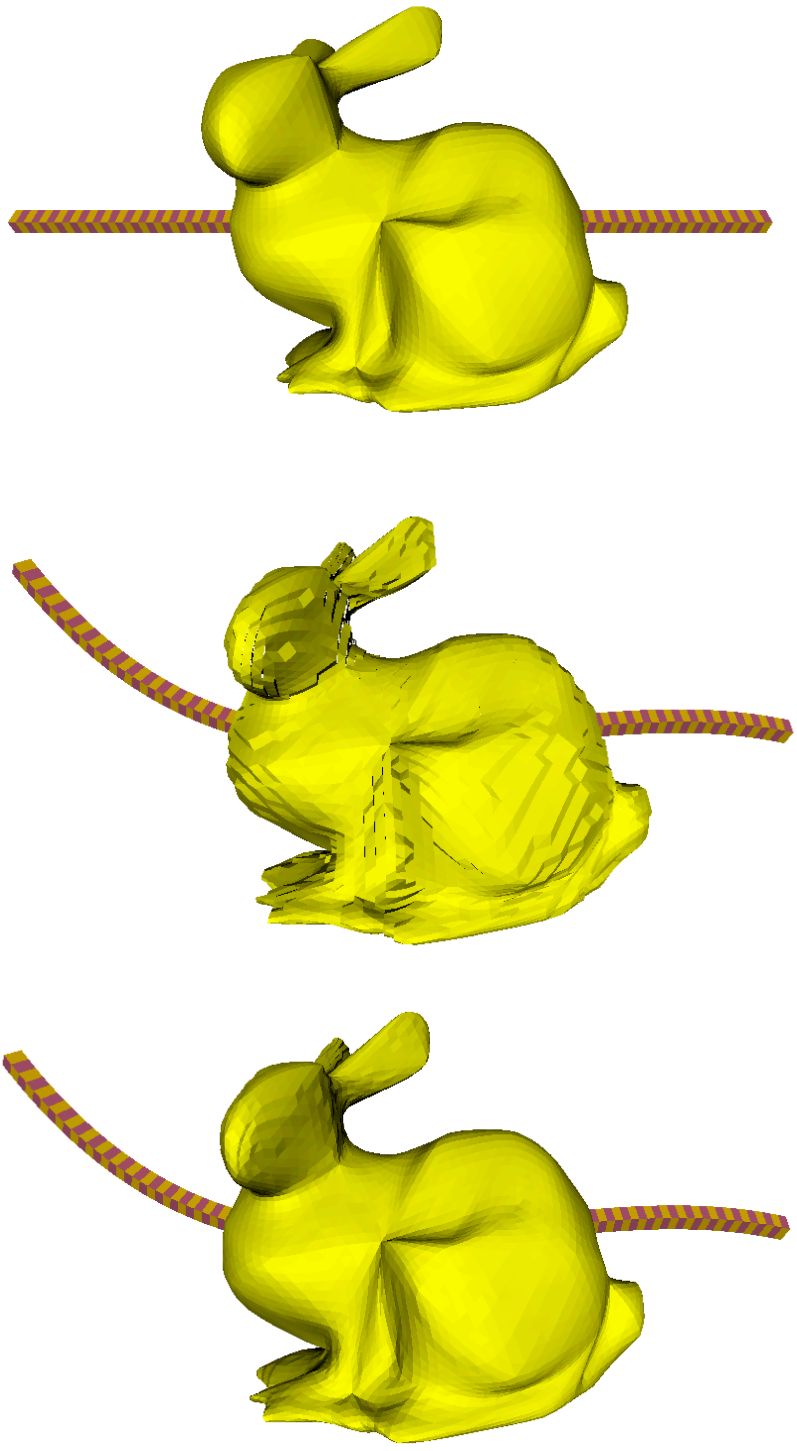


Figure 39: Results of using simple and more accurate projections in deformation with a finely sampled spine curve.

Figure 39 shows the results of deforming a bunny with a finely sampled spine curve. The top of Figure 39 shows the original bunny which is a smooth triangle mesh and the initial spine. The center of Figure 39 shows the deformation result by the simple projection implementation, for which the reconstructed mesh has visible artifacts, roughness. The relative volumetric error is around 2% in this simple implementation compare to 0.2% in the implementation with more accurate projection, shown at the bottom of Figure 39. The reconstructed mesh with the more accurate projection appears smooth as the original and has few visible artifacts.

To better show the difference between the two implementations, Figure 40 shows the results of deforming a bunny with a coarsely sampled spine curve. The top of Figure 40 shows the original bunny and the initial spine which has only 10 sample points. The center of Figure 40 shows the deformation result by the simple projection implementation, for which the reconstructed mesh has been 'sliced' into sections. The relative volumetric error is very large. In the figure shown at the bottom of Figure 40, the reconstructed mesh with the more accurate projection has some banding artifacts, but the overall result is much better compared to the above.

Figure 41 shows the results of different projection implementations with the spine surface. The top of Figure 41 shows the result of the simple implementation in which the closest projections are approximated by sample vertices of the surface. The bottom figure shows the result of the accurate implementation in which the closest projections are closest projections on the approximating triangle mesh. The corresponding reconstructed mesh has a higher quality.

Figure 42 shows bending a subdivision mesh at different resolutions. From left to right we show the deformation results of uncorrected solution, the radial, normal and binormal solutions on different levels of a subdivision mesh. From top to bottom, the number of vertices are 32, 482, 1922. The increase of the subdivision depth greatly decreases the relative volume errors of the three solutions (from 2.5% to 0.001%),

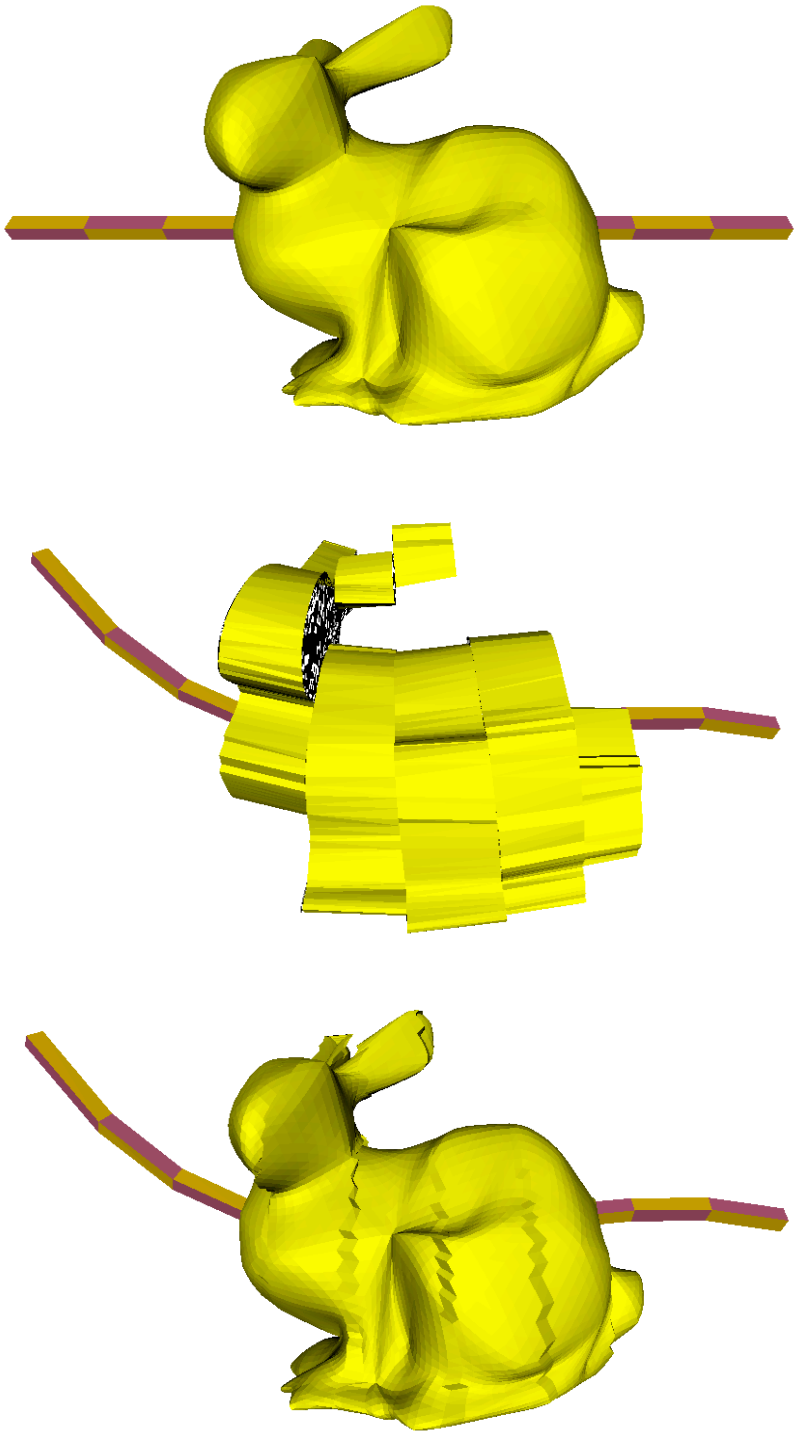


Figure 40: Results of using simple and more accurate projections in deformation with a coarsely sampled spine curve.

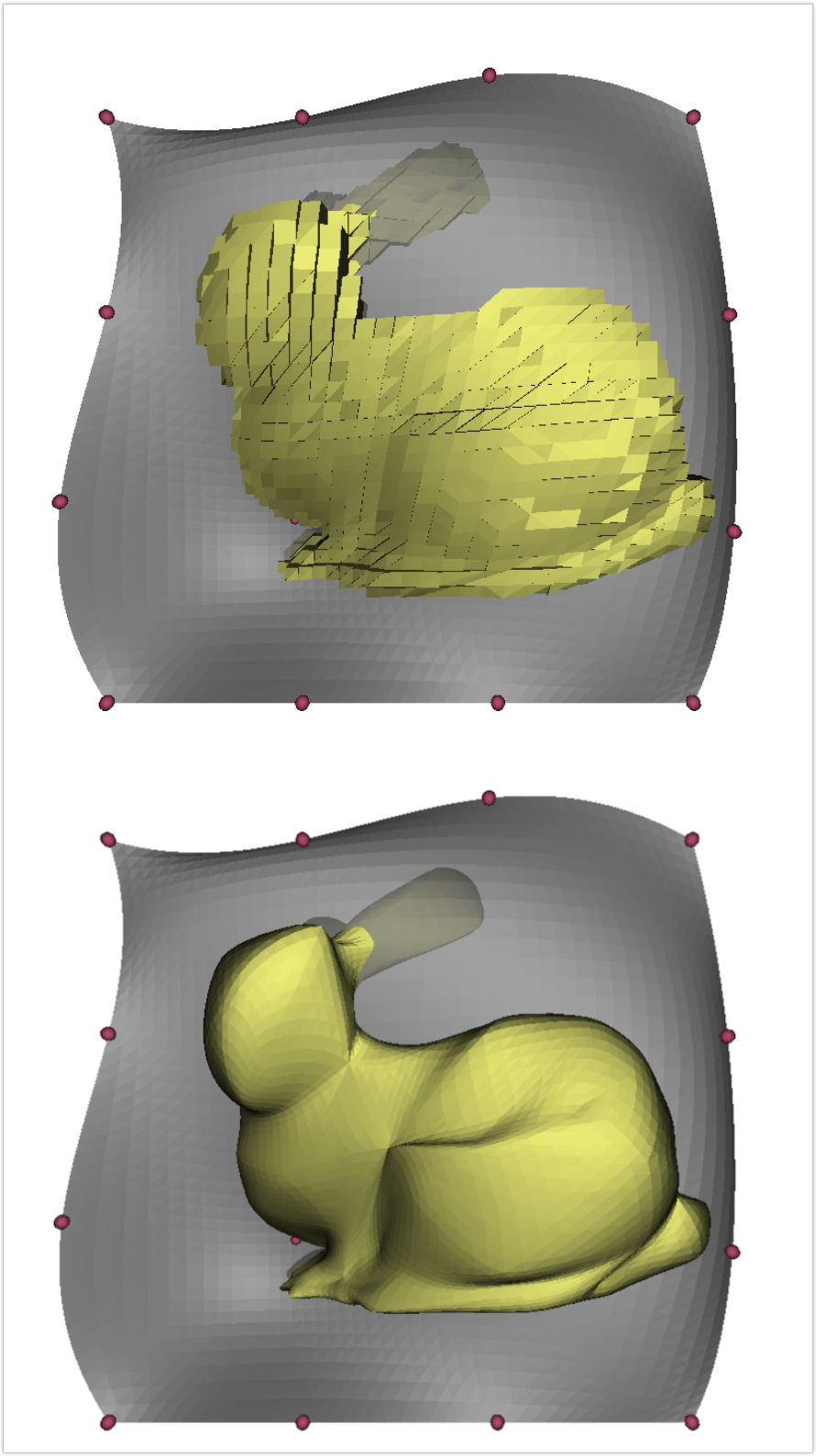


Figure 41: Results of using simple and more accurate projections in deformation with a spine surface.

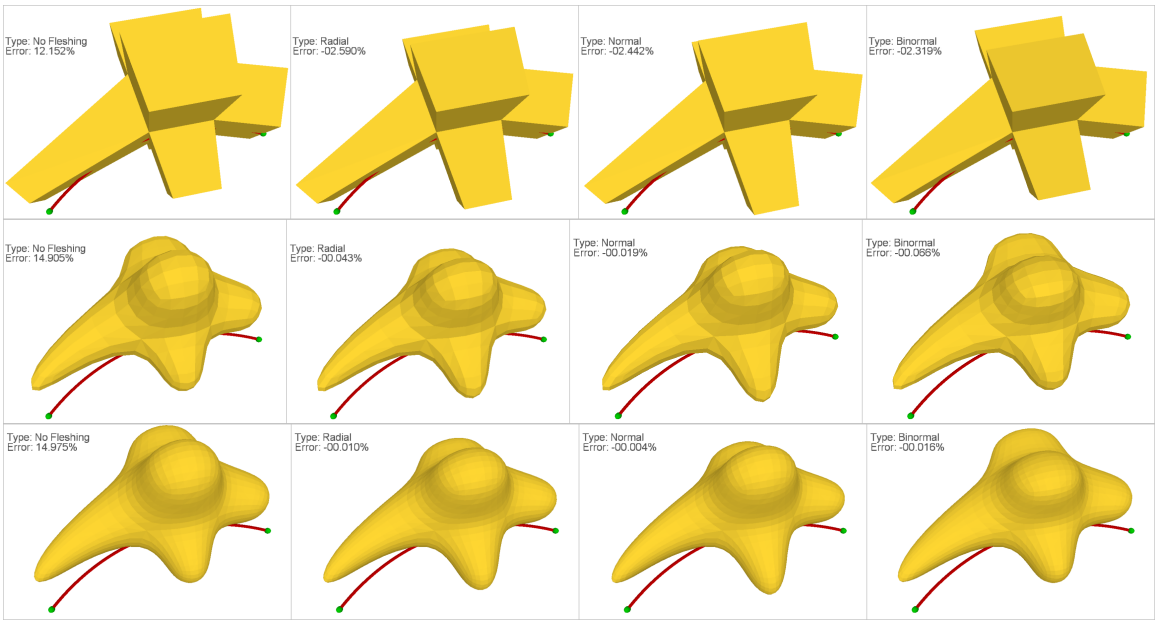


Figure 42: Deformations of a subdivision mesh at different levels of subdivisions.

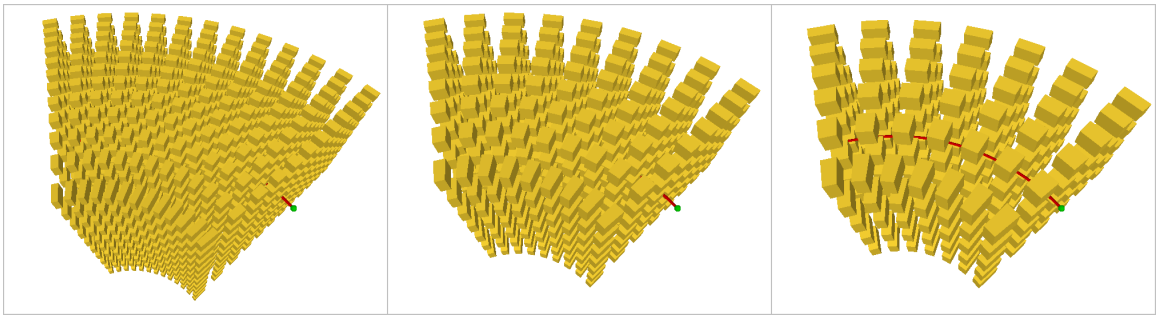


Figure 43: Bending a cloud of cubes at different, initial uniform sizes.

normal, binormal and radial. On the contrary for bending without correction, the relative volume error increases slightly (from 12% to 15%). This shows that the volume preserving solutions give accurate total volume-preserving results for high resolution meshes.

In fact, our objective is not only to preserve the total volume, but to preserve the local volume for each small chunk of the solid. Hence, the proper measure of volume error that should be used to demonstrate the benefit of the corrections is to report the average of the absolute volume errors of the small chunks. Figure 43 shows bending a cloud of cubes at different sizes. From left to right, the original cube sizes are 15,

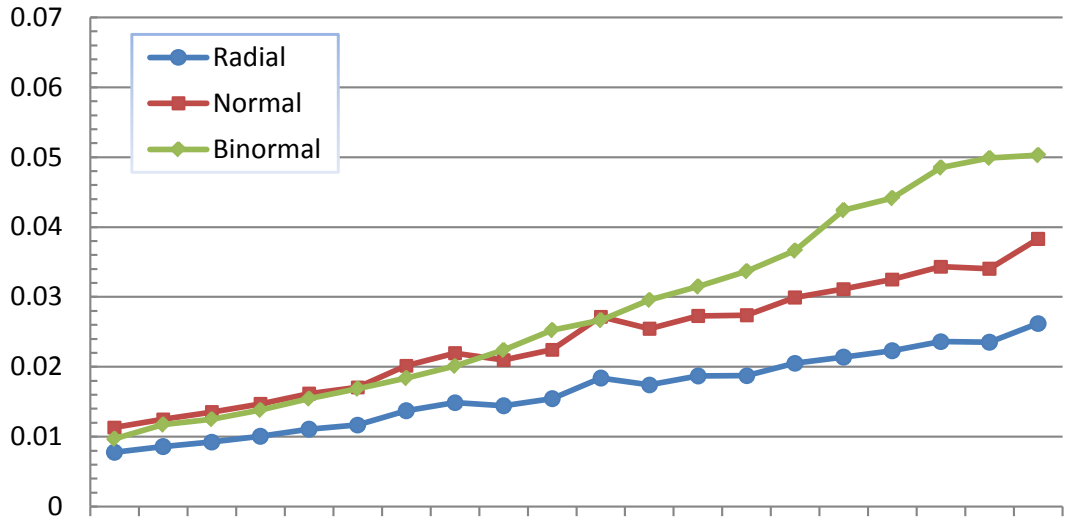


Figure 44: Plot of the percentage mean absolute error versus the cube size.

22, 30. We compute the volume of each cube deformed by the spine. The relative error for each cube is computed as

$$\epsilon = (v - v_0)/v_0,$$

where v is the volume after bending, v_0 is the initial volume.

We report the percentage mean absolute value, ϵ_{mean}, of the relative errors for all cubes. Figure 44 shows the plot of ϵ_{mean} versus the cube size. In general, the relative error scales with the cube size in all three solutions for local volume preservation. The volume error reported for large cubes comes from approximating the curved shape of the deformed cube by a polyhedron that interpolates the mappings of the vertices of the initial cube.

CHAPTER VIII

RELATION TO PHYSICAL REALISM

In Chapter 4, we present three solutions for deformation driven by a 3D spine curve with local volume preservation. In 3D, we use the curve parameter s and the updated offset vector within the cross section to reconstruct the point after deformation. In order to preserve the local volume of any chunk in space, we want to adjust the offset vector by adding a displacement. Assume that this displacement should be orthogonal to the tangent. There are two degrees of freedom to move the vertex. We have introduced the closed-form solutions for the corresponding two degrees of freedom, *normal* and *binormal*. We also introduce another affine-like solution, *radial*, which can be regarded as a compromise between the normal and binormal solutions.

There remains a important question that which one of the solutions simulates the real-world deformation behavior. For example, when the bending is gentle, the 'meat' on the concave side may slightly bulge in the normal direction. In this case, the normal solution captures the deformation result. However, when the bending is sharp, the 'meat' may move sideways in the binormal direction in order to compensate the volume change. In this case, the binormal solution captures a more accurate deformation result. Moreover, material with isotropic property may not stretch along a specific direction within the cross-sectional plane. And the radial solution may better capture the deformation result.

Therefore, the answer to this question depends on several factors including the deformation magnitude (or the strain of the deformation) and the stuffed material's physical structure, properties. Though a comprehensive discussion on these factors is outside of the scope of this thesis, this chapter summarizes the family of spine-driven,

local-volume-preserving solutions by revisiting the formulation previously proposed. Specifically, Section 8.1 provides the insights on the two basis bending modes and suggests a scheme to combine the two solutions. Section 8.2 presents the combined results in a chart for future research work on comparing with various types of deformation with real-world material.

8.1 Basis bending modes: Normal and Binormal

Recall that in the normal and binormal solutions, a point on the initial solid S_0 is expressed as

$$P_0(s, x, y) = C_0(s) + x_0 N_0(s) + y_0 B_0(s),$$

where $N_0(s)$ and $B_0(s)$ are the unit normal and binormal at $C_0(s)$. For the normal solution, we allow the parameter x to change from x_0 to x_1, so that the point on the deformed solid S_1 is expressed as

$$P_1(s, x, y) = C_1(s) + x_1 N_1(s) + y_0 B_1(s),$$

for which the closed form solution for x_1 is the root of the following quadratic equation,

$$x_1 - \kappa_1 x_1^2/2 = x_0 - \kappa_0 x_0^2/2.$$

For the binormal solution, we allow the parameter y to change from y_0 to y_1, so that the point on the deformed solid S_1 is expressed as

$$P_1(s, x, y) = C_1(s) + x_0 N_1(s) + y_1 B_1(s).$$

for which the closed form solution for y_1 is the root of the following linear equation,

$$(1 - \kappa_1 x) y_1 = (1 - \kappa_0 x) y_0.$$

It is possible that the offset point moves in a direction which is a combination of normal and binormal. In this case, both x and y should be updated. The corresponding deformed point is

$$P_1(s, x, y) = C_1(s) + x_1 N_1(s) + y_1 B_1(s).$$

Let $det(\frac{\partial P_1}{\partial P_0}) = 1$ (the local volume preserving constraint) we have,

$$(1 - \kappa_1 x_1)\partial x_1 \partial y_1 = (1 - \kappa_0 x_0)\partial x_0 \partial y_0.$$

Assuming that ∂x and ∂y are independent, integrate on both sides to obtain,

$$(x_1 - \frac{1}{2}\kappa_1 x_1^2)y_1 = (x_0 - \frac{1}{2}\kappa_0 x_0^2)y_0. \tag{44}$$

Adding the constraint for local volume preservation $det(\frac{\partial P_1}{\partial P_0}) = 1$ is not enough to solve both x_1 and y_1. Therefore, we need to introduce a another parameter, ϕ, to denote the direction in which to move the point to compensate the volume change.

8.2 Problem with combining two basis bending modes

This section introduces the combination of two basis bending mode towards a family of solutions for deformation with local volume preservation. As shown by Equation 44 in Section 8.1, it is not enough to solve both x_1 and y_1 when introducing the local volume preserving constraint. Therefore, we predetermine a parameter, denoted by ϕ, to specify the direction in which to move the point:

$$x_1 = x_0 + l\cos\phi$$

$$y_1 = y_0 + l\sin\phi$$

Equation 44 becomes,

$$(-\frac{1}{2}\kappa_1 l^2\cos^2\phi + (1 - \kappa_1 x_0)l\cos\phi + x_0 - \frac{1}{2}\kappa_1 x_0^2)(y_0 + l\sin\phi) = (x_0 - \frac{1}{2}\kappa_0 x_0^2)y_0$$

In unbending, setting $\kappa_1 = 0$ gives

$$l^2\sin\phi\cos\phi + l(y_0\cos\phi + x_0\sin\phi) + \frac{1}{2}\kappa_0 y_0 x_0^2 = 0 \tag{45}$$

In bending, setting $\kappa_0 = 0$ gives the following equation,

$$\begin{aligned} &-\frac{1}{2}\kappa_1\cos^2\phi\sin\phi l^3 + (-\frac{1}{2}\kappa_1 y_0\cos^2\phi + (1 - \kappa_1 x_0)\sin\phi\cos\phi)l^2 \\ &+((1 - \kappa_1 x_0)y_0\cos\phi + (1 - \frac{1}{2}\kappa_1 x_0)x_0\sin\phi)l - \frac{1}{2}\kappa_1 x_0^2 y_0 = 0. \end{aligned} \tag{46}$$

Notice that when $\phi = 0^o$, Equation 46 becomes

$$-\frac{1}{2}\kappa_1 l^2 + (1 - \kappa_1 x_0)l - \frac{1}{2}\kappa_1 x_0^2 = 0,$$

which is the same as the normal bending case shown by Equation 11. However, when $\phi = 90^o$, Equation 46 becomes

$$(1 - \frac{1}{2}\kappa_1 x_0)l - \frac{1}{2}\kappa_1 x_0 y_0 = 0, \tag{47}$$

which is not exactly the same as the binormal bending case shown by Equation 12: Replacing y_1 by $y_0 + l$ would arrive

$$(1 - \kappa_1 x_0)l - \kappa_1 x_0 y_0 = 0. \tag{48}$$

The difference between Equation 48 and Equation 47 indicate that the assumption from which Equation 44 is derived may be incorrect. In another words, ∂x and ∂y are not independent in combining two basis bending modes together. If this assumption does not hold, then it is unsure wether a closed form solution for l exists to compensate the local volume change by offsetting in an arbitrary direction specified by ϕ.

8.3 Solution for a compromise between two bending modes

Section 8.2 has introduces a formulation for combining two basis bending modes by introducing a parameter ϕ, which denotes the offset direction for local volume compensation. However, the closed form solution for the offset distance l proved to be incorrect: For example when $\phi = 90^o$. So how can we combining the two basis bending modes? Is there a closed-form solution for a compromise between them?

Answering the first question is relatively easy. We can always cascade the normal and binormal solutions to produce a mapping that is also local volume preserving. The Jacobian determinant of the concatenated solution is the product of the Jacobian determinants of the decomposed solutions. For example, one may first solve for an offset distance, Δx, in the normal direction by deforming the spine from C_0 to $C_{\frac{1}{2}}$,

which is defined as a intermediate spine between C_0 and C_1. We assume that C_0 is able to smoothly evolve into C_1. Then, one can solve for an offset distance in the binormal direction, Δy, by deforming the spine from $C_{\frac{1}{2}}$ to C_1. However, the ratio of Δy to Δx is initially unknown and may require a iterative process if we want the constraint $\frac{\Delta y}{\Delta x} = \tan\phi$ to hold in the final solution.

Though we are not able to provide a closed-form solution for the offset distance l in a arbitrary direction, the answer to the second question, "Is there a closed-form solution for a compromise between them?", is "Yes". Previously in Section 4.1.3 we have shown the radial solution, which is exactly a compromise between the normal and binormal solutions. In the radial solution, the offset distance is $l = r_1 - r_0$; the offset direction is $\tan\phi = \frac{y_0}{x_0}$. The ratio, $\frac{y_1}{x_1}$, after deformation remains the same as $\tan\phi$ by definition. Therefore, we can have a closed-form solution in which the offset direction ϕ is spatially varying, depending on the relative position of the point with respect to the intersection of the spine with the crossection. In conclusion, the radial solution is not a combination of the normal and the binormal solutions, but a compromise between the two with an additional assumption of the spatially varying offset direction.

To help characterizing different deformation behaviors driven by a 3D spine curve, the chart in Figure 47 provides the cross-sectional plots of the deformation result computed by the normal, radial and binormal methods with two different curvatures. The original example is illustrated by Figure 29. Here we provide more crossectional plots with the curvature increasing from left to right. This shows different trends of the deformation which may be used to compare with real, spine-driven deformation behaviors.

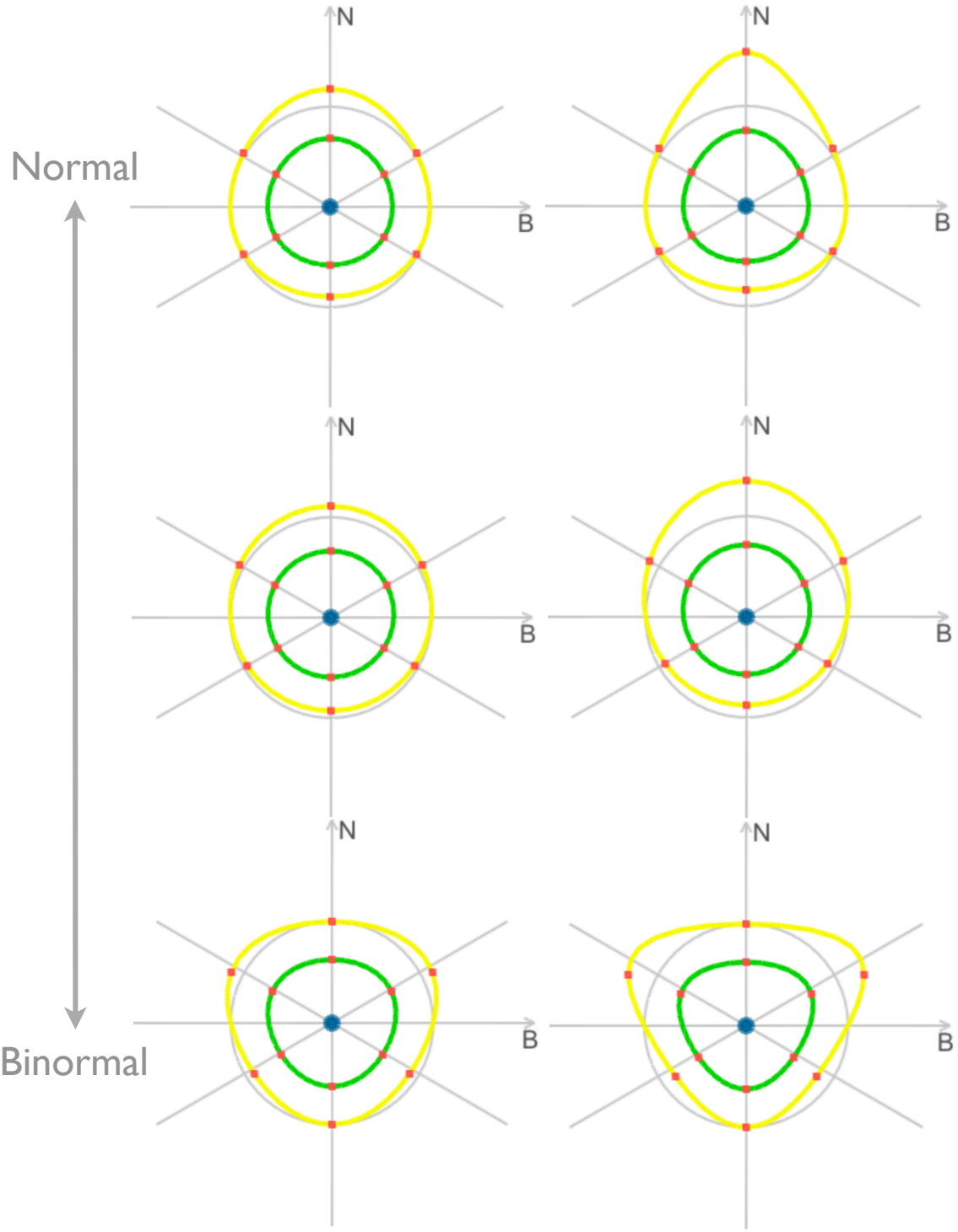
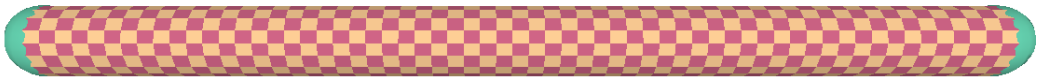
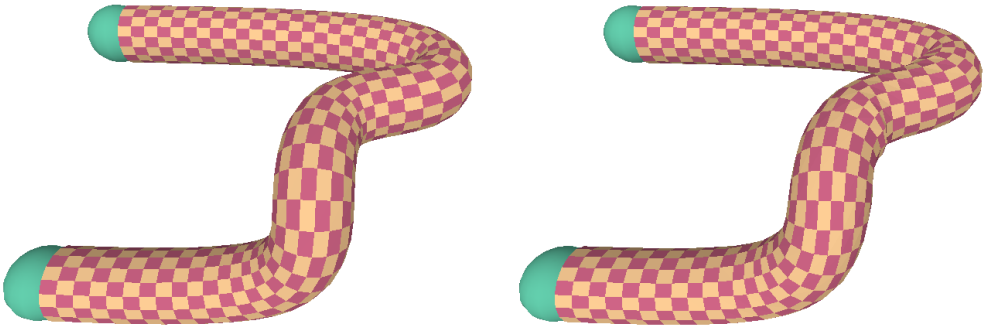


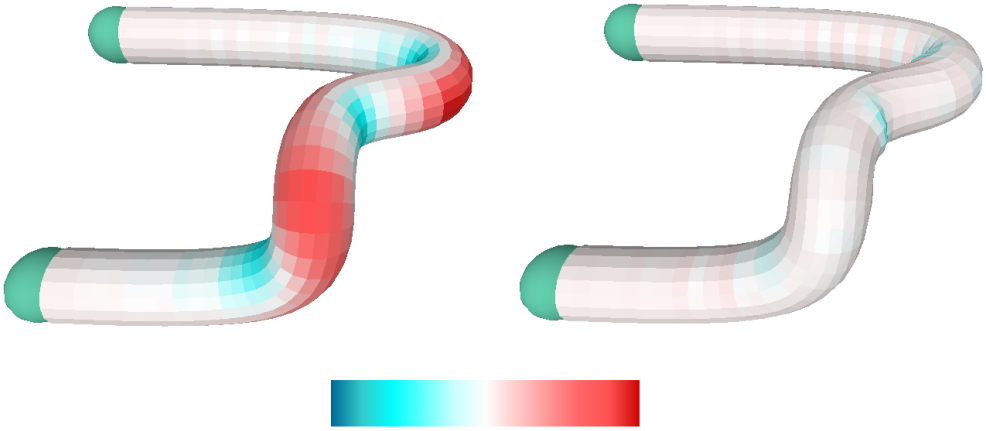
Figure 45: The crossectional plots of the deformation results computed by the normal, radial and binormal methods with the curvature increasing from left to right.



(a) The original tube



(b) The twisted tube with checker texture mapping



(c) The twisted tube with tone mapping

Figure 46: Using checker texture and tone mapping, we see a decreased local volume variation after applying radial offset distance correction (right).

8.4 Relationship between curvature and local volume variations

In this section we discuss the relationship between the local curvature and the local volume variation. Usually, the effect of local volume variations caused by curvature changes are not intuitive to see from the appearance of the object's surface after deformation. Here we provide a example showing how the curvature contributes to the variance of local volume distribution.

Consider the uniformly spaced tube shown in Figure 46(a). It has 64 edges of equal length on its axis and its cross-section is a equilateral 16-gon. So there are 1024 quads on the tube surface. Each quad and the corresponding edge on the axis form a wedge. The volume of every wedge of the tube is initially the same.

Figure 46(b) shows two versions after bending and twisting the tube while the length of each edge on the axis remain the same. The left one is without offset distance correction. The right one is the result computed by radial offset distance correction. Though some of the quads appear larger on the left, it is not easy to see the difference between the two results. In Figure 46(c), we map the local volume change to a color in the blue-red color ramp, and paint the quad with the color. Consequently, the difference between the two results becomes clear: the volume of each wedge varies much in the version without correcting the offset distance. The chart in Figure 47 shows the volumes of the one-ring wedges (in the uncorrected version) and the curvature at each vertex of the tube's axis: The local volume variation is strongly correlated with the local curvature.

8.5 Realtime performance

We now briefly provide an overview of the realtime performance in our locally volume preserving spine-driven deformation framework implemented in Java SE 7. Each demo or experiment is wrapped by a JVM process that runs on a 2.7GHz Intel Core i7. For

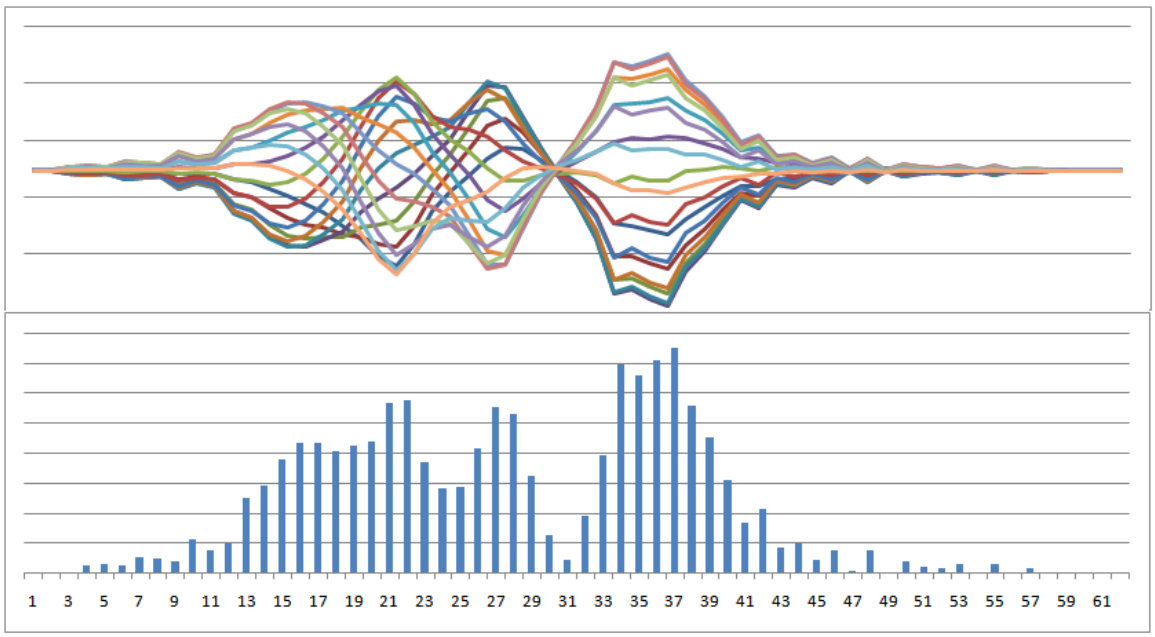


Figure 47: Compare the local curvatures (bottom) with the local volumes of the wedges of the twisted tube in Figure 46

a spine curve with 100 sampling points and a 3D mesh model with 6082 vertices, the registration takes 36 milliseconds. When the user manipulates the spine curve, the deformation (by the radial solution) takes 10 milliseconds on average for all vertices of the object. For a spine surface with 40×40 sampling points and a object model with 6082 vertices, the registration takes 432 milliseconds. When the user changes the spine surface, the transformation takes 8 milliseconds for all vertices of the objects.

CHAPTER IX

CONCLUSION

This dissertation makes several unique contributions. First, we study the problem of spine-driven deformation and the challenge of applying the constraint of local volume preservation everywhere during the deformation. We develop a framework for deforming an object driven by a spine (Chapter 1) to let the designer bend or stretch an object with a lower dimensional proxy. In this framework, the object is registered to the lower dimensional proxy, or the spine. As the spine changes, points of the deformed object are reconstructed as offsets from the spine. However, due to that the spine is curved, stretched or compressed in different places, there are global and local volume losses or gains in the reconstructed image or object. The key to preserve the local area or volume in the reconstructed image or object is to adjust the offset distance from the spine.

One of the most important ideas proposed by this dissertation is to use closed-form solutions of the offset distances based on the local curvature and stretch of the spine. We have considered different representation of the spine. To deform a planar shape or an image, the spine is a planar curve modeled by an arc or a parametric curve (Chapter 3). The offset distance is the solution of a quadratic equation with coefficients specified by the local curvatures and the original offset distance. The offset direction is along the normal of the planar curve. To deform a 3D object modeled by a triangle mesh or quad mesh, the spine curve is either planar or non-planar (Chapter 4). To reconstruct a 3D object from the spine curve, there is an additional degree of freedom in the crossectional plane: the offset direction can be along the binormal, normal of,

or radially outward from, the spine curve. The corresponding offset distances are solutions to linear, quadratic and cubic equations. In addition to using a curve, the spine can also be a surface (Chapter 6). In this case, the offset distance is the solution of a cubic equation with coefficients specified by the local mean and Gaussian curvatures. The offset direction is along the surface normal. To sum, we have derived a suite of closed-form solutions, by exploiting the requirement of constant unit Jacobian of the locally volume-preserving transformation.

Second, to support not just planar bending or stretching but also extend to more complex deformations, we devise a technique that decompose the deformation into three steps: unbending, transfer and bending. This decomposition is necessary to allow registration and reconstruction with normal-propagated frames while deforming the object registered to the spine curve with local volume preservation. Using the normal propagated frame rather than the Frenet frame reduces the twist along a 3D curve and prevents sudden changes of the normal direction. It approximated the twist-minimized frame and is widely used for computing the frame along a curve. In the unbending step, we assume that the spine curve or surface is locally straightened or flattened, and compute a intermediate offset solution. The transfer step performs a change of basis and scaling with respect to the local stretch parameter. The final solution is computed by the bending step where the spine curve or surface is deformed into its target position. Overall, the decomposition makes it easier to combine modules of implementations towards more complex defromations where the spine can undergoes non-planar bending and stretching.

Third, we also study the accuracy (Chapter 7) and issues related to physical realism (Chapter 8) of our solutions. It is very important to compute the precise projection on the spine as the anchor point for the purpose of accurate registration and reconstruction in spine-driven deformation. The choice of projection eventually

affects the precision in the deformation result, as well as the visual quality of the reconstructed 3D object. Therefore, instead of selecting the closest vertex of the spine, we propose to compute the closest point on the polyline or mesh approximation of the spine. This closest point may be interpolated from nearby vertices. We develop interpolation methods to evaluate the curvature and the normal at this closest point for solving the offset distance. At the end of this dissertation, we discuss issues related to physical realism, though a comprehensive discussion on the subject is outside the scope of this work. Specifically, we look at how to combine the two basis bending modes by revisiting the derivations. To facilitate comparison with real-world deformations with local volume preservation, we provide a chart showing the change of crossection points in different bending modes with increased curvatures, and an additional example revealing the relationship between the local curvature and the local volume variation.

The closed-form solutions could apply easily to many other problem spaces. Here are three general examples.

- *Tool path planning and generation* ([63, 36, 16], etc.) General milling tools have sufficient degrees of freedom which allow them to follow arbitrary planar paths. One of the challenges is to define a tool path that lead to constant material removal rate in milling for a target shape. Since we want to keep the translational speed of the milling tool as a constant, the removed area per unit length should be constant in order to achieve stable power consumption. Our quadratic equation in 2D provides the solution to the offset distance that defines the tool path with removed area per unit length equal to a constant.

- *Curvature-based volume correction or compensation* ([63, 37, 56, 17], etc.) The deformation of a 2D or 3D shape may be the result of subdivision or smoothing operations, which often cause an area or volume loss. We want to obtain an offset shape that is similar to the original shape, but with a different area or

volume enclosed. It has been proved that the solution, which minimizes the Hausdorff distance between the two shapes, is the constant distance offset from one to the other, assuming that the two shapes are ball compatible [15]. The quadratic or cubic equation presented in this dissertation provides the exact solution to the offset distance (in 2D or 3D) to recover the original area or volume.

- *Spine-driven deformation and animation* ([64, 46, 11, 7, 27, 55], etc.) The spine is a lower dimensional proxy used to control the deformation of a shape which is roughly aligned along the spine. Note that the spine does not need to be the medial axis or surface of the shape. Given a deformed version of the spine, we want to compute the deformation of the shape that preserves the original area or volume locally. The approaches advocated in this dissertation defines the deformed shape as the normal offset from the spine with the offset distances computed by different solutions presented.

Implementation techniques introduced in this dissertation focus on the second and third examples. Specifically, the unbending-transfer-bending technique can be applied to all types of spine-driven deformations (Chapter 4 and Chapter 5). A solver for the cubic equation in the the deformation driven by a spine surface contributes to the implementation for curvature-based volume compensation as well, however, coefficients of the equation represent the *global, not local,* mean and Gaussian curvature measures on the base surface in the constant distance offsetting (Chapter 6).

There are several interesting research directions that we intend to pursue. First, our current work focuses on simple object representations in terms of the limited mesh data available. So, we have subdivided a mesh at different levels, or created multiples of one model at different locations, for more experimental results with large number of points in space. It would be interesting to parallel the deformation computation

for an object represented by a large set of points. Second, in our problem formulation the control proxy does not have any bifurcation. Our local-volume preserving deformation algorithms are valid for a point with unique closest projection on the spine. This assumption might not hold in the presence of a spine bifurcation. Hence, it is worthwhile to explore a new mathematical framework or algorithms that can generalize to different spine requirements including bifurcations. It would also be an interesting topic to study the opportunity in developing professional animation tools that integrate these offsetting solutions we advocate.

REFERENCES

[1] Amdahl, G. M., "Validity of the single processor approach to achieving large scale computing capabilities," in *Proceedings of the April 18-20, 1967, spring joint computer conference*, pp. 483–485, ACM, 1967.

[2] Angelidis, A., Cani, M.-P., Wyvill, G., and King, S., "Swirling-sweepers: Constant-volume modeling," *Graphical Models*, vol. 68, no. 4, pp. 324–332, 2006.

[3] Angelidis, A. and Singh, K., "Kinodynamic skinning using volume-preserving deformations," in *Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pp. 129–140, Eurographics Association, 2007.

[4] Antiga, L., Ene-Iordache, B., and Remuzzi, A., "Computational geometry for patient-specific reconstruction and meshing of blood vessels from mr and ct angiography," *Medical Imaging, IEEE Transactions on*, vol. 22, no. 5, pp. 674–684, 2003.

[5] Aubert, F. and Bechmann, D., "Volume-preserving space deformation," *Computers & Graphics*, vol. 21, no. 5, pp. 625–639, 1997.

[6] Baran, I., Lehtinen, J., and Popović, J., "Sketching clothoid splines using shortest paths," in *Computer Graphics Forum*, vol. 29, pp. 655–664, Wiley Online Library, 2010.

[7] Barr, A. H., "Global and local deformations of solid primitives," in *ACM Siggraph Computer Graphics*, vol. 18, pp. 21–30, ACM, 1984.

[8] Bathe, K.-J., *Finite element procedures*, vol. 2. Prentice hall Englewood Cliffs, 1996.

[9] Bauer, U. and Polthier, K., "Parametric reconstruction of bent tube surfaces," in *Proceedings of International Conference on Cyberworlds 2007* (Wolter, F.-E. and Sourin, A., eds.), pp. 465–474, IEEE, 2007.

[10] Botsch, M. and Kobbelt, L., "Multiresolution surface representation based on displacement volumes," in *Computer Graphics Forum*, vol. 22, pp. 483–491, Wiley Online Library, 2003.

[11] Botsch, M. and Sorkine, O., "On linear variational surface deformation methods," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 14, no. 1, pp. 213–230, 2008.

[12] Burtnyk, N. and Wein, M., "Interactive skeleton techniques for enhancing motion dynamics in key frame animation," *Communications of the ACM*, vol. 19, no. 10, pp. 564–569, 1976.

[13] Capell, S., Burkhart, M., Curless, B., Duchamp, T., and Popović, Z., "Physically based rigging for deformable characters," in *Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pp. 301–310, ACM, 2005.

[14] Capell, S., Green, S., Curless, B., Duchamp, T., and Popović, Z., "Interactive skeleton-driven dynamic deformations," in *ACM Transactions on Graphics (TOG)*, vol. 21, pp. 586–593, ACM, 2002.

[15] Chazal, F., Lieutier, A., Rossignac, J., and Whited, B., "Ball-map: Homeomorphism between compatible surfaces," *International Journal of Computational Geometry & Applications*, vol. 20, no. 03, pp. 285–306, 2010.

[16] Chirikjian, G. S., "Closed-form primitives for generating locally volume preserving deformations," *Journal of Mechanical Design*, vol. 117, p. 347, 1995.

[17] Desbrun, M., Meyer, M., Schröder, P., and Barr, A. H., "Implicit fairing of irregular meshes using diffusion and curvature flow," in *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pp. 317–324, ACM Press/Addison-Wesley Publishing Co., 1999.

[18] Do Carmo, M. P., *Riemannian geometry*. Springer, 1992.

[19] Farouki, R. T. and Han, C. Y., "Rational approximation schemes for rotation-minimizing frames on pythagorean-hodograph curves," *Computer Aided Geometric Design*, vol. 20, no. 7, pp. 435–454, 2003.

[20] Flynn, M., "Very high-speed computing systems," *Proceedings of the IEEE*, vol. 54, no. 12, pp. 1901–1909, 1966.

[21] Foley, J. D., *Computer graphics: Principles and practice, in C*, vol. 12110. Addison-Wesley Professional, 1996.

[22] Foote, R. L., "The volume swept out by a moving planar region," *Mathematics Magazine*, pp. 289–297, 2006.

[23] Hahmann, S., Sauvage, B., and Bonneau, G.-P., "Area preserving deformation of multiresolution curves," *Computer aided geometric design*, vol. 22, no. 4, pp. 349–367, 2005.

[24] Hanson, A. J. and Ma, H., "Parallel transport approach to curve framing," *Indiana University, Techreports-TR425*, vol. 11, pp. 3–7, 1995.

[25] Hirota, G., Maheshwari, R., and Lin, M. C., "Fast volume-preserving free-form deformation using multi-level optimization," *Computer-Aided Design*, vol. 32, no. 8, pp. 499–512, 2000.

[26] Hsu, S. C. and Lee, I. H., "Drawing and animation using skeletal strokes," in *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pp. 109–118, ACM, 1994.

[27] Hsu, S. C., Lee, I. H., and Wiseman, N. E., "Skeletal strokes," in *Proceedings of the 6th annual ACM symposium on User interface software and technology*, pp. 197–206, ACM, 1993.

[28] Irving, G., Schroeder, C., and Fedkiw, R., "Volume conserving finite element simulations of deformable models," *ACM Transactions on Graphics (TOG)*, vol. 26, no. 3, p. 13, 2007.

[29] Kälberer, F., Nieser, M., and Polthier, K., "Stripe parameterization of tubular surfaces," in *Topological Methods in Data Analysis and Visualization*, pp. 13–26, Springer, 2011.

[30] Kass, M. and Anderson, J., "Animating oscillatory motion with overlap: wiggly splines," in *ACM Transactions on Graphics (TOG)*, vol. 27, p. 28, ACM, 2008.

[31] Kim, B., Liu, Y., Llamas, I., Jiao, X., and Rossignac, J., "Simulation of bubbles in foam with the volume control method," in *ACM Transactions on Graphics (TOG)*, vol. 26, p. 98, ACM, 2007.

[32] Kim, M.-S., Park, E.-J., and Lim, S.-B., "Approximation of variable-radius offset curves and its application to bezier brush-stroke design," *Computer-Aided Design*, vol. 25, no. 11, pp. 684–698, 1993.

[33] Krogh, F. T., "Efficient algorithms for polynomial interpolation and numerical differentiation," *Mathematics of Computation*, vol. 24, no. 109, pp. 185–190, 1970.

[34] Lewis, J. P., Cordner, M., and Fong, N., "Pose space deformation: a unified approach to shape interpolation and skeleton-driven deformation," in *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pp. 165–172, ACM Press/Addison-Wesley Publishing Co., 2000.

[35] Llamas, I., Powell, A., Rossignac, J., and Shaw, C. D., "Bender: a virtual ribbon for deforming 3d shapes in biomedical and styling applications," in *Proceedings of the 2005 ACM symposium on Solid and physical modeling*, pp. 89–99, ACM, 2005.

[36] Moon, H. P., "Equivolumetric offsets for 2d machining with constant material removal rate," *Computer Aided Geometric Design*, vol. 25, no. 6, pp. 397–410, 2008.

[37] Moon, H. P., "Equivolumetric offset surfaces," *Computer Aided Geometric Design*, vol. 26, no. 1, pp. 17–36, 2009.

[38] Müller, M., Dorsey, J., McMillan, L., Jagnow, R., and Cutler, B., "Stable real-time deformations," in *Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pp. 49–54, ACM, 2002.

[39] O'Donnell, T., Boult, T. E., Fang, X.-S., and Gupta, A., "The extruded generalized cylinder: A deformable model for object recovery," in *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR'94., 1994 IEEE Computer Society Conference on*, pp. 174–181, IEEE, 1994.

[40] Pieper, S. D., *More than skin deep: Physical modeling of facial tissue*. PhD thesis, Massachusetts Institute of Technology, 1989.

[41] Raveendran, K., Thuerey, N., Wojtan, C., and Turk, G., "Controlling liquids using meshes," in *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pp. 255–264, Eurographics Association, 2012.

[42] Raveendran, K., Wojtan, C., Thuerey, N., and Turk, G., "Blending liquids," in *Proceedings of the annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., 2014.

[43] Reddy, J. N., *Theory and analysis of elastic plates and shells*. CRC press, 2007.

[44] Reuter, M., Biasotti, S., Giorgi, D., Patanè, G., and Spagnuolo, M., "Discrete laplace–beltrami operators for shape analysis and segmentation," *Computers & Graphics*, vol. 33, no. 3, pp. 381–390, 2009.

[45] Rohmer, D., Hahmann, S., and Cani, M.-P., "Local volume preservation for skinned characters," in *Computer Graphics Forum*, vol. 27, pp. 1919–1927, Wiley Online Library, 2008.

[46] Rohmer, D., Hahmann, S., and Cani, M.-P., "Exact volume preserving skinning with shape control," in *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pp. 83–92, ACM, 2009.

[47] Rossignac, J. and Schaefer, S., "J-splines," *Computer-Aided Design*, vol. 40, no. 10, pp. 1024–1032, 2008.

[48] Rossignac, J. R. and Requicha, A. A., "Piecewise-circular curves for geometric modeling," *IBM Journal of Research and Development*, vol. 31, no. 3, pp. 296–313, 1987.

[49] Salomon, D., *Computer graphics and geometric modeling*. Springer, 1999.

[50] Sauvage, B., Hahmann, S., and Bonneau, G.-P., "Length preserving multiresolution editing of curves," *Computing*, vol. 72, no. 1-2, pp. 161–170, 2004.

[51] Schaefer, S., McPhail, T., and Warren, J., "Image deformation using moving least squares," in *ACM Transactions on Graphics (TOG)*, vol. 25, pp. 533–540, ACM, 2006.

[52] Sederberg, T. W. and Parry, S. R., "Free-form deformation of solid geometric models," in *ACM Siggraph Computer Graphics*, vol. 20, pp. 151–160, ACM, 1986.

[53] Stam, J., "Stable fluids," in *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pp. 121–128, ACM Press/Addison-Wesley Publishing Co., 1999.

[54] Steiner, J., "über parallele flächen," *Monatsberichte der Akademie der Wissenschaft zu Berlin (Monthly Report of the Academy of Sciences, Berlin)*, pp. 114–118, 1840.

[55] Storti, D. W., Turkiyyah, G. M., Ganter, M. A., Lim, C. T., and Stal, D. M., "Skeleton-based modeling operations on solids," in *Proceedings of the fourth ACM symposium on Solid modeling and applications*, pp. 141–154, ACM, 1997.

[56] Taubin, G., "A signal processing approach to fair surface design," in *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pp. 351–358, ACM, 1995.

[57] von Funck, W., Theisel, H., and Seidel, H.-P., "Vector field based shape deformations," *ACM Transactions on Graphics (TOG)*, vol. 25, no. 3, pp. 1118–1125, 2006.

[58] Wang, W., Jüttler, B., Zheng, D., and Liu, Y., "Computation of rotation minimizing frames," *ACM Transactions on Graphics (TOG)*, vol. 27, no. 1, p. 2, 2008.

[59] Weber, O., Sorkine, O., Lipman, Y., and Gotsman, C., "Context-aware skeletal shape deformation," in *Computer Graphics Forum*, vol. 26, pp. 265–274, Wiley Online Library, 2007.

[60] Yan, H.-B., Hu, S.-M., Martin, R. R., and Yang, Y.-L., "Shape deformation using a skeleton to drive simplex transformations," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 14, no. 3, pp. 693–706, 2008.

[61] Yoshizawa, S., Belyaev, A. G., and Seidel, H.-P., "Free-form skeleton-driven mesh deformations," in *Proceedings of the eighth ACM symposium on Solid modeling and applications*, pp. 247–253, ACM, 2003.

[62] Zhuo, W., Prabhat, P., Paciorek, C., Kaufman, C., and Bethel, W., "Parallel kriging analysis for large spatial datasets," in *Data Mining Workshops (ICDMW), 2011 IEEE 11th International Conference on Climate Knowledge Discovery*, pp. 38–44, IEEE, 2011.

[63] Zhuo, W. and Rossignac, J., "Curvature-based offset distance: Implementations and applications," *Computers & Graphics*, vol. 36, no. 5, pp. 445–454, 2012.

[64] Zhuo, W. and Rossignac, J., "Fleshing: Spine-driven bending with local volume preservation," in *Computer Graphics Forum*, vol. 32, pp. 295–304, Wiley Online Library, 2013.

VITA

Wei Zhuo was born and raised in Changsha, a big city in south-central China. She received a Bachelor of Engineering degree from Hong Kong University of Science and Technology in 2009. Subsequently, she moved to Atlanta, United States to pursue a Ph.D. in Computer Science at the College of Computing at Georgia Institute of Technology. As a member of the geometry research group and GVU Center at the College of Computing, she conducted research on various aspects of shape modeling, deformation and visualization under the guidance of Prof. Jarek Rossignac. Her research has resulted in publications that have appeared in international conferences and journals on computer graphics and geometric modeling. She has also been a collaborator with the IBM T.J. Waston Research Center and Lawrence Berkeley National Lab. She received Raytheon Research Award in 2013. Her work at IBM has been filed for patent, dealing with geometric design in visual analysis.