COMPUTING CURVATURE AND CURVATURE NORMALS ON SMOOTH LOGICALLY CARTESIAN SURFACE MESHES

by

John Thomas Hutchins

A thesis

submitted in partial fulfillment

of the requirements for the degree of

Master of Science in Mathematics

Boise State University

December 2013

BOISE STATE UNIVERSITY GRADUATE COLLEGE

DEFENSE COMMITTEE AND FINAL READING APPROVALS

of the thesis submitted by

John Thomas Hutchins

Thesis Title: Computing Curvature and Curvature Normals on Smooth Logically Cartesian Surface Meshes

Date of Final Oral Examination: 02 August 2013

The following individuals read and discussed the thesis submitted by student John Thomas Hutchins, and they evaluated his presentation and response to questions during the final oral examination. They found that the student passed the final oral examination.

Donna Calhoun, Ph.D.	Chair, Supervisory Committee
Jodi Mead, Ph.D.	Member, Supervisory Committee
Grady Wright, Ph.D.	Member, Supervisory Committee

The final reading approval of the thesis was granted by Donna Calhoun, Ph.D., Chair of the Supervisory Committee. The thesis was approved for the Graduate College by John R. Pelton, Ph.D., Dean of the Graduate College.

ACKNOWLEDGMENTS

The author wishes to express gratitude to Megan Hutchins for her valuable assistance and support.

ABSTRACT

This thesis describes a new approach to computing mean curvature and mean curvature normals on smooth logically Cartesian surface meshes. We begin by deriving a finite-volume formula for one-dimensional curves embedded in two- or three-dimensional space. We show the exact results on curves for specific cases as well as second-order convergence in numerical experiments. We extend this finite-volume formula to surfaces embedded in three-dimensional space. Exact results are again derived for special cases and second-order convergence is shown numerically for more general cases. We show that our formula for computing curvature is an improvement over using the "cotan" formula on a triangulated quadrilateral mesh and is conceptually much simpler than the formula proposed by Liu *et al.* ("A discrete scheme of Laplace-Beltrami operator and its convergence over quadrilateral meshes," Computers and Mathematics with Applications, 2008), and is equivalent in performance.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF SYMBOLS

κ	Mean curvature, scalar
$\mathbf{H}$	Mean curvature normal, vector in $\mathbb{R}^3$
$\mathbf{n}$	Edge normal, vector in $\mathbb{R}^3$
$\mathbf{n_s}$	Surface normals, vector in $\mathbb{R}^3$
$\mathbf{T}(\xi, \eta)$	Mapping from $\mathbb{R}^2 \rightarrow \mathbb{R}^3$
$\mathbf{X_d}$	Dual point in mesh, in $\mathbb{R}^3$
$\mathbf{X_p}$	Primal point in mesh, in $\mathbb{R}^3$
τ	Torsion, scalar
$\mathbf{b}$	Binormal, vector in $\mathbb{R}^3$
C	One-dimensional mesh cell
S	Two-dimensional surface patch defined by coordinate directions

CHAPTER 1

INTRODUCTION

The calculation of mean curvature and mean curvature normals on surfaces is an important task in discrete differential geometry [5]. It is used in signal processing, surface smoothing, image processing, and although we will not consider the solving of partial differential equations in this thesis, curvature plays an important role in the dynamics between fluid interfaces [9].

We will extend the work of D. A. Calhoun and C. Helzel [3] to the computing of mean curvature and mean curvature normals on quadrilateral surface meshes. In that paper, a finite-volume method for solving parabolic equations was introduced, which derived a nine-point stencil for the Laplace-Beltrami operator. We apply the ideas behind the stencil along with results from discrete differential geometry to obtain a new finite volume scheme for calculating curvature on quadrilateral surface meshes.

After explaining the background and previous work, we will first derive a finite-volume scheme for calculating curvature and normals on one-dimensional planar curves. We then extend this to calculating binormals and torsion for non-planar curves, in addition to normals and curvature. Then, we derive a finite-volume form of calculating mean curvature and the mean curvature normals on surfaces. We then give the results for the surface scheme and compare it to the well-known "cotan" formula for calculating curvature on triangular meshes and a quadrilateral scheme

proposed by D. Liu *et al.* [7].

1.1 Background

1.1.1 Overview of Curves and Surfaces

Curvature is a measure of how much a curve or surface deviates from being straight or flat with respect to the underlying space. On a surface, every curve that can be traced through a point on the surface will have its own curvature at that point. The maximum and minimum values of these curvatures are known as the principle curvatures of the surface. An equivalent way of considering principle curvatures is that they are the eigenvalues of the metric tensor [5]. On a surface, mean curvature is the average of the two principle curvatures and it locally describes the embedded surface in the underlying space, which for us will always be Euclidean space. Gaussian curvature, not examined in this paper, is the product of the principle curvatures.

For one-dimensional curves we can use the Frenet-Serret formulas, or Frenet frame, to relate the tangent vector, the surface normal, $\mathbf{n_s}$, the binormal $\mathbf{b}$, along with the scalar quantities curvature, κ, and torsion, τ. In presenting the formulas here, we will use the arc-length parameterization, although later we will relax this and allow for more general parameterizations. Given a mapping $\mathbf{r}(s) = (X(s), Y(s), Z(s))$, mapping $\mathbb{R} \to \mathbb{R}^3$ where s is assumed to be arc length, the tangent vector to the curve is computed as

$$\mathbf{t} = \frac{d\mathbf{r}(s)}{ds}, \tag{1.1}$$

where d/ds is the derivative with respect to arc length. The Frenet-Serret formulas

relating $\mathbf{t}, \mathbf{n_s}$, and $\mathbf{b}$ are given by

$$\begin{aligned} \frac{d\mathbf{t}}{ds} &= \kappa \mathbf{n}_s \\ \frac{d\mathbf{n}_s}{ds} &= -\kappa \mathbf{t} + \tau \mathbf{b} \\ \frac{d\mathbf{b}}{ds} &= -\tau \mathbf{n}_s. \end{aligned} \tag{1.2}$$

One direct consequence of the first equation is that $\kappa = \|\mathbf{r}''(s)\|$, since $\mathbf{n}_s$ has unit length. For surfaces, an analogous frame is the Darboux frame.

For surfaces, the Laplace-Beltrami operator (LBO) leads directly to the mean curvature normal. The LBO is a generalization of the Laplace operator to surfaces and can be expressed as the divergence of the gradient of a function defined on the surface. We let ∇^2 be the LBO, $\mathbf{H}$ be the mean curvature normal, and $\mathbf{n_s}$ the surface normal. Given a surface mapping $\mathbf{T}(\xi, \eta)$, $\mathbb{R}^2 \rightarrow \mathbb{R}^3$, the surface analog of equation (1.2) is given by

$$\begin{aligned} \frac{1}{2}\nabla^2 \mathbf{T}(\xi, \eta) = \mathbf{H} &= \kappa \mathbf{n}_s \\ \kappa &\equiv \|\mathbf{H}\|, \end{aligned} \tag{1.3}$$

where κ is the mean curvature on the surface. Below, we define what the Laplacian of the vector $\mathbf{T}$ means.

1.1.2 Previous Work

Because of the importance of computing curvature and normals in various application domains and in solving partial differential equations involving the Laplacian, as equation (1.3) shows is the same task as computing curvature, there have been several approaches proposed for computing the LBO. Many of these schemes, especially those

from computer graphics, assume a triangulated mesh and as shown in G. Xu [9] many of these schemes do not converge generally.

A result from Xu *et al.* [10] is that one cannot build a discrete scheme for mean curvature or the LBO that converges over general triangulated surfaces, though second-order accuracy can be obtained under specific conditions. While the result of Xu *et al.* does extend to quadrilateral meshes, a scheme built of quadrilateral meshes will converge more generally over the same points than under a triangularization scheme, demonstrated numerically both in this paper and in D. Liu *et al.* [7]. A heuristic argument for this is that a quadrilateral does not have to exist on a plane while a triangle always exists on a plane, allowing a quadrilateral to more closely capture the surface curvature than does a triangulated mesh produced from the quadrilateral mesh.

Though triangulated surfaces are the most common type of surface meshes used in surface processing and other applications, quadrilateral meshes are also frequently used. For solving partial differential equations, logically Cartesian meshes are considered more desirable than triangular meshes. Recent research efforts are directed towards designing logically Cartesian surface meshes from more general surface triangularizations [6]. The LBO is often used when solving partial differential equations on surfaces, but in many cases the particular discrete operator is not formally consistent. As a result, use of that operator for computing curvature may not lead to an approximation that converges as the mesh is refined. There are multiple options for creating triangularizations of quadrilateral meshes, each of which produce different results when using the cotan formula, as shown in [7]. Due to this, new methods for approximating the LBO on quadrilateral meshes are needed. In [7], a form of the LBO is presented that converges at a quadratic rate on mesh refinement. We will

compare our results to those of both the cotan formula and the formula from [7].

CHAPTER 2

CALCULATING CURVATURE ON ONE-DIMENSIONAL SPATIAL CURVES

2.1 Finite Volume Schemes for Computing Curvature on a One-Dimensional Planar Curve

The basis for our work is the use of finite-volume discretizations for computing the surface normals and curvature. In this section, we describe these ideas using a one-dimensional planar curve. In Chapter 3, we extend the one-dimensional case to surface meshes embedded in three-dimensional space.

We begin by assuming that we have a one-dimensional smooth planar curve described parametrically by the mapping $\mathbf{T} : \mathbb{R} \rightarrow \mathbb{R}^2$

$$\mathbf{T}(\xi) = (X(\xi), Y(\xi)). \tag{2.1}$$

The basis vector, $\mathbf{t}_{(1)}$, is given by

$$\mathbf{t}_{(1)} = (X_\xi, Y_\xi), \qquad \mathbf{t}_{(1)} \in \mathbb{R}^2, \tag{2.2}$$

and the metric tensor, $\mathbf{a}$, is

$$\mathbf{a} = [\mathbf{t}_{(1)} \cdot \mathbf{t}_{(1)}] \equiv [a_{11}], \qquad \mathbf{a} \in \mathbb{R}, \tag{2.3}$$

leading to an inverse metric tensor of

$$\mathbf{a}^{-1} = \left[\frac{1}{a_{11}}\right] \equiv [a^{11}], \qquad \mathbf{a}^{-1} \in \mathbb{R}. \tag{2.4}$$

We define a conjugate vector field

$$\mathbf{t}^{(1)} = a^{11}\mathbf{t}_{(1)}, \qquad \mathbf{t}^{(1)} \in \mathbb{R}^2 \tag{2.5}$$

from which we get that

$$\mathbf{t}_{(1)} \cdot \mathbf{t}^{(1)} = 1 \tag{2.6}$$

and

$$\mathbf{t}^{(1)} \cdot \mathbf{t}^{(1)} = a^{11}. \tag{2.7}$$

Given any vector $\mathbf{w} \in \mathbb{R}^2$, we can write

$$\mathbf{w} = w^1\mathbf{t}_{(1)} \tag{2.8}$$

or

$$\mathbf{w} = w_1\mathbf{t}^{(1)} \tag{2.9}$$

where w^1 is the contravariant component of $\mathbf{w}$ and w_1 is the covariant component. These are computed as

$$w^1 = \mathbf{w} \cdot \mathbf{t}^{(1)} = (w^1\mathbf{t}_{(1)}) \cdot \mathbf{t}^{(1)} \tag{2.10}$$

and

$$w_1 = \mathbf{w} \cdot \mathbf{t}_{(1)} = (w_1 \mathbf{t}^{(1)}) \cdot \mathbf{t}_{(1)}. \tag{2.11}$$

The gradient of a scalar function $\phi(\xi)$ defined on the curve can be expressed in terms of its covariant component,

$$\widetilde{\nabla}\phi = \phi_\xi \mathbf{t}^{(1)}. \tag{2.12}$$

We can use these ideas in a finite-volume setting to obtain expressions for normal vectors and the surface curvature.

For the planar curve, we will actually use the one-dimensional analogue of equation (1.3) rather than that suggested by the Frenet frame. One reason for this is that we can then easily extend these ideas to two dimensions without disruptive changes to notation. The one-dimensional analogue to equation (1.3) is given by

$$\nabla^2 \mathbf{T} = \nabla \cdot \widetilde{\nabla}\mathbf{T} = \mathbf{H}, \tag{2.13}$$

where the operator $\nabla \cdot ()$ is the surface divergence and $\mathbf{H}$ is a vector normal to the planar curve with its magnitude equal to the curvature. The gradient of $\mathbf{T}$ is understood to be

$$\widetilde{\nabla}\mathbf{T} = \begin{pmatrix} \widetilde{\nabla}X \\ \widetilde{\nabla}Y \end{pmatrix}. \tag{2.14}$$

where $\widetilde{\nabla}X$ and $\widetilde{\nabla}Y$ are defined as in equation (2.12).

We would like to use this formalism to approximate the surface curvature and mean curvature normal using a finite-volume approximation. We begin by using the divergence theorem to approximate the average value of the Laplacian of $\mathbf{T}$ over a cell, C. From this we get

$$\int_C \nabla \cdot \widetilde{\nabla}\mathbf{T} dA = \widetilde{\nabla}\mathbf{T} \cdot \mathbf{n}\Big|_{\text{left}}^{\text{right}} = \widetilde{\nabla}\mathbf{T} \cdot \mathbf{n}_r - \widetilde{\nabla}\mathbf{T} \cdot \mathbf{n}_l, \tag{2.15}$$

where $\mathbf{n}_r$ and $\mathbf{n}_l$ are the "edge normals", or tangents to the curve at mesh cell edges, calculated at the left and right edges respectively. This leads to $\widetilde{\nabla}\mathbf{T} \cdot \mathbf{n}$ in equation (2.15), which we understand as

$$\widetilde{\nabla}\mathbf{T} \cdot \mathbf{n} = \begin{pmatrix} \widetilde{\nabla}X \\ \widetilde{\nabla}Y \end{pmatrix} \cdot \mathbf{n} = \begin{pmatrix} \widetilde{\nabla}X \cdot \mathbf{n} \\ \widetilde{\nabla}Y \cdot \mathbf{n} \end{pmatrix}. \tag{2.16}$$

We can now evaluate the integral in equation (2.15) by writing the "edge normal," $\mathbf{n}$ as

$$\mathbf{n} = \frac{\mathbf{t}^{(1)}}{\|\mathbf{t}^{(1)}\|} = \frac{\mathbf{t}_{(1)}}{\|\mathbf{t}_{(1)}\|}. \tag{2.17}$$

This, when combined with equation (2.16), leads to

$$\widetilde{\nabla}X \cdot \mathbf{n} = (X_\xi \mathbf{t}^{(1)}) \cdot \frac{\mathbf{t}_{(1)}}{\|\mathbf{t}_{(1)}\|} = \frac{1}{\|\mathbf{t}_{(1)}\|} X_\xi \tag{2.18}$$

and

$$\widetilde{\nabla}Y \cdot \mathbf{n} = (Y_\xi \mathbf{t}^{(1)}) \cdot \frac{\mathbf{t}_{(1)}}{\|\mathbf{t}_{(1)}\|} = \frac{1}{\|\mathbf{t}_{(1)}\|} Y_\xi \tag{2.19}$$

From equation (2.17), we now get the result that

$$\widetilde{\nabla}\mathbf{T} \cdot \mathbf{n} = \frac{1}{\|\mathbf{t}_{(1)}\|} \begin{pmatrix} X_\xi \\ Y_\xi \end{pmatrix} \equiv \mathbf{n}. \tag{2.20}$$

To compute the curvature normal vector, we will be interested in the average value of $\nabla^2\mathbf{T}$ over a finite volume. Letting L(C) be the length of a one-dimensional "cell,"

C, we can write

$$\begin{aligned}\frac{1}{L(C)}\int_C \nabla\cdot\widetilde{\nabla}\mathbf{T}d\ell &= \frac{1}{L(C)}\{\widetilde{\nabla}\mathbf{T}\cdot\mathbf{n}_r - \widetilde{\nabla}\mathbf{T}\cdot\mathbf{n}_l\}\\ &= \frac{1}{L(C)}\{\mathbf{n}_r - \mathbf{n}_l\}.\end{aligned} \tag{2.21}$$

The curvature normal vector can then be written as

$$\mathbf{H} = \frac{1}{L(C)}\{\mathbf{n}_r - \mathbf{n}_l\}. \tag{2.22}$$

From this, we have the surface normal and curvature given by

$$\mathbf{n}_s = \frac{\mathbf{H}}{\|\mathbf{H}\|} = \frac{\mathbf{H}}{\kappa}, \tag{2.23}$$

where

$$\kappa \equiv \|\mathbf{H}\|. \tag{2.24}$$

2.1.1 Computational Scheme

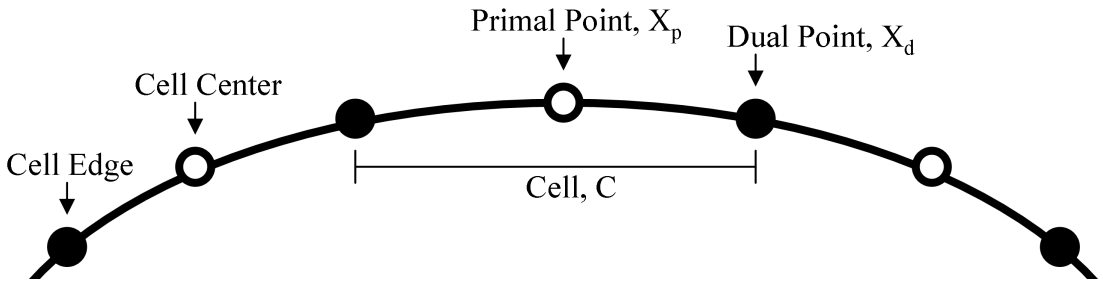

Figure 2.1: Curve showing primal and dual grid points with cell centers and edges.

We now refer to Figure 2.1 to show the computational grid consisting of dual points, $\mathbf{T}_d$, which are put down first and are considered to be the edges of cells, and primal

points, $\mathbf{T}_p$, which are at the midpoints of cells. This allows us to computationally approximate the basis vectors as in equation (2.2). Our approximation of these basis vectors begins by differencing the primal points along the grid,

$$\mathbf{t}_{(1),j} = \mathbf{T}_{p,j} - \mathbf{T}_{p,j-1}. \tag{2.25}$$

This directly gives us the ability to calculate at each primal grid point j,

$$a_{11,j} = \mathbf{t}_{(1),j} \cdot \mathbf{t}_{(1),j} \tag{2.26}$$

$$a_j^{11} = \frac{1}{a_{11,j}} \tag{2.27}$$

and

$$\mathbf{t}_j^{(1)} = a_j^{11} \cdot \mathbf{t}_{(1),j}. \tag{2.28}$$

Edge normals, which are tangents in one-dimension, are then just

$$\mathbf{n}_j = \frac{\mathbf{t}_{(1),j}}{\|\mathbf{t}_{(1),j}\|}. \tag{2.29}$$

From the edge normals, we can compute the surface normals, $\mathbf{n}_{s,j}$, and curvature κ_j at cell centers, which are again the primal points. We approximate the length of a cell by differencing the dual points X_d, which are at the edge of a cell

$$L(C_j) \approx \|\mathbf{T}_{d,j+1} - \mathbf{T}_{d,j}\|. \tag{2.30}$$

The surface normals are given as

$$\mathbf{n}_{s,j} = \frac{\mathbf{t}_{(1),j}}{\|\mathbf{t}_{(1),j}\|} = \frac{\mathbf{T}_{p,j} - \mathbf{T}_{p,j-1}}{\|\mathbf{T}_{p,j} - \mathbf{T}_{p,j-1}\|}. \tag{2.31}$$

Applying the approximation of the cell length and the normals into equation equation (2.22) gives the following formula for κ_j for the one-dimensional case

$$\kappa_j = \frac{1}{\|\mathbf{T}_{d,j+1} - \mathbf{T}_{d,j}\|} \left\| \frac{\mathbf{T}_{p,j+1} - \mathbf{T}_{p,j}}{\|\mathbf{T}_{p,j+1} - \mathbf{T}_{p,j}\|} - \frac{\mathbf{T}_{p,j} - \mathbf{T}_{p,j-1}}{\|\mathbf{T}_{p,j} - \mathbf{T}_{p,j-1}\|} \right\|. \tag{2.32}$$

This is the formula we will use to approximate curvature on one-dimensional curves.

2.1.2 Scheme on a Linear Equation

Having defined our scheme for calculating curvature and normals, it is important to investigate the accuracy of the scheme. The easiest case to consider is that of a linear equation, which has a curvature of zero. Given a linear mapping

$$\begin{aligned} \mathbf{T}(\xi) &= (X(\xi), Y(\xi)) \\ X(\xi) &= a\xi + b \\ Y(\xi) &= c\xi + d \end{aligned} \tag{2.33}$$

then, defining $h_j = \xi_{p,j+1} - \xi_{p,j}$, we get

$$\begin{aligned} \|\mathbf{T}_{p,j+1} - \mathbf{T}_{p,j}\| &= \sqrt{(a\xi_{p,j+1} + b - a\xi_{p,j} - b)^2 + (c\xi_{p,j+1} + d - c\xi_{p,j} - d)^2} \\ &= \sqrt{(ah_j)^2 + (ch_j)^2} \\ &= h_j\sqrt{a^2 + c^2}. \end{aligned} \tag{2.34}$$

Similarly,

$$\|\mathbf{T}_{p,j} - \mathbf{T}_{p,j-1}\| = h_{j-1}\sqrt{a^2 + c^2}, \tag{2.35}$$

and

$$\begin{aligned} \mathbf{T}_{p,j+1} - \mathbf{T}_{p,j} &= (ah_j, ch_j) \\ \mathbf{T}_{p,j} - \mathbf{T}_{p,j-1} &= (ah_{j-1}, ch_{j-1}) \end{aligned} \tag{2.36}$$

Only the x-coordinate is given here, as the y direction is the same,

$$\begin{aligned} \left[\frac{\mathbf{T}_{p,j+1} - \mathbf{T}_{p,j}}{\|\mathbf{T}_{p,j+1} - \mathbf{T}_{p,j}\|} - \frac{\mathbf{T}_{p,j} - \mathbf{T}_{p,j-1}}{\|\mathbf{T}_{p,j} - \mathbf{T}_{p,j-1}\|} \right] &= \frac{ah_j}{h_j\sqrt{a^2+c^2}} - \frac{ah_{j-1}}{h_{j-1}\sqrt{a^2+c^2}} \\ &= 0 \end{aligned} \tag{2.37}$$

It follows that the curvature of the straight line, as computed using our formula, is zero. This formula cannot be used to compute a normal to the line, since even in general the normal to a straight line is not uniquely defined (although the addition of the right hand rule defines the normal in 2d).

2.1.3 Scheme on the Circle

Given that the zero curvature case is captured exactly, we now consider the case of constant curvature, which occurs in the circle in one dimension.

The circle is defined by the mapping

$$\mathbf{T}(\theta) = (R\cos(\theta), R\sin(\theta)). \tag{2.38}$$

As above, we define $h_j = \theta_{p,j+1} - \theta_{p,j}$ and proceed to apply the scheme to this mapping.

$$\begin{aligned}\|\mathbf{T}_{p,j+1} - \mathbf{T}_{p,j}\| &= \sqrt{(R\cos(\theta_{p,j} + h_j) - R\cos(\theta_{p,j}))^2 + (R\sin(\theta_{p,j} + h_j) - R\sin(\theta_{p,j}))^2} \\ &= \left(2R\sin\left(\frac{2\theta_{p,j} + h_j}{2}\right)\sin\left(\frac{h_j}{2}\right)\right)^2 + \left(2R\cos\left(\frac{2\theta_{p,j} + h_j}{2}\right)\sin\left(\frac{h_j}{2}\right)\right)^2 \\ &= 2R\sin\left(\frac{h_j}{2}\right)\end{aligned} \tag{2.39}$$

using the sum-to-product formulas

$$\cos(a) - \cos(b) = -2\sin\left(\frac{a+b}{2}\right)\sin\left(\frac{a-b}{2}\right) \tag{2.40}$$

and

$$\sin(a) - \sin(b) = 2\cos\left(\frac{a+b}{2}\right)\sin\left(\frac{a-b}{2}\right). \tag{2.41}$$

Using the law of cosines leads to the same result. Similarly, using the sum-to-product formulas leads to

$$\mathbf{T}_{p,j+1} - \mathbf{T}_{p,j} = \left(-2R\sin\left(\frac{2\theta_{p,j} + h_j}{2}\right)\sin\left(\frac{h_j}{2}\right), 2R\cos\left(\frac{2\theta_{p,j} + h_j}{2}\right)\sin\left(\frac{h_j}{2}\right)\right). \tag{2.42}$$

Rather than write the analogous formula for $j - 1$, we write

$$\mathbf{T}_{p,j} - \mathbf{T}_{p,j-1} = \left(-2R\sin\left(\frac{2\theta_{p,j} - h_{j-1}}{2}\right)\sin\left(\frac{h_{j-1}}{2}\right), 2R\cos\left(\frac{2\theta_{p,j} - h_{j-1}}{2}\right)\sin\left(\frac{h_{j-1}}{2}\right)\right) \tag{2.43}$$

and so, focusing on the X component,

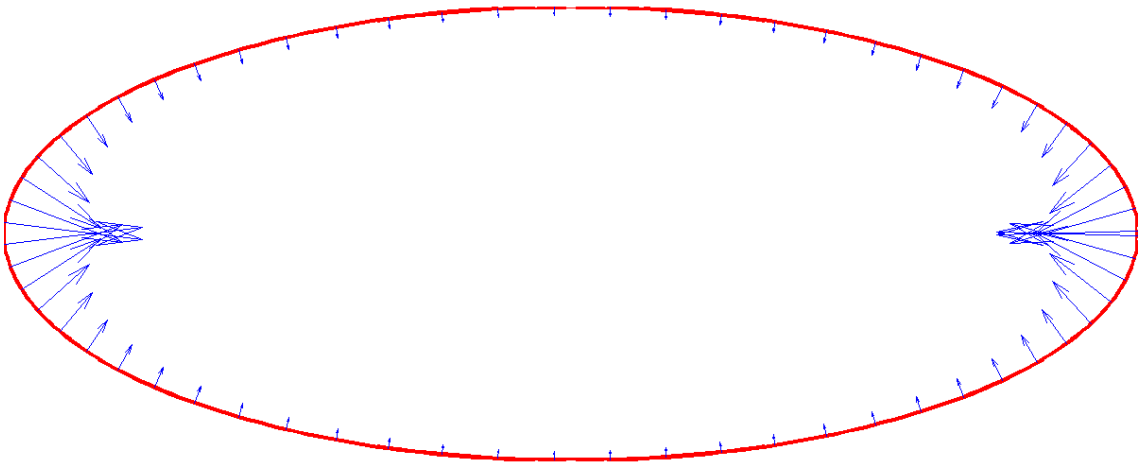

Figure 2.2: Ellipse with calculated mean curvature normals $\mathbf{H}$.

$$\begin{aligned}\left[\frac{\mathbf{T}_{p,j+1}-\mathbf{T}_{p,j}}{\|\mathbf{T}_{p,j+1}-\mathbf{T}_{p,j}\|}-\frac{\mathbf{T}_{p,j}-\mathbf{T}_{p,j-1}}{\|\mathbf{T}_{p,j}-\mathbf{T}_{p,j-1}\|}\right] &= \frac{-2R\sin\left(\frac{2\theta_{p,j}+h_j}{2}\right)\sin\left(\frac{h_j}{2}\right)}{2R\sin\left(\frac{h_j}{2}\right)} \\ &\quad - \frac{-2R\sin\left(\frac{2\theta_{p,j}-h_{j-1}}{2}\right)\sin\left(\frac{h_{j-1}}{2}\right)}{2R\sin\left(\frac{h_{j-1}}{2}\right)} \\ &= \sin\left(\frac{2\theta_{p,j}-h_{j-1}}{2}\right)-\sin\left(\frac{2\theta_{p,j}+h_j}{2}\right) \\ &= 2\cos\left(\frac{4\theta_{p,j}+h_j-h_{j-1}}{4}\right)\sin\left(\frac{h_j+h_{j-1}}{4}\right).\end{aligned} \tag{2.44}$$

Similar calculations on the Y component give the combined expression as,

$$\left[\frac{\mathbf{T}_{p,j+1}-\mathbf{T}_{p,j}}{\|\mathbf{T}_{p,j+1}-\mathbf{T}_{p,j}\|}-\frac{\mathbf{T}_{p,j}-\mathbf{T}_{p,j-1}}{\|\mathbf{T}_{p,j}-\mathbf{T}_{p,j-1}\|}\right]=\begin{bmatrix} 2\cos\left(\frac{4\theta_{p,j}+h_j-h_{j-1}}{4}\right)\sin\left(\frac{h_j+h_{j-1}}{4}\right) \\ -2\sin\left(\frac{4\theta_{p,j}+h_j-h_{j-1}}{4}\right)\sin\left(\frac{h_j+h_{j-1}}{4}\right)\end{bmatrix}. \tag{2.45}$$

The norm of this is given as

$$=2\sin\left(\frac{h_j+h_{j-1}}{4}\right). \tag{2.46}$$

Then, for our approximation of L(C), we define h_j^d as the difference between dual points to get

$$\|\mathbf{T}_{d,j+1}-\mathbf{T}_{d,j}\|=2R\sin\left(\frac{h_j^d}{2}\right). \tag{2.47}$$

Noting that since primal points are halfway between any two dual points, we get

$$\frac{h_j+h_{j-1}}{2}=h_j^d. \tag{2.48}$$

Then, κ is just

$$\begin{aligned}\kappa &= \frac{2\sin\left(\frac{h_j+h_{j-1}}{4}\right)}{2R\sin\left(\frac{h_j^d}{2}\right)} \\ &= \frac{\sin\left(\frac{h_j^d}{2}\right)}{R\sin\left(\frac{h_j^d}{2}\right)} \\ &= \frac{1}{R}\end{aligned} \tag{2.49}$$

which is the exact solution.

2.1.4 Results of the Scheme on an Ellipse

Given that we have shown that the scheme is accurate for the case of zero curvature and constant curvature, we then implemented the scheme in Matlab in order to test

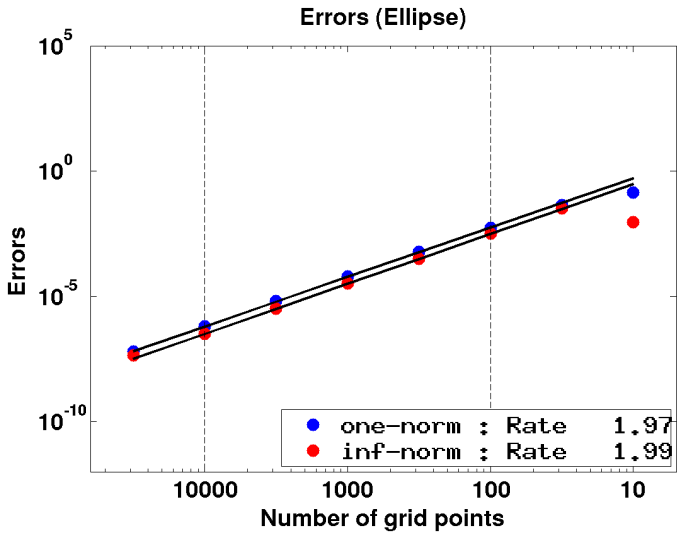

Figure 2.3: This figure shows second-order convergence of the curvature computation of our finite-volume scheme on an ellipse. The convergence rate is computed as the slope of the best fit line through the points between the vertical dashed lines.

it for cases in which proving convergence results is harder. We obtained second-order accurate results on mesh refinement.

The base test case is then the ellipse, as we have an exact formula for the curvature and the curvature varies over the ellipse. Since calculating the mesh for the ellipse is straightforward and calculating the curvature is also straightforward then we are able to test the scheme itself on mesh refinement, rather than running into precision of the points or curvature calculation.

The curvature of the ellipse is given by

$$\kappa = \frac{ab}{(b^2\cos(\xi)^2 + a^2\sin(\xi)^2)^{3/2}}, \tag{2.50}$$

where a, b, and ξ are from the parametric equations of the ellipse

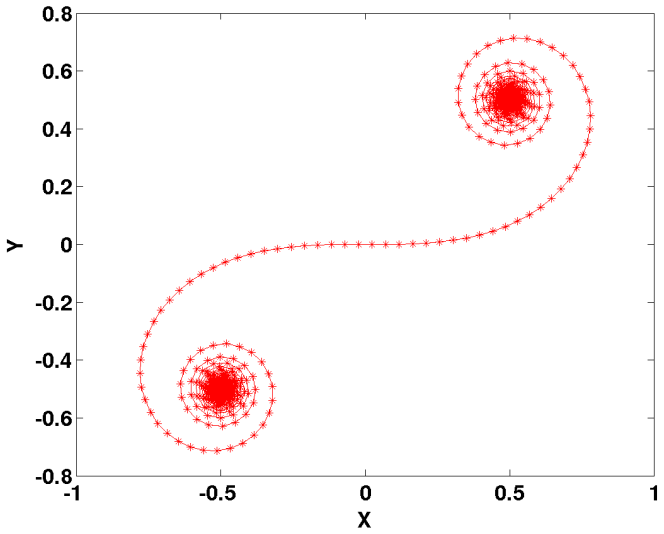

Figure 2.4: The Euler spiral from t = (-30, 30) with equal spacing in t.

$$\begin{aligned} X(\xi) &= a\cos(\xi), \\ Y(\xi) &= b\sin(\xi) \qquad -\pi \leq \xi < \pi. \end{aligned} \tag{2.51}$$

The results of the comparison, where $a = 5, b = 2$, are as shown in Figure 2.3 and a plot of the normals in Figure 2.2.

2.1.5 Results on the Euler Spiral

The Euler spiral, as seen in Figure 2.4, is a curve whose curvature changes linearly with the length of the curve making it an interesting test case for our scheme. It is defined in terms of Fresnel integrals given as

$$\begin{aligned} S(x) &= \int_0^x \sin(t^2)dt \\ C(x) &= \int_0^x \cos(t^2)dt. \end{aligned} \tag{2.52}$$

The Euler spiral is then $(C(t), S(t))$. It can be shown that the curvature is $\kappa = 2t$, where t is the arc-length. As $t \to \infty$ then $\kappa \to \infty$ giving two points around which the spiral infinitely winds. As t gets large, equally spaced values of t will fail to resolve the curve, and so any discrete curvature formula will have difficulty converging. If t is too large, then the limitations on floating point precision will make convergence impossible. The second issue is the ability to accurately compute the Fresnel integrals. Here, we use the method, and code for $C(t), S(t)$, presented by Alazah *et al.* [1] with the accuracy set to be about 10^{-15} for each integral, the standard Matlab functions are less accurate than the functions used. We surprisingly found that our scheme converged at a near quartic rate instead of the expected quadratic rate, see Figure 2.5. We think this likely due to the fact that the spiral locally approximates a circle causing cancellation of terms as in the circle. The last point on the plot diviates from the trend due to the precision of calculating the Fresnel integral.

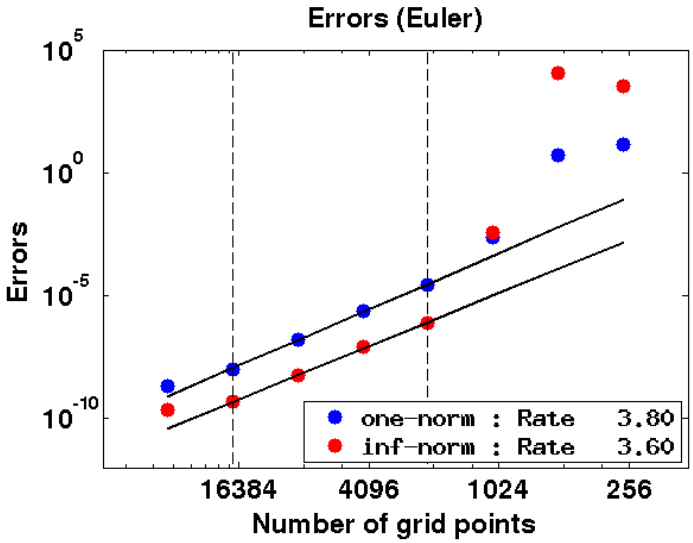

Figure 2.5: The quartic convergence of our scheme on the Euler spiral. For small numbers of grid points, the spiral is under-resolved. The convergence rate is the slope of the best fit line through the points between the vertical dashed lines.

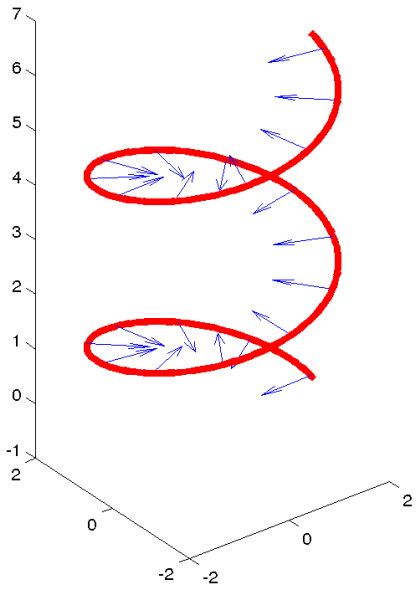

Figure 2.6: The helix and computed normals **H**.

2.2 Extending the Scheme to Curves in $\mathbb{R}^3$

The previous derivations extend directly to a non-planar curve,

$$\mathbf{T}(\xi) = (X(\xi), Y(\xi), Z(\xi)) \,. \tag{2.53}$$

The curvature and the mean curvature formulas derived in this section can be applied directly. A new quantity, torsion, denoted τ, and its vector, the binormal, apply for such curves. Neither torsion nor the binormal are interesting in the case where the curve exists logically on a plane.

To start, we approximate the derivative of the normals as

$$\frac{d\mathbf{n}_s}{ds} \approx \frac{\mathbf{n}_j - \mathbf{n}_{j-1}}{\|\mathbf{t}_j^{(1)}\|}. \tag{2.54}$$

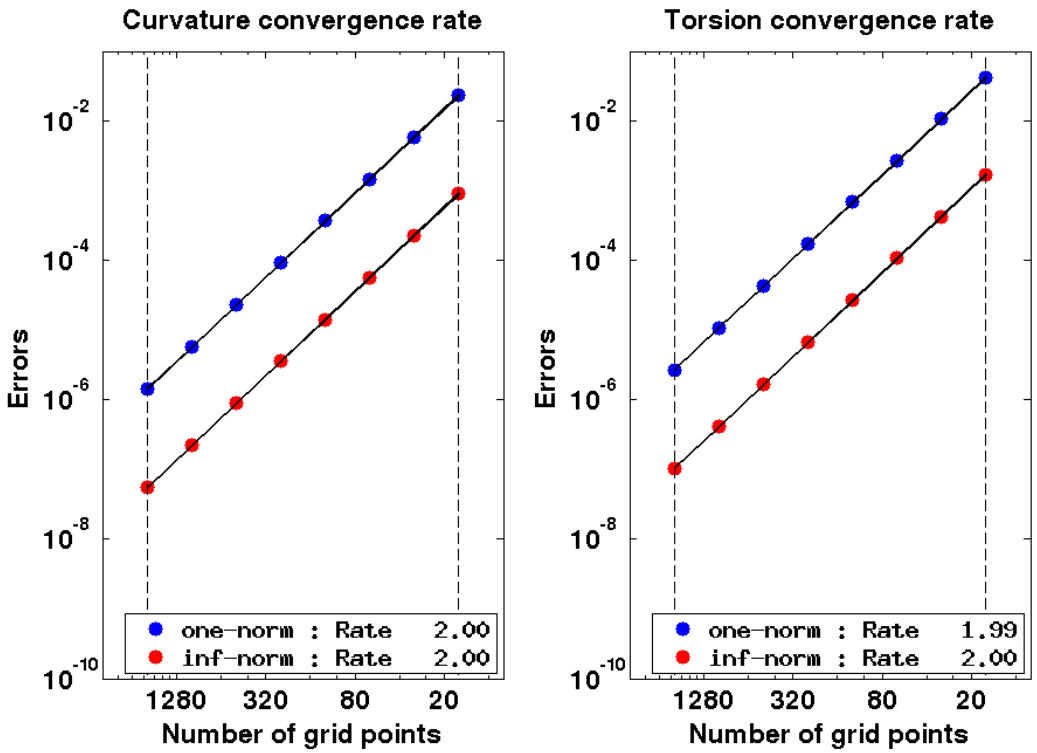

Figure 2.7: Convergence of curvature and torsion on the helix. The convergence rate is the slope of the best fit line through the points between the vertical dashed lines.

From the Frenet frame, equation (1.2), we have

$$\tau \mathbf{b} = \frac{d\mathbf{n}_s}{ds} + \kappa \frac{\mathbf{t}^{(1)}}{\|\mathbf{t}^{(1)}\|} \tag{2.55}$$

The torsion is then extracted by normalizing the binormal, the quantity on the right hand side of equation (2.55).

$$\tau = \|\tau \mathbf{b}\|. \tag{2.56}$$

The exact results obtained for the circle do not extend in an obvious way to curves with non-zero torsion, as seen in the helix, Figure 2.6. However, second-order convergence in curvature and torsion are observed, as in Figure 2.7.

CHAPTER 3

FINITE-VOLUME SCHEME FOR COMPUTING METRIC TERMS ON A TWO-DIMENSIONAL SURFACE

3.1 Derivation of the Scheme

We now extend the finite-volume discretization from the last chapter to certain two-dimensional surfaces. We start by assuming that we have a two-dimensional smooth surface described parametrically by the smooth mapping $\mathbf{T} : \mathbb{R}^2 \to \mathbb{R}^3$ where

$$\mathbf{T}(\xi, \eta) = (X(\xi, \eta), Y(\xi, \eta), Z(\xi, \eta)). \tag{3.1}$$

The basis vectors are given by

$$\begin{aligned} \mathbf{t}_{(1)} &= (X_\xi, Y_\xi, Z_\xi), \qquad \mathbf{t}_{(1)} \in \mathbb{R}^3 \\ \mathbf{t}_{(2)} &= (X_\eta, Y_\eta, Z_\eta), \qquad \mathbf{t}_{(2)} \in \mathbb{R}^3 \end{aligned} \tag{3.2}$$

and the metric tensor is given by

$$\mathbf{a} = \begin{bmatrix} \mathbf{t}_{(1)} \cdot \mathbf{t}_{(1)} & \mathbf{t}_{(1)} \cdot \mathbf{t}_{(2)} \\ \mathbf{t}_{(2)} \cdot \mathbf{t}_{(1)} & \mathbf{t}_{(2)} \cdot \mathbf{t}_{(2)} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}, \qquad \mathbf{a} \in \mathbb{R}^{2\times 2}. \tag{3.3}$$

The inverse metric tensor is given by

$$\mathbf{a}^{-1} = \frac{1}{\det(\mathbf{a})} \begin{bmatrix} a_{22} & -a_{21} \\ -a_{12} & a_{11} \end{bmatrix} = \begin{bmatrix} a^{11} & a^{12} \\ a^{21} & a^{22} \end{bmatrix}, \qquad \mathbf{a}^{-1} \in \mathbb{R}^{2\times 2}, \tag{3.4}$$

where $\det(\mathbf{a})$ is the determinant of $\mathbf{a}$ in equation (3.3).

We can now define a conjugate vector field

$$\begin{aligned} \mathbf{t}^{(1)} &= a^{11}\mathbf{t}_{(1)} + a^{12}\mathbf{t}_{(2)}, \qquad \mathbf{t}^{(1)} \in \mathbb{R}^3 \\ \mathbf{t}^{(2)} &= a^{12}\mathbf{t}_{(1)} + a^{22}\mathbf{t}_{(2)}, \qquad \mathbf{t}^{(2)} \in \mathbb{R}^3. \end{aligned} \tag{3.5}$$

This field gives us

$$\begin{aligned} \mathbf{t}_{(i)} \cdot \mathbf{t}^{(j)} &= \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases} \\ \mathbf{t}^{(i)} \cdot \mathbf{t}^{(j)} &= a^{ij}. \end{aligned} \tag{3.6}$$

Given any vector $\mathbf{w} \in \mathbb{R}^3$, we can write

$$\mathbf{w} = w^1\mathbf{t}_{(1)} + w^2\mathbf{t}_{(2)} \tag{3.7}$$

or

$$\mathbf{w} = w_1\mathbf{t}^{(1)} + w_2\mathbf{t}^{(2)} \tag{3.8}$$

where again w^1, w^2 are the contravarient components of $\mathbf{w}$ and w_1, w_2 are the covariant components.

This gives the relations

$$w^1 = \mathbf{w} \cdot \mathbf{t}^{(1)} = (w^1\mathbf{t}_{(1)}) \cdot \mathbf{t}^{(1)} \tag{3.9}$$

and

$$w_1 = \mathbf{w} \cdot \mathbf{t}_{(1)} = (w_1 \mathbf{t}^{(1)}) \cdot \mathbf{t}_{(1)}. \tag{3.10}$$

So given any $\mathbf{w}$ we can always find its covariant or contravariant components. The surface gradient of a scalar function $\phi(\xi, \eta)$ can be expressed in terms of its covariant components as

$$\widetilde{\nabla}\phi = \phi_\xi \mathbf{t}^{(1)} + \phi_\eta \mathbf{t}^{(2)} \tag{3.11}$$

As in the one-dimensional case, we now seek to formulate an expression for the surface normal and curvature using a finite-volume approach. We proceed here as in the one-dimensional case, but where now the formula for the curvature normal is the standard result from differential geometry

$$\nabla^2 \mathbf{T} = \nabla \cdot \widetilde{\nabla}\mathbf{T} = 2\mathbf{H}, \tag{3.12}$$

where $\mathbf{T}(\xi, \eta)$ is our mapping , $\nabla \cdot ()$ is the surface divergence operator and $\mathbf{H}$ is the surface curvature vector. From the divergence theorem on the surface, we can use the above to write

$$\int_S \nabla \cdot \widetilde{\nabla}\mathbf{T} dS = \int_{\partial S} \widetilde{\nabla}\mathbf{T} \cdot \mathbf{n}\, ds \tag{3.13}$$

where S is a quadrilateral surface patch, defined in terms of coordinate directions, $\widetilde{\nabla}\mathbf{T}$ is

$$\widetilde{\nabla}\mathbf{T} = \begin{pmatrix} \widetilde{\nabla}X \\ \widetilde{\nabla}Y \\ \widetilde{\nabla}Z \end{pmatrix} \tag{3.14}$$

and, as in the one-dimensional case,

$$\widetilde{\nabla}\mathbf{T}\cdot\mathbf{n}=\begin{pmatrix}\widetilde{\nabla}X\\ \widetilde{\nabla}Y\\ \widetilde{\nabla}Z\end{pmatrix}\cdot\mathbf{n}=\begin{pmatrix}\widetilde{\nabla}X\cdot\mathbf{n}\\ \widetilde{\nabla}Y\cdot\mathbf{n}\\ \widetilde{\nabla}Z\cdot\mathbf{n}\end{pmatrix}. \tag{3.15}$$

There are two "edge normals", $\mathbf{n}^{(1)}$ and $\mathbf{n}^{(2)}$, which can be written as

$$\begin{aligned}\mathbf{n}^{(1)}&=\frac{\mathbf{t}^{(1)}}{\|\mathbf{t}^{(1)}\|}\\ \mathbf{n}^{(2)}&=\frac{\mathbf{t}^{(2)}}{\|\mathbf{t}^{(2)}\|}.\end{aligned} \tag{3.16}$$

Focusing on $\mathbf{n}^{(1)}$, we can write

$$\begin{aligned}\widetilde{\nabla}X\cdot\mathbf{n}^{(1)}&=(X_\xi\mathbf{t}^{(1)}+X_\eta\mathbf{t}^{(2)})\cdot\frac{\mathbf{t}^{(1)}}{\|\mathbf{t}^{(1)}\|}=\frac{a^{11}X_\xi+a^{12}X_\eta}{\|\mathbf{t}^{(1)}\|}\\ \widetilde{\nabla}Y\cdot\mathbf{n}^{(1)}&=(Y_\xi\mathbf{t}^{(1)}+Y_\eta\mathbf{t}^{(2)})\cdot\frac{\mathbf{t}^{(1)}}{\|\mathbf{t}^{(1)}\|}=\frac{a^{11}Y_\xi+a^{12}Y_\eta}{\|\mathbf{t}^{(1)}\|}\\ \widetilde{\nabla}Z\cdot\mathbf{n}^{(1)}&=(Z_\xi\mathbf{t}^{(1)}+Z_\eta\mathbf{t}^{(2)})\cdot\frac{\mathbf{t}^{(1)}}{\|\mathbf{t}^{(1)}\|}=\frac{a^{11}Z_\xi+a^{12}Z_\eta}{\|\mathbf{t}^{(1)}\|},\end{aligned} \tag{3.17}$$

from which we get

$$\widetilde{\nabla}\mathbf{T}\cdot\mathbf{n}^{(1)}=\frac{a^{11}\mathbf{t}_{(1)}+a^{12}\mathbf{t}_{(2)}}{\|\mathbf{t}^{(1)}\|}=\frac{\mathbf{t}^{(1)}}{\|\mathbf{t}^{(1)}\|}\equiv\mathbf{n}^{(1)}. \tag{3.18}$$

Similar calculations lead to

$$\widetilde{\nabla}\mathbf{T}\cdot\mathbf{n}^{(2)}=\frac{a^{12}\mathbf{t}_{(1)}+a^{22}\mathbf{t}_{(2)}}{\|\mathbf{t}^{(2)}\|}=\frac{\mathbf{t}^{(2)}}{\|\mathbf{t}^{(2)}\|}\equiv\mathbf{n}^{(2)}. \tag{3.19}$$

Given a quadrilateral surface patch, S, aligned with the coordinate directions, we can now write

$$\begin{aligned}\frac{1}{\text{Area}(S)}\int_S \nabla\cdot\widetilde{\nabla}\mathbf{T}dA &= \frac{1}{\text{Area}(S)}\int_{\partial S}\widetilde{\nabla}\mathbf{T}\cdot\mathbf{n}ds = \frac{1}{\text{Area}(S)}\int_{\partial S}\mathbf{n}\cdot ds \\ &\approx \frac{1}{\text{Area}(S)}\left\{\sum_{k=1}^{4}\mathbf{n}_k\right\}\end{aligned} \tag{3.20}$$

where Area(S) is the area and the $\mathbf{n}_k$ are the four edge normals, taken in a counter clockwise direction, of the quadrilateral, S.

The curvature normal vector can then be written as

$$\mathbf{H} \approx \frac{1}{2\text{Area}(S)}\left\{\sum_{k=1}^{4}\mathbf{n}_k\Delta s_k\right\}, \tag{3.21}$$

where Δs_k is the length of the k-th edge of the patch S. From this, we have the surface normal and curvature given by:

$$\mathbf{n}_s = \frac{\mathbf{H}}{\|\mathbf{H}\|} = \frac{\mathbf{H}}{\kappa}, \tag{3.22}$$

where

$$\kappa = \|\mathbf{H}\|. \tag{3.23}$$

3.2 The Computational Scheme on a Surface Mesh

Computationally, we can approximate the basis vectors. We set up two sets of points on the surface: the primal points $\{\mathbf{T}_{p,i,j}\}$ and the dual points $\{\mathbf{T}_{d,i,j}\}$ as in Figure 3.1. Then

$$\mathbf{t}_{(1),i,j} = \mathbf{T}_{p,i,j} - \mathbf{T}_{p,i,j-1} \tag{3.24}$$

and

$$\mathbf{t}_{(2),i,j} = \mathbf{T}_{d,i+1,j} - \mathbf{T}_{d,i,j}. \tag{3.25}$$

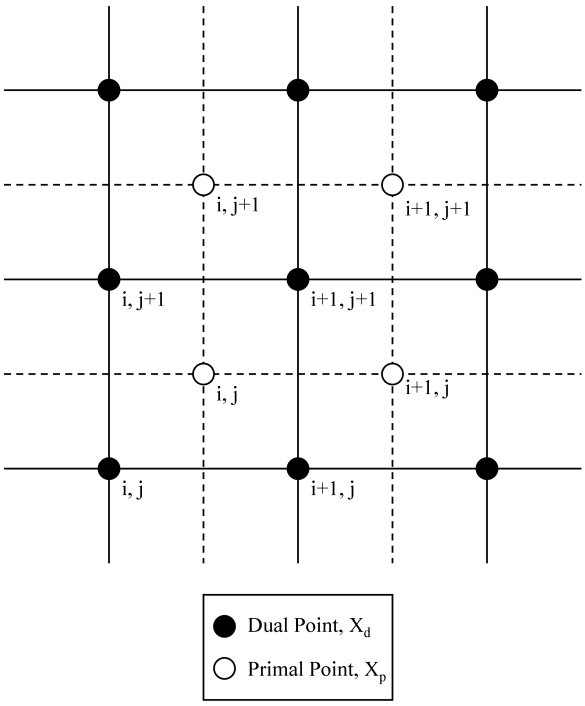

Figure 3.1: Computational grid showing primal and dual points. Dual points are the cell nodes and primal points are the cell centers.

This leads to the construction of the **a** matrix as

$$\mathbf{a} = \begin{bmatrix} (\mathbf{T}_{p,i,j} - \mathbf{T}_{p,i,j-1}) \cdot (\mathbf{T}_{p,i,j} - \mathbf{T}_{p,i,j-1}) & (\mathbf{T}_{p,i,j} - \mathbf{T}_{p,i,j-1}) \cdot (\mathbf{T}_{d,i+1,j} - \mathbf{T}_{d,i,j}) \\ (\mathbf{T}_{p,i,j} - \mathbf{T}_{p,i,j-1}) \cdot (\mathbf{T}_{d,i+1,j} - \mathbf{T}_{d,i,j}) & (\mathbf{T}_{d,i+1,j} - \mathbf{T}_{d,i,j}) \cdot (\mathbf{T}_{d,i+1,j} - \mathbf{T}_{d,i,j}) \end{bmatrix}. \tag{3.26}$$

Since **a** is a two-by-two matrix, the inverse is easily computed. The determinant, which will always be positive for non-degenerate meshes, is

$$\det(\mathbf{a}) = a_{1,1}a_{2,2} - a_{1,2}a_{2,1} \tag{3.27}$$

leading to the inverse as

$$\mathbf{a}^{-1} = \frac{1}{\det(\mathbf{a})} \cdot \begin{bmatrix} (\mathbf{T}_{d,i+1,j} - \mathbf{T}_{d,i,j}) \cdot (\mathbf{T}_{d,i+1,j} - \mathbf{T}_{d,i,j}) & -(\mathbf{T}_{p,i,j} - \mathbf{T}_{p,i,j-1}) \cdot (\mathbf{T}_{d,i+1,j} - \mathbf{T}_{d,i,j}) \\ -(\mathbf{T}_{p,i,j} - \mathbf{T}_{p,i,j-1}) \cdot (\mathbf{T}_{d,i+1,j} - \mathbf{T}_{d,i,j}) & (\mathbf{T}_{p,i,j} - \mathbf{T}_{p,i,j-1}) \cdot (\mathbf{T}_{p,i,j} - \mathbf{T}_{p,i,j-1}) \end{bmatrix}. \tag{3.28}$$

Edge normals are then given by

$$\mathbf{n}_{\xi_{i,j}} = \frac{a^{11}(\mathbf{T}_{p,i,j} - \mathbf{T}_{p,i,j-1}) + a^{22}(\mathbf{T}_{d,i+1,j} - \mathbf{T}_{d,i,j})}{\|a^{11}(\mathbf{T}_{p,i,j} - \mathbf{T}_{p,i,j-1}) + a^{22}(\mathbf{T}_{d,i+1,j} - \mathbf{T}_{d,i,j})\|} \tag{3.29}$$

and Δs_k, the length of the k-th edge of the patch S, is approximated as

$$\Delta s_k \approx \|\mathbf{T}_{d,k+1} - \mathbf{T}_{d,k}\|. \tag{3.30}$$

From the edge normals, we can compute the surface normals, $\mathbf{n}_{s,i,j}$, and curvature $\kappa_{i,j}$ at cell centers. We approximate the area of a cell by assuming the surface spanning the mesh cell has the bilinear function, as done in Calhoun and Helzel [3],

$$S(u, v) = \mathbf{c}_{00} + \mathbf{c}_{01}u + \mathbf{c}_{10}v + \mathbf{c}_{11}uv \tag{3.31}$$

where coefficients $\mathbf{c}_{k\ell} \in \mathbb{R}^3$ are computed from the known locations of the mesh cell corners. Using these corner points for $\mathbf{T}_{d,i+k,j+\ell}$ for $k, \ell = 0, 1$, we get

$$\begin{aligned} \mathbf{c}_{00} &= \mathbf{T}_{d,i,j}, \\ \mathbf{c}_{01} &= \mathbf{T}_{d,i+1,j} - \mathbf{T}_{d,i,j}, \\ \mathbf{c}_{10} &= \mathbf{T}_{d,i,j+1} - \mathbf{T}_{d,i,j}, \\ \mathbf{c}_{11} &= \mathbf{T}_{d,i+1,j+1} - \mathbf{T}_{d,i+1,j} - \mathbf{T}_{d,i,j+1} + \mathbf{T}_{d,i,j}. \end{aligned} \tag{3.32}$$

We can write down the exact area of a mesh cell with surface mapping $S(u, v)$ as

$\int_{[0,1]x[0,1]} \|S_u \times S_v\| dudv$. For our purposes, we approximate this area using a midpoint rule to evaluate the integral and obtain

$$\text{Area}(S_{i,j}) \approx \|(\mathbf{c}_{01} + \frac{\mathbf{c}_{11}}{2}) \times (\mathbf{c}_{10} + \frac{\mathbf{c}_{11}}{2})\|. \tag{3.33}$$

Using equation (3.21), we then have

$$\mathbf{H} \approx \frac{1}{2\|(\mathbf{c}_{01} + \frac{\mathbf{c}_{11}}{2}) \times (\mathbf{c}_{10} + \frac{\mathbf{c}_{11}}{2})\|} \left\{ \sum_{k=1}^{4} \frac{{\mathbf{t}^{(1)}}_{i,j}}{\|{\mathbf{t}^{(1)}}_{i,j}\|} \|\mathbf{T}_{d,k+1} - \mathbf{T}_{d,k}\| \right\}, \tag{3.34}$$

and from equation (3.23)

$$\kappa \approx \left\| \frac{1}{2\|(\mathbf{c}_{01} + \frac{\mathbf{c}_{11}}{2}) \times (\mathbf{c}_{10} + \frac{\mathbf{c}_{11}}{2})\|} \left\{ \sum_{k=1}^{4} \frac{{\mathbf{t}^{(1)}}_{i,j}}{\|{\mathbf{t}^{(1)}}_{i,j}\|} \|\mathbf{T}_{d,k+1} - \mathbf{T}_{d,k}\| \right\} \right\|. \tag{3.35}$$

3.3 Results of the Scheme in Two Dimensions

We implemented the scheme in Fortran and obtained second-order accurate results for smooth surface mappings and first-order for piecewise smooth mappings. The base case was the sphere as exact results were obtained on the cylinder. We also investigated hyperboloids, cylinders, paraboloids, and hyperbolic tangent sheets. All showed second-order convergence or better.

3.3.1 The Scheme on a Cylinder

Since the circle is exact in one dimension, we checked if the cylinder has the same properties as the circle in one dimension. Indeed, numerical testing of the scheme shows the results on the cylinder are precisely the same as that of a circle in one-

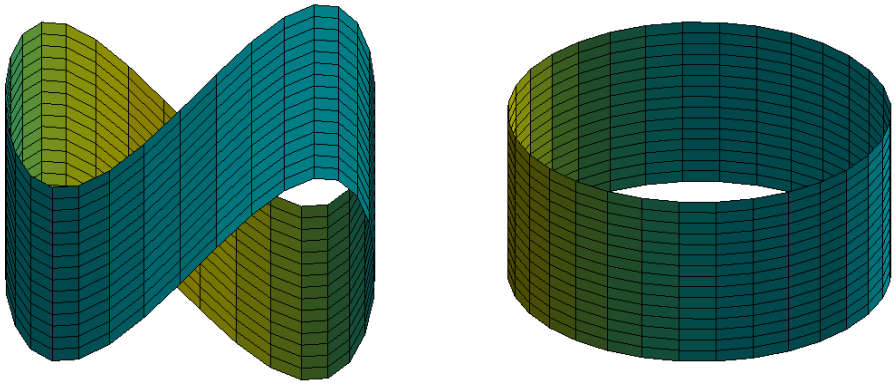

Figure 3.2: Cylinder mapping, in the left figure $f(\theta) = \sin(2\theta)$ and in the right $f(\theta) = 0$. The color and lighting in this figure is to show the shape and otherwise has no meaning.

dimension. Working through the equations should likewise lead to the exact result as it did for the circle.

In order for the mapping to be more general we introduce $f(\theta)$, a function operating in the Z direction, which depends only on θ. This gives us our general parametric mapping as

$$\begin{aligned} X(\theta,\zeta) &= R\cos(\theta) \\ Y(\theta,\zeta) &= R\sin(\theta) \\ Z(\theta,\zeta) &= \zeta + f(\theta). \end{aligned} \tag{3.36}$$

We use the notation from the circle proof with the added notation of $\Delta Z_{i-1/2,j} = Z_{i,j} - Z_{i-1,j}$, which depends solely on θ, while noting that differences ΔX and ΔY can be assumed to be 1. Also note that $\theta_{d,i,j} = \theta_{d,i,j-1}$, which leads to

$$\begin{aligned}\mathbf{t}_{(1),i,j} &= \begin{bmatrix} R\cos(\theta_{p,i,j}) - R\cos(\theta_{p,i-1,j}) \\ R\sin(\theta_{p,i,j}) - R\sin(\theta_{p,i-1,j}) \\ \zeta_{p,i,j} - \zeta_{p,i-1,j} \end{bmatrix} \\ &= \begin{bmatrix} -2R\sin\left(\frac{2\theta_{p,i,j}+h_{p,i,j}}{2}\right)\sin\left(\frac{h_{p,i,j}}{2}\right) \\ 2R\cos\left(\frac{2\theta_{p,i,j}+h_{p,i,j}}{2}\right)\sin\left(\frac{h_{p,i,j}}{2}\right) \\ \Delta Z_{i-1/2,j} \end{bmatrix}.\end{aligned} \tag{3.37}$$

Likewise, for the other basis vector we get

$$\mathbf{t}_{(2),i,j} = \begin{bmatrix} R\cos(\theta_{d,i,j}) - R\cos(\theta_{d,i,j-1}) \\ R\sin(\theta_{d,i,j}) - R\sin(\theta_{d,i,j-1}) \\ \zeta_{d,i,j} - \zeta_{d,i,j-1} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}. \tag{3.38}$$

For the $\mathbf{a}$, we have

$$\begin{aligned} a_{11} &= (-2R\sin\left(\frac{2\theta_{p,i,j}+h_{p,i,j}}{2}\right)\sin\left(\frac{h_{p,i,j}}{2}\right))^2 \\ &+ (2R\cos\left(\frac{2\theta_{p,i,j}+h_{p,i,j}}{2}\right)\sin\left(\frac{h_{p,i,j}}{2}\right))^2 \\ &+ \Delta Z_{i-1/2,j}^2 \\ &= 4R^2\sin^2\left(\frac{h_{p,i,j}}{2}\right) + \Delta Z_{i-1/2,j}^2 \end{aligned} \tag{3.39}$$

and

$$a_{12} = a_{21} = \Delta Z_{i-1/2,j} \tag{3.40}$$

and

$$a_{22} = 1. \tag{3.41}$$

The determinant is then given as

$$\det(\mathbf{a}) = a_{11}a_{22} - a_{12}a_{21}, \tag{3.42}$$

which is best calculated in pieces and then combined as follows:

$$\begin{aligned} a_{12}a_{21} &= \Delta Z^2_{i-1/2,j} \\ a_{11}a_{22} &= 4R^2 \sin^2\left(\frac{h_{p,i,j}}{2}\right) + \Delta Z^2_{i-1/2,j}. \end{aligned} \tag{3.43}$$

Combining these two parts leads to $\det(\mathbf{a})$ as

$$a_{11}a_{22} - a_{12}a_{21} = 4R^2 \sin^2\left(\frac{h_{p,i,j}}{2}\right). \tag{3.44}$$

For the $\mathbf{a}^{-1}$ matrix we get

$$a^{11} = \frac{1}{4R^2 \sin^2\left(\frac{h_{p,i,j}}{2}\right)} \tag{3.45}$$

$$a^{22} = 1 + \frac{\Delta Z^2_{i-1/2,j}}{4R^2 \sin^2\left(\frac{h_{p,i,j}}{2}\right)} \tag{3.46}$$

$$a^{12} = -\frac{\Delta Z_{i-1/2,j}}{4R^2 \sin^2\left(\frac{h_{p,i,j}}{2}\right)}. \tag{3.47}$$

This leads to

$$\mathbf{t}_1^{(1)} = -\frac{2R\sin\left(\frac{2\theta_{p,i,j}+h_{p,i,j}}{2}\right)\sin\left(\frac{h_{p,i,j}}{2}\right)}{4R^2\sin^2\left(\frac{h_{p,i,j}}{2}\right)} = -\frac{\sin\left(\frac{2\theta_{p,i,j}+h_{p,i,j}}{2}\right)}{2R\sin\left(\frac{h_{p,i,j}}{2}\right)}$$

$$\mathbf{t}_2^{(1)} = \frac{2R\cos\left(\frac{2\theta_{p,i,j}+h_{p,i,j}}{2}\right)\sin\left(\frac{h_{p,i,j}}{2}\right)}{4R^2\sin^2\left(\frac{h_{p,i,j}}{2}\right)} = \frac{\cos\left(\frac{2\theta_{p,i,j}+h_{p,i,j}}{2}\right)}{2R\sin\left(\frac{h_{p,i,j}}{2}\right)} \tag{3.48}$$

$$\mathbf{t}_3^{(1)} = \frac{\Delta Z_{i-1/2,j}}{4R^2\sin^2\left(\frac{h_{p,i,j}}{2}\right)} - \frac{\Delta Z_{i-1/2,j}}{4R^2\sin^2\left(\frac{h_{p,i,j}}{2}\right)} = 0.$$

Then $\mathbf{t}^{(2)}$ follows

$$\mathbf{t}_1^{(2)} = \frac{2R\Delta Z_{i-1/2,j}\sin\left(\frac{2\theta_{p,i,j}+h_{p,i,j}}{2}\right)\sin\left(\frac{h_{p,i,j}}{2}\right)}{4R^2\sin^2\left(\frac{h_{p,i,j}}{2}\right)} = -\frac{\Delta Z_{i-1/2,j}\sin\left(\frac{2\theta_{p,i,j}+h_{p,i,j}}{2}\right)}{2R\sin\left(\frac{h_{p,i,j}}{2}\right)} \tag{3.49}$$

$$\mathbf{t}_2^{(2)} = -\frac{2R\Delta Z_{i-1/2,j}\cos\left(\frac{2\theta_{p,i,j}+h_{p,i,j}}{2}\right)\sin\left(\frac{h_{p,i,j}}{2}\right)}{4R^2\sin^2\left(\frac{h_{p,i,j}}{2}\right)} = -\frac{\Delta Z_{i-1/2,j}\cos\left(\frac{2\theta_{p,i,j}+h_{p,i,j}}{2}\right)}{2R\sin\left(\frac{h_{p,i,j}}{2}\right)} \tag{3.50}$$

$$\mathbf{t}_3^{(2)} = -\frac{\Delta Z_{i-1/2,j}^2}{4R^2\sin\left(\frac{h_{p,i,j}}{2}\right)} + 1 + \frac{\Delta Z_{i-1/2,j}^2}{4R^2\sin\left(\frac{h_{p,i,j}}{2}\right)} = 1. \tag{3.51}$$

The norm of $\mathbf{t}^{(1)}$ is

$$\|\mathbf{t}^{(1)}\| = \frac{1}{2R\sin\left(\frac{h_{p,i,j}}{2}\right)}. \tag{3.52}$$

The norm of $\mathbf{t}^{(2)}$ is

$$\|\mathbf{t}^{(2)}\| = \sqrt{\left(\frac{\Delta Z_{i-1/2,j}\sin\left(\frac{2\theta_{p,i,j}+h_{p,i,j}}{2}\right)}{2R\sin\left(\frac{h_{p,i,j}}{2}\right)}\right)^2 + \left(\frac{\Delta Z_{i-1/2,j}\cos\left(\frac{2\theta_{p,i,j}+h_{p,i,j}}{2}\right)}{2R\sin\left(\frac{h_{p,i,j}}{2}\right)}\right)^2 + (1)^2} \tag{3.53}$$

$$= \sqrt{\left(\frac{\Delta Z_{i-1/2,j}}{2R\sin\left(\frac{h_{p,i,j}}{2}\right)}\right)^2 + 1} = \frac{\sqrt{\Delta Z_{i-1/2,j}^2 + 4R^2\sin^2\left(\frac{h_{p,i,j}}{2}\right)}}{2R\sin\left(\frac{h_{p,i,j}}{2}\right)}. \tag{3.54}$$

From this, we can now normalize $\mathbf{t}^{(1)}$ and $\mathbf{t}^{(2)}$. This again must be done component-wise

$$\frac{\mathbf{t}_1^{(1)}}{\|\mathbf{t}^{(1)}\|} = -\frac{\frac{\sin\left(\frac{2\theta_{p,i,j}+h_{p,i,j}}{2}\right)}{2R\sin\left(\frac{h_{p,i,j}}{2}\right)}}{\frac{1}{2R\sin\left(\frac{h_{p,i,j}}{2}\right)}} = -\sin\left(\frac{2\theta_{p,i,j}+h_{p,i,j}}{2}\right) \tag{3.55}$$

$$\frac{\mathbf{t}_2^{(1)}}{\|\mathbf{t}^{(1)}\|} = \frac{\frac{\cos\left(\frac{2\theta_{p,i,j}+h_{p,i,j}}{2}\right)}{2R\sin\left(\frac{h_{p,i,j}}{2}\right)}}{\frac{1}{2R\sin\left(\frac{h_{p,i,j}}{2}\right)}} = \cos\left(\frac{2\theta_{p,i,j}+h_{p,i,j}}{2}\right) \tag{3.56}$$

$$\frac{\mathbf{t}_3^{(1)}}{\|\mathbf{t}^{(1)}\|} = 0, \tag{3.57}$$

which is convenient. The normalization of $\mathbf{t}^{(2)}$ is

$$\frac{\mathbf{t}_1^{(2)}}{\|\mathbf{t}^{(2)}\|} = \frac{\frac{\Delta Z_{i-1/2,j}\sin\left(\frac{2\theta_{p,i,j}+h_{p,i,j}}{2}\right)}{2R\sin\left(\frac{h_{p,i,j}}{2}\right)}}{\frac{\sqrt{\Delta Z_{i-1/2,j}^2+4R^2\sin^2\left(\frac{h_{p,i,j}}{2}\right)}}{2R\sin\left(\frac{h_{p,i,j}}{2}\right)}} = \frac{\Delta Z_{i-1/2,j}\sin\left(\frac{2\theta_{p,i,j}+h_{p,i,j}}{2}\right)}{\sqrt{\Delta Z_{i-1/2,j}^2 + 4R^2\sin^2\left(\frac{h_{p,i,j}}{2}\right)}} \tag{3.58}$$

$$\frac{\mathbf{t}_2^{(2)}}{\|\mathbf{t}^{(2)}\|} = \frac{\frac{\Delta Z_{i-1/2,j}\cos\left(\frac{2\theta_{p,i,j}+h_{p,i,j}}{2}\right)}{2R\sin\left(\frac{h_{p,i,j}}{2}\right)}}{\frac{\sqrt{\Delta Z_{i-1/2,j}^2+4R^2\sin^2\left(\frac{h_{p,i,j}}{2}\right)}}{2R\sin\left(\frac{h_{p,i,j}}{2}\right)}} = \frac{\Delta Z_{i-1/2,j}\cos\left(\frac{2\theta_{p,i,j}+h_{p,i,j}}{2}\right)}{\sqrt{\Delta Z_{i-1/2,j}^2+4R^2\sin^2\left(\frac{h_{p,i,j}}{2}\right)}} \tag{3.59}$$

$$\frac{\mathbf{t}_3^{(2)}}{\|\mathbf{t}^{(2)}\|} = \frac{1}{\frac{\sqrt{\Delta Z_{i-1/2,j}^2+4R^2\sin^2\left(\frac{h_{p,i,j}}{2}\right)}}{2R\sin\left(\frac{h_{p,i,j}}{2}\right)}} = \frac{2R\sin\left(\frac{h_{p,i,j}}{2}\right)}{\sqrt{\Delta Z_{i-1/2,j}^2+4R^2\sin^2\left(\frac{h_{p,i,j}}{2}\right)}}. \tag{3.60}$$

An interesting point to notice here is that if $\Delta Z_{i-1/2,j} = 0$, as is the case in the traditional parametric mapping of the cylinder, then $\mathbf{t}^{(2)}$ and its norm would be greatly simplified. Now $\|\mathbf{X}_k - \mathbf{X}_{k-1}\|$ can be either the norm of $\mathbf{t}_{(1)}$ or $\mathbf{t}_{(2)}$ depending on the side being examined, just as $\mathbf{t}^{(1)}$ and $\mathbf{t}^{(2)}$ are used for different portions of the grid. So doing first one direction and then the other

$$\|\mathbf{T}_k - \mathbf{T}_{k-1}\| = \|\mathbf{t}_{(1)}\| = \sqrt{a_{1,1}} = \sqrt{4R^2\sin^2\left(\frac{h_{p,i,j}}{2}\right) + \Delta Z_{i-1/2,j}^2} \tag{3.61}$$

or

$$\|\mathbf{T}_k - \mathbf{T}_{k-1}\| = \|\mathbf{t}_{(2)}\| = \sqrt{a_{2,2}} = 1 \tag{3.62}$$

These are convenient as they lead to

$$\frac{\mathbf{t}^{(1)}}{\|\mathbf{t}^{(1)}\|}\|\mathbf{T}_k - \mathbf{T}_{k-1}\| = \frac{\mathbf{t}^{(1)}}{\|\mathbf{t}^{(1)}\|}, \tag{3.63}$$

which was given previously and

$$\frac{\mathbf{t}_1^{(2)}}{\|\mathbf{t}^{(2)}\|}\|\mathbf{T}_k - \mathbf{T}_{k-1}\| = \frac{\Delta Z_{i-1/2,j}\sin\left(\frac{2\theta_{p,i,j}+h_{p,i,j}}{2}\right)}{\sqrt{\Delta Z_{i-1/2,j}^2 + 4R^2\sin^2\left(\frac{h_{p,i,j}}{2}\right)}}\sqrt{4R^2\sin^2\left(\frac{h_{p,i,j}}{2}\right) + \Delta Z_{i-1/2,j}^2}$$
$$= \Delta Z_{i-1/2,j}\sin\left(\frac{2\theta_{p,i,j}+h_{p,i,j}}{2}\right), \tag{3.64}$$

$$\frac{\mathbf{t}_2^{(2)}}{\|\mathbf{t}^{(2)}\|}\|\mathbf{T}_k - \mathbf{T}_{k-1}\| = \frac{\Delta Z_{i-1/2,j}\cos\left(\frac{2\theta_{p,i,j}+h_{p,i,j}}{2}\right)}{\sqrt{\Delta Z_{i-1/2,j}^2 + 4R^2\sin^2\left(\frac{h_{p,i,j}}{2}\right)}}\sqrt{4R^2\sin^2\left(\frac{h_{p,i,j}}{2}\right) + \Delta Z_{i-1/2,j}^2}$$

$$= \Delta Z_{i-1/2,j}\cos\left(\frac{2\theta_{p,i,j}+h_{p,i,j}}{2}\right), \tag{3.65}$$

$$\frac{\mathbf{t}_3^{(2)}}{\|\mathbf{t}^{(2)}\|}\|\mathbf{T}_k - \mathbf{T}_{k-1}\| = \frac{2R\sin\left(\frac{h_{p,i,j}}{2}\right)}{\sqrt{\Delta Z_{i-1/2,j}^2 + 4R^2\sin^2\left(\frac{h_{p,i,j}}{2}\right)}}\sqrt{4R^2\sin^2\left(\frac{h_{p,i,j}}{2}\right) + \Delta Z_{i-1/2,j}^2}$$

$$= 2R\sin\left(\frac{h_{p,i,j}}{2}\right). \tag{3.66}$$

We are now looking at four "edge normals," two in each direction, as well as the four dual points, making up a grid, which are used to calculate the area of the cell. The major point to notice is that while the derivations are the same, the numbering is not. First notice that

$$\mathbf{c}_{01} = \mathbf{T}_{d,i+1,j} - \mathbf{T}_{d,i,j} = \mathbf{t}_{((2),i+1,j)}, \tag{3.67}$$

which becomes, using the simplification assumptions,

$$\mathbf{t}_{((2),i+1,j)} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}. \tag{3.68}$$

Then, note that

$$\mathbf{c}_{10} = \mathbf{T}_{d,i,j+1} - \mathbf{T}_{d,i,j} = \mathbf{t}_{((1),i,j+1)}, \tag{3.69}$$

which is given previously, and then

$$\begin{aligned} \mathbf{c}_{11} &= \mathbf{T}_{d,i+1,j+1} - \mathbf{T}_{d,i+1,j} - \mathbf{T}_{d,i,j+1} + \mathbf{T}_{d,i,j} \\ &= (\mathbf{T}_{d,i+1,j+1} - \mathbf{T}_{d,i,j+1}) - (\mathbf{T}_{d,i+1,j} - \mathbf{T}_{d,i,j}) \\ &= \mathbf{t}_{((1),i+1,j+1)} - \mathbf{t}_{((1),i+1,j)} = 0. \end{aligned} \tag{3.70}$$

The last simplification comes from noticing that the change is only in the Z direction and that $\mathbf{t}_{(1)}$ does not vary in Z. Then, the area calculation becomes

$$\|(\mathbf{c}_{01}) \times (\mathbf{c}_{10})\| = \|(\mathbf{t}_{(2)} \times \mathbf{t}_{(1)})\| \tag{3.71}$$

$$= \sqrt{\left(2R\sin\left(\frac{2\theta_{p,i,j} + h_{p,i,j}}{2}\right)\sin\left(\frac{h_{p,i,j}}{2}\right)\right)^2 + \left(2R\cos\left(\frac{2\theta_{p,i,j} + h_{p,i,j}}{2}\right)\sin\left(\frac{h_{p,i,j}}{2}\right)\right)^2} \tag{3.72}$$

$$= 2R\sin\left(\frac{h_{d,i,j}}{2}\right). \tag{3.73}$$

Now clearly the $\mathbf{t}^{(2)}$ portions of the summation simplifies as $\Delta Z_{i-1/2,j} = \Delta Z_{i-1/2,j+1}$

which leaves just the $\mathbf{t}^{(1)}$ portion, which is very similar to the circle case. For X we get

$$\begin{aligned}\sin\left(\frac{2\theta_{p,i,j}+h_{p,i,j}}{2}\right)-\sin\left(\frac{2\theta_{p,i,j}+h_{p,i,j}}{2}\right)=\\ 2\cos\left(\frac{2\theta_{p,i,j}+2\theta_{p,i+1,j}+h_{p,i,j}-h_{p,i+1,j}}{4}\right)\sin\left(\frac{h_{p,i+1,j}+h_{p,i,j}}{4}\right),\end{aligned} \tag{3.74}$$

while similar calculation for Y give

$$\left\{\sum_{k=1}^{4}\frac{{\mathbf{t}^{(1)}}_{i,j}}{\|{\mathbf{t}^{(1)}}_{i,j}\|}\|\mathbf{T}_{d,k+1}-\mathbf{T}_{d,k}\|\right\}=\begin{bmatrix}2\cos\left(\frac{2\theta_{p,i,j}+2\theta_{p,i+1,j}+h_{p,i,j}-h_{p,i+1,j}}{4}\right)\sin\left(\frac{h_{p,i+1,j}+h_{p,i,j}}{4}\right)\\ 2\sin\left(\frac{2\theta_{p,i,j}+2\theta_{p,i+1,j}+h_{p,i,j}-h_{p,i+1,j}}{4}\right)\sin\left(\frac{h_{p,i+1,j}+h_{p,i,j}}{4}\right)\\ 0\end{bmatrix} \tag{3.75}$$

The norm of which leads to

$$\left\|\left\{\sum_{k=1}^{4}\frac{{\mathbf{t}^{(1)}}_{i,j}}{\|{\mathbf{t}^{(1)}}_{i,j}\|}\|\mathbf{T}_{d,k+1}-\mathbf{T}_{d,k}\|\right\}\right\|=2\sin\left(\frac{h_{p,i+1,j}+h_{p,i,j}}{4}\right), \tag{3.76}$$

and this is, just like in the circle case

$$2\sin\left(\frac{h_{p,i+1,j}+h_{p,i,j}}{4}\right)=2\sin\left(\frac{h_{d,i,j}}{2}\right). \tag{3.77}$$

This leads directly to

$$\begin{aligned}\kappa&=\frac{2\sin\left(\frac{h_{d,i,j}}{2}\right)}{4R\sin\left(\frac{h_{d,i,j}}{2}\right)}\\ &=\frac{1}{2R},\end{aligned} \tag{3.78}$$

which is the exact analytical solution of the mean curvature for a cylinder.

3.3.2 Results on Minimal Surfaces

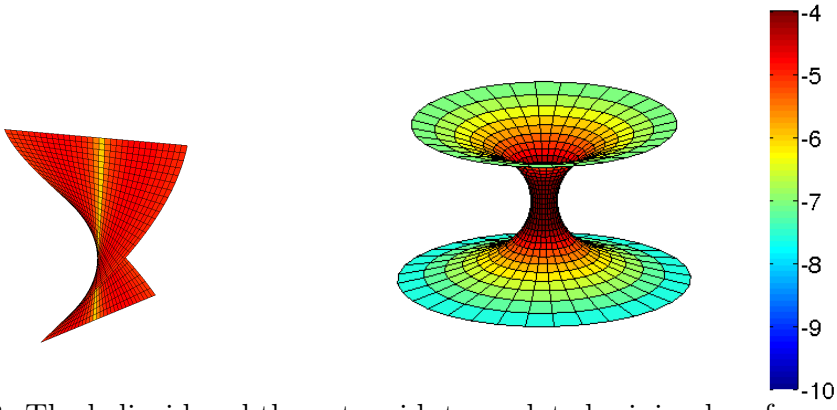

Figure 3.3: The helicoid and the catenoid, two related minimal surfaces, having mean curvature zero, colored by errors. The color bar applies to both figures.

A minimal surface is defined as one which has a constant mean curvature of zero. The trivial example of a minimal surface is the plane, and exact results are trivially obtained when calculating curvature on the plane.

The next two examples of minimal surfaces are catenoids and helicoids, as in Figure 3.3, which are locally isometric surfaces. Both have relatively simple parameterizations. The catenoid is given by the mapping

$$\begin{aligned} X(\xi,\eta) &= \cosh(\xi)\cos(\eta) \\ Y(\xi,\eta) &= \cosh(\xi)\sin(\eta) \\ Z(\xi,\eta) &= \eta \end{aligned} \tag{3.79}$$

and the helicoid by the mapping

$$\begin{aligned} X(\xi,\eta) &= \xi\cos(\eta) \\ Y(\xi,\eta) &= \xi\sin(\eta) \\ Z(\xi,\eta) &= \eta. \end{aligned} \tag{3.80}$$

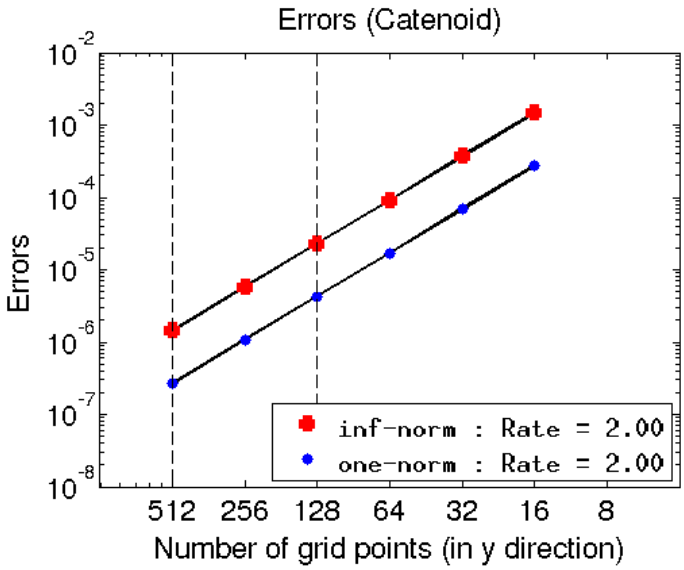

Figure 3.4: Convergence rate for curvature on the catenoid.

The scheme has second-order convergence on both of these surfaces, as seen in Figure 3.5 and Figure 3.4.

The next two minimal surfaces are the Enneper surface and the Henneberg surface. This scheme on the Henneberg surfaces fails to converge around the origin as the Henneberg surface does not have a consistent choice of surface normals around that point and is not continuous at that point. The Enneper surface is complete, with no singularities, and the scheme shows second-order convergence (Figure 3.7). The errors on the Enneper surface are centered on the area of the origin as seen in Figure 3.6. There are other implicit minimal surfaces we do not consider, since they do not have parametric representations.

3.3.3 Results on the Sphere

The sphere, and other genus 0 orientable surfaces, is not able to be tiled with quadrilaterals in a smooth fashion. The results of any scheme operating on the sphere

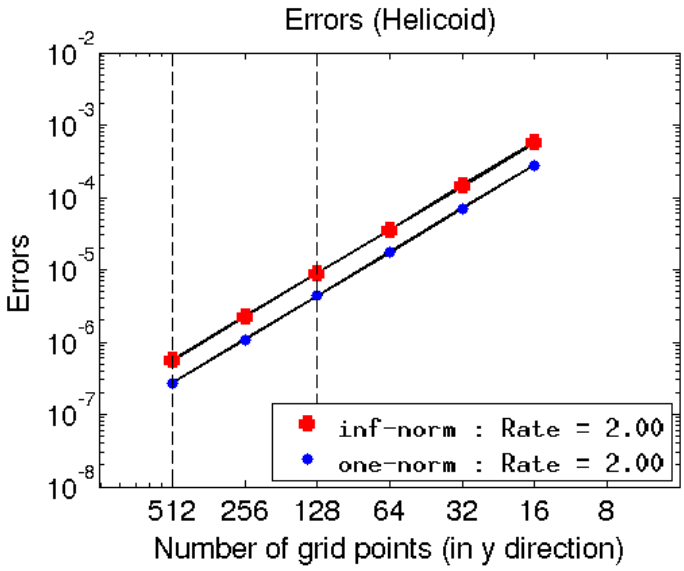

Figure 3.5: Convergence rate for curvature on the helicoid

will depend on which discretization is used. In the mapping obtained from spherical coordinates, there are discontinuities at the poles. There are other mappings that treat the discontinuity in a different manner.

In [2], Calhoun, Helzel, and LeVeque introduced the 'pillow grid mapping' for the sphere, which consists of two square logically Cartesian grids mapped to the northern and southern hemispheres of the sphere, respectively. Discontinuities or "seams" in this mapping occur along the equator and along the four diagonals leading from the equator to the poles. Unlike the spherical grid, the pillow grid has the property that the ratio of the area of the largest to smallest grid cells remains relatively fixed as the mesh is refined. This makes this grid attractive for numerical computations, which are restricted by the smallest mesh cells. The grid and the error can be seen in Figure 3.8. The curvature calculation of this scheme on the pillow mapping of the sphere gives first-order accurate results in the infinity norm, demonstrating that we obtain convergence even along cells that lie along the seams of the mapping. The scheme

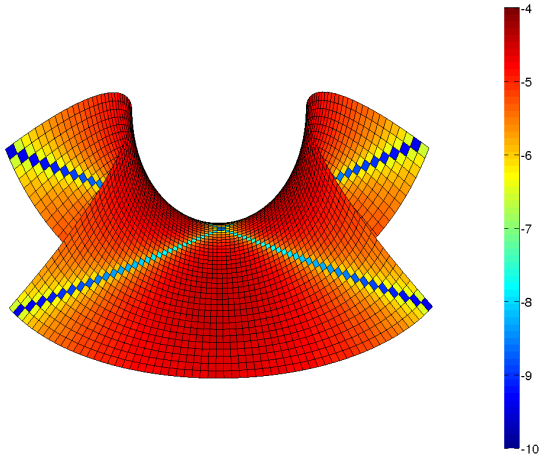

Figure 3.6: The portion of the Enneper surface on which errors in the curvature calculation occur. The errors vanish outside of the area shown and the self intersection of this complete surface hides the errors from view if a larger area is viewed. The color scale is the same log scale used in Figure 3.3

is second-order accurate in the one-norm. Figure 3.9 shows the convergence in the pillow grid sphere.

For spherical coordinates, second-order accurate results were obtained when the poles were excluded as seen in Figure 3.10. The overall errors were greater on the spherical coordinates over the pillow grid as seen by comparing the coloring of Figure 3.8 and Figure 3.11.

As shown above, the circle in one dimension was exact yet Figure 3.9 clearly shows that these results do not fully extend to the sphere.

3.4 Computational Performance

Here we consider the computational complexity of our numerical scheme. That is, we quantify the growth in the time it takes our scheme to run given an input size. The

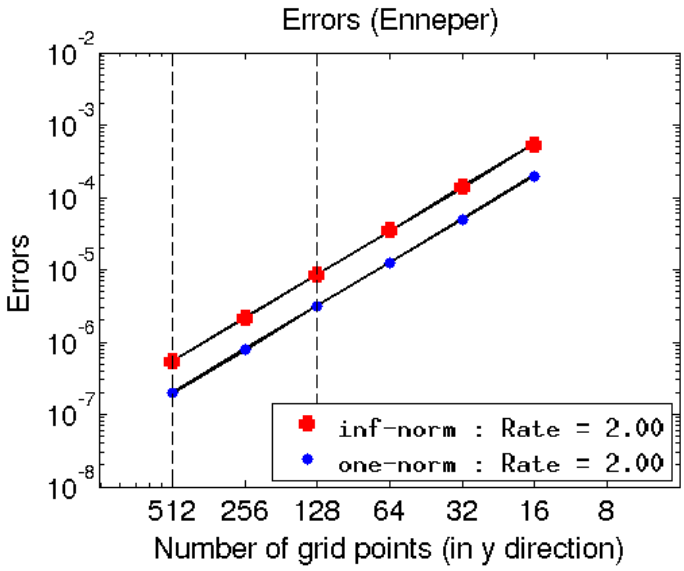

Figure 3.7: Convergence rate of the errors on the Enneper surface

upper limit on this growth is known as big O, and in cases where an algorithm runs in $O(n^2)$ it means that doubling the size of the input will roughly cause the algorithm to take four times as long, subject to constants and lower-order terms. For non-square matrix operations it is common to use m and n for the the two matrix directions, here though we assume that $m = n$ as the two directions are equivalent. This means that calculating a single value at each grid point will be $O(n^2)$. It is also possible to give the big Θ notation, which is the exact bound above and below. It is common to only list the big O notation as the upper bound is of primary interest and it can be assumed that it is the least upper bound known.

Due to the fact that we calculate curvature at every primal point, we must have $O(n^2)$ in time. Specifically, it is $197n^2 + C$ based on the counting of the floating point operations, where C is very small. Figure 3.12 shows this $O(n^2)$ time complexity by plotting timed runs and fitting a quadratic to these points.

Each curvature calculation only depends on the eight surrounding points, meaning

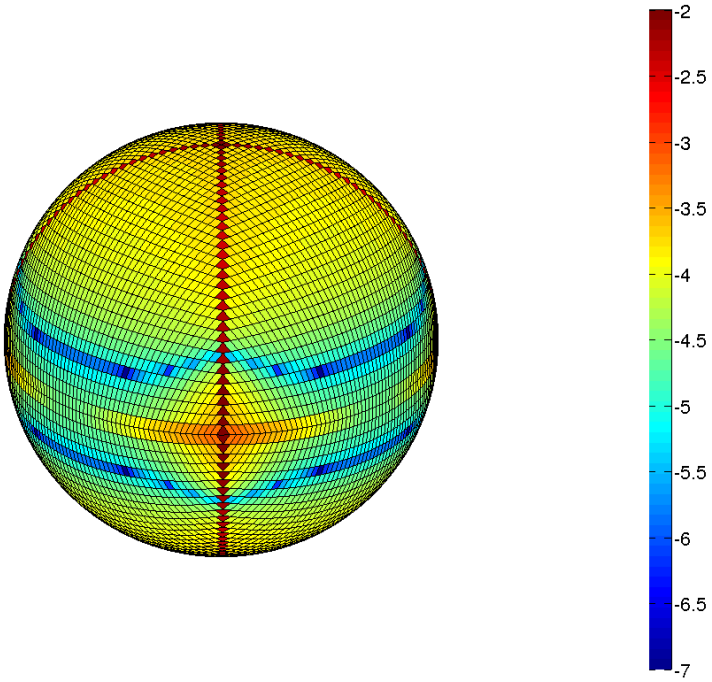

Figure 3.8: The pillow mapping of the sphere showing the log of the computed errors

the scheme is easy to parallelize with the use of ghost points. Any parallelization will add additional floating point operations, at the very least because of the ghost points. For the serialized code, it is easy to minimize calculations by sharing calculations between points, meaning that every edge between two points needs to be calculated only once. In the case of MPI, Message Passing Interface, it is best to minimize communication between nodes so some edges should be recalculated. In the case of calculating on a GPU, Graphics Processing Unit, it would be best to actually calculate the edge differences at each mesh point so as to maximize GPU performance and minimize memory accesses, which in the case of the GPU is the primary consideration.

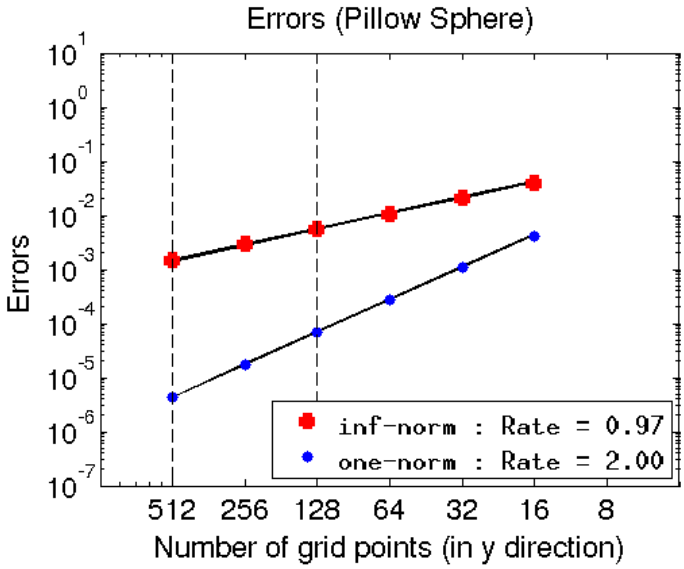

Figure 3.9: Convergence rate of the errors on the pillow sphere grid, note the first order convergence in the inf-norm along the seams

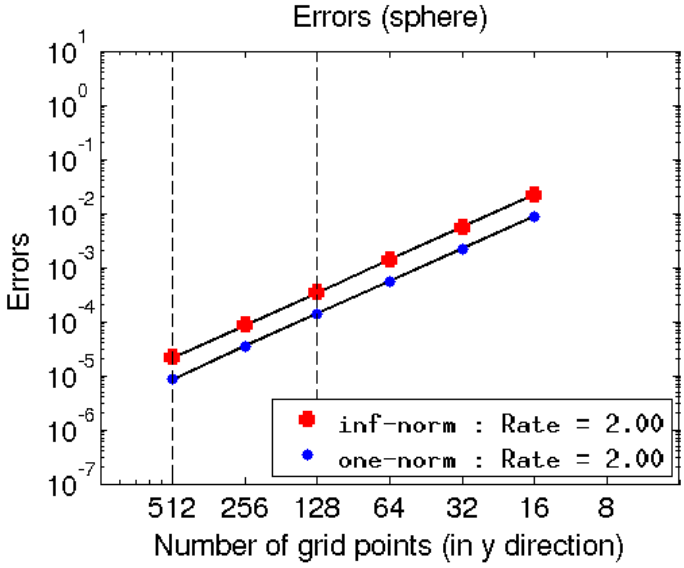

Figure 3.10: Convergence of the curvature errors using the spherical coordinate mapping for the sphere, excluding the poles

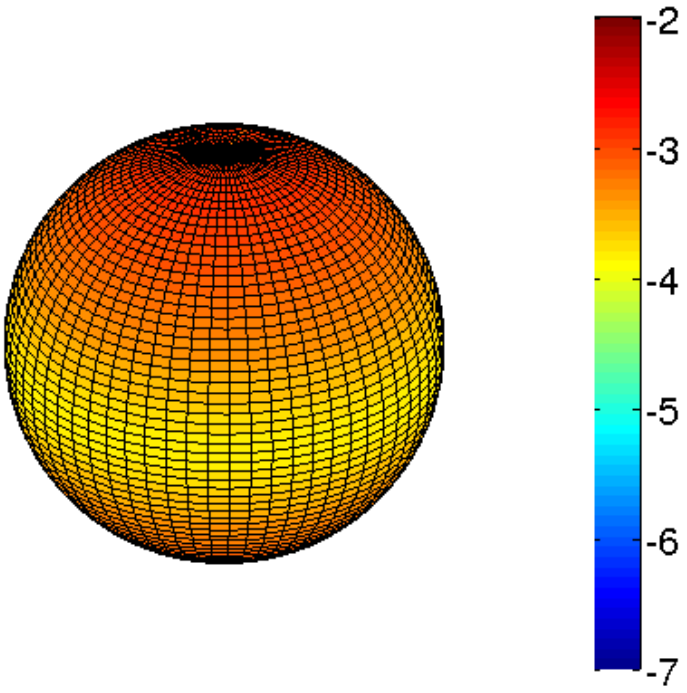

Figure 3.11: Curvature errors computed on the spherical coordinate mapping. The color map used here is the same as that used for the pillow grid. The color bar shows the log of the errors.

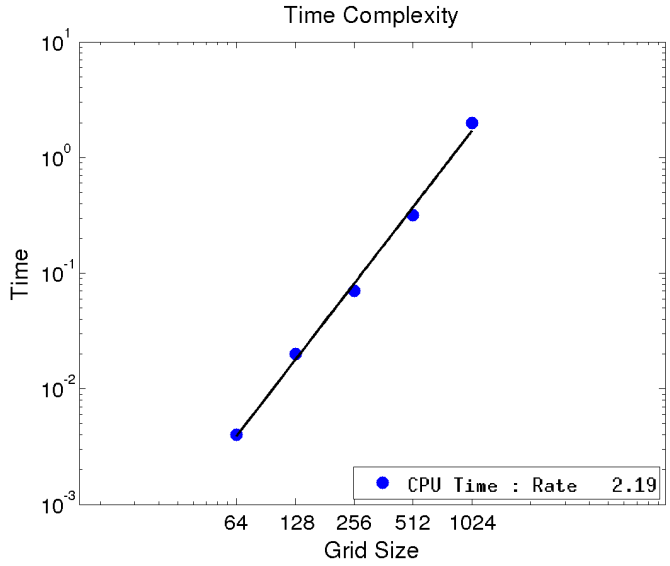

Figure 3.12: Graph showing the second-order time complexity of our scheme on a surface.

CHAPTER 4

COMPARISON WITH OTHER CURVATURE FORMULAS

We now compare our scheme with other schemes for computing curvature on quadrilateral and triangular surface meshes. Before doing so, we reformulate our scheme so that it can be used without reference to primal and dual points.

4.1 Our Formula Applied to Prescribed Quadrilateral Surface Meshes

Generally, quadrilateral meshes are not given as primal and dual points as we have presented, but as a single mesh grid. We would like to still be able to use our scheme and also to be able to calculate curvature at every mesh point. One obvious way of doing so would be to assume double the mesh spacing and create sets of primal and dual points. The problem with doing that is that on each calculation the nearest neighbors are not all being used and so accuracy is reduced. An alternative is to use the stencil in Figure 4.1, which allows for calculation at every mesh point and uses all of the nearest neighbors of a point. In comparing with other schemes this is the stencil used. This stencil does not assume any more information about the mesh grid than do any of the other formulas.

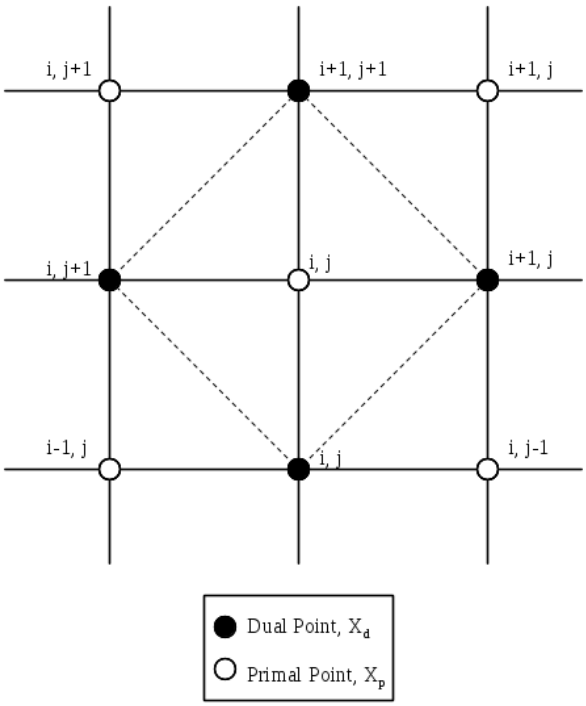

Figure 4.1: Diagram showing how we apply our scheme to a prescribed mesh

4.2 Cotan Formula

As defined in Meyer *et al.* [8], the cotan formula for mean curvature normal is given as

$$\mathbf{K}(x_i) = \frac{3}{2A_{mixed}} \sum (\cot \alpha_{i,j} + \cot \beta_{i,j})(\mathbf{X}_i - \mathbf{X}_j), \tag{4.1}$$

where α and β are defined as the opposing angles from the vector formed by $\mathbf{X}_i - \mathbf{X}_j$ and the cotangent of these are found by

$$\cot(\mathbf{u}, \mathbf{v}) = \frac{\mathbf{u} \cdot \mathbf{v}}{\sqrt{\|\mathbf{u}\|^2 \|\mathbf{v}\|^2 - (\mathbf{u} \cdot \mathbf{v})^2}}, \tag{4.2}$$

where $\mathbf{u}$ and $\mathbf{v}$ are the vectors defining the angle.

The area $2A_{mixed}$ in Meyer *et al.* [8] assumes that the triangles are acute, which with our meshing is not guaranteed. Since the triangles may be obtuse, then the area calculation reduces to being equivalent to the earlier formula by Desbrun *et al.* [4] given as

$$\frac{3}{4A}\sum(\cot\alpha_{i,j}+\cot\beta_{i,j})(\mathbf{X}_i-\mathbf{X}_j), \tag{4.3}$$

where A is the full 1-ring area of the triangles. Finding κ is a simple case of taking the norm of the vector formed from this formula. As shown by G. Xu [7] [9] these formulas do not converge in the general case, but do in special cases. It was also shown that they cannot converge in general [10]. Obviously, we are working on quadrilaterals and not triangles, meaning that we must subdivide the mesh in order to be able to use the "cotan" formula. There are a few possible ways of subdividing the quadilaterals: there is a valence 4 triangulation based on the dual points, the valence 8 triangulation, which uses all of the 1-ring points, and a valence 6 triangulation, which removes two of the primal points [7]. As noted in [7], and seen in Figure 4.2, only the valence 6 triangulation actually converges while the valence 8 fails to converge to the correct solution.

As proven earlier, our scheme returns the exact solution for the case of the cylinder and gets second-order accurate results for the sphere. The cotan formula with valence 6 does converge on the cylinder but does not return the exact result. For the case of the sphere, the cotan formula does not replicate our results, Figure 4.4 shows that the valence 6 fails to converge in the inf-norm and in Figure 4.3 it has only first order convergence in the one-norm.

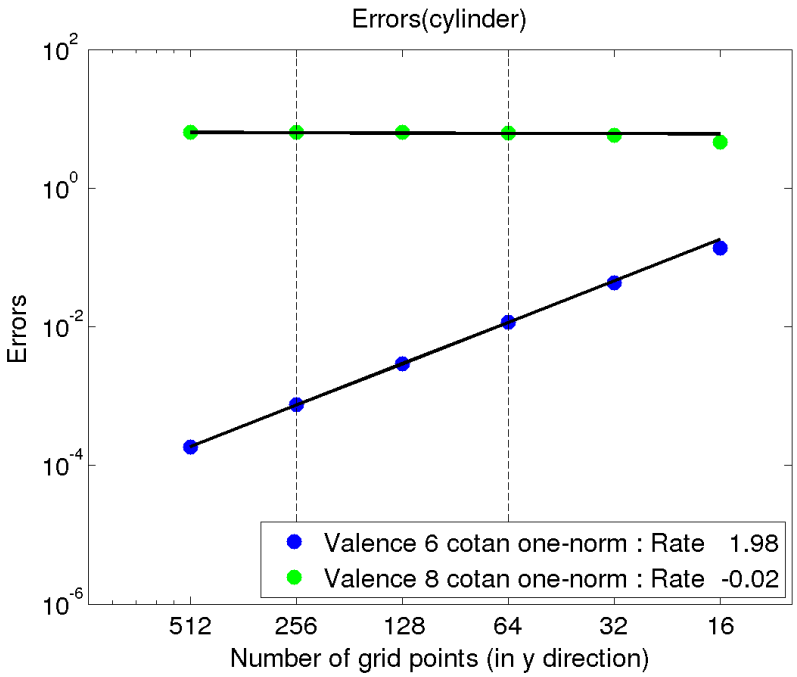

Figure 4.2: Convergence of the curvature errors computed on the cylinder for the cotan formula using both valence 6 and valence 8 rings. Our scheme is exact for the cylinder (see Section 3.3.1) and so is not shown. Inf-norm mirrors exactly the one-norm.

4.3 Liu's Formula

Here we consider and compare our scheme with that of D. Liu *et al.* [7]. The equations are derived and proven to be convergent on smooth parametric surface meshes, which are set up in a manner similar to what our scheme is derived on and for the one-point integration formula. It was shown in that paper that little to no additional accuracy is obtained by using higher-order integration formulas for the coefficients. The one-point formula from that paper is now presented. Given a quadrilateral made of points $\{\mathbf{p}_i, \mathbf{p}_j, \mathbf{p}'_j, \mathbf{p}_{j+1}\}$, we define two derivatives of a bilinear parametric surface that interpolates the four vertices as

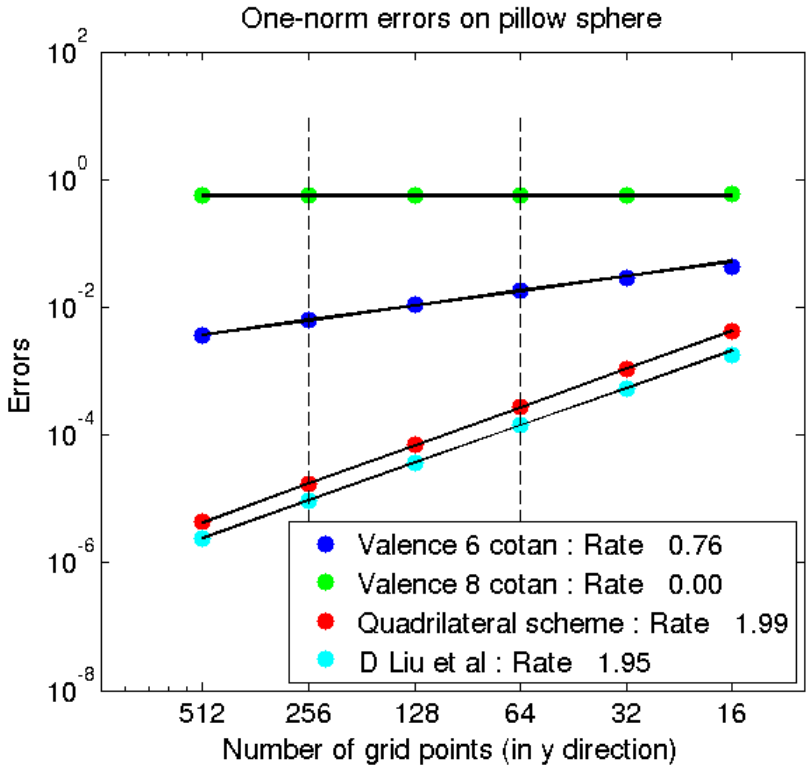

Figure 4.3: Comparison of one-norm convergence rates for the cotan formula, our quadrilateral scheme, and Liu's formula.

$$\begin{aligned} S_u &= \frac{1}{2}(\mathbf{p}_{j+1} - \mathbf{p}_i) + \frac{1}{2}(\mathbf{p}'_j - \mathbf{p}_j) \\ S_v &= \frac{1}{2}(\mathbf{p}_j - \mathbf{p}_i) + \frac{1}{2}(\mathbf{p}'_j - \mathbf{p}_{j+1}). \end{aligned} \tag{4.4}$$

We then use this to get the area of the quadrilateral

$$A_j = \sqrt{\|S_v\|^2\|S_u\|^2 - (S_u \cdot S_v)^2}. \tag{4.5}$$

The full area surrounding p_i is just the sum of the quadrilaterals surrounding that point. With the area, we can continue to get the coefficients, as in equation (2.10) of [7],

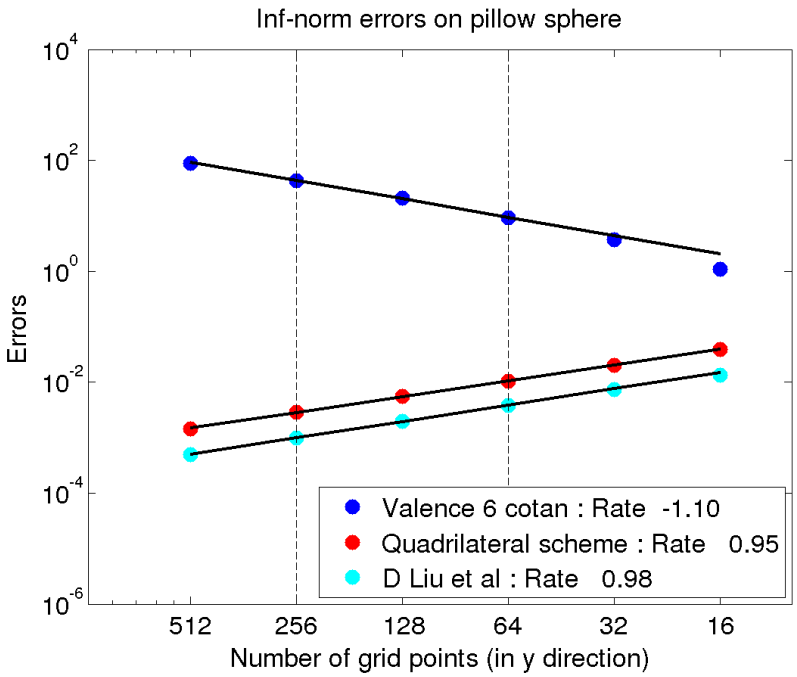

Figure 4.4: Comparison of inf-norm convergence rates for our quadrilateral scheme, the "cotan" formula, and D. Liu's formula.

$$\begin{aligned} \alpha_j &= \frac{\|S_u\|^2 - \|S_v\|^2}{4A_j} \\ \beta_{j+1} &= \frac{\|S_v\|^2 - \|S_u\|^2}{4A_j} \\ \gamma'_j &= \frac{\|S_u - S_v\|^2}{4A_j}. \end{aligned} \tag{4.6}$$

The mean curvature is given by equation (2.4) of [7] as

$$H(p_i) = \frac{2}{A(\mathbf{p}_i)} \sum_j [\alpha_j(\mathbf{p}_j - \mathbf{p}_i) + \beta_{j+1}(\mathbf{p}_{j+1} - \mathbf{p}_i) + \gamma'_j(\mathbf{p}'_j - \mathbf{p}_i)]. \tag{4.7}$$

As seen in both Figure 4.3 and Figure 4.4, the convergence rate of this scheme mirrors our own for the case of the pillow grid on the sphere. It is conceptually easier to understand what our scheme is doing and how one gets to the surface normal when

compared to that of Liu, which is less clear as to the approximations being done. Both schemes do use the concept of a bilinear parametric surface that interpolates the quadrilateral mesh. In our scheme, the surface is used around the point in question and approximates just the area while Liu's scheme patches the surface around the point in question and does all calculations on the patches.

CHAPTER 5

CONCLUSION

We have constructed a finite-volume scheme for calculating curvature and surface normals along a curve and shown this scheme to give the exact result for the line and the circle. We then demonstrated it to be second-order accurate numerically along the ellipse. Then, it was modified slightly to more general curves allowing for the additional calculation of the torsion and the binormals.

An extension of the one-dimensional scheme was created for surfaces using quadrilateral meshes. This computational scheme was proven as exact for the case of the cylinder and to be second-order accurate in the one-norm for two particular sphere grids. Second-order accurate results were also seen on other surfaces, such as minimal surfaces.

We compared our scheme to the "cotan" formula. This comparison showed that our scheme is an improvement upon using the "cotan" formula on quadrilateral meshes.

The scheme was shown to be numerically similar to that of Liu *et al.* Given a complete meshing our scheme requires fewer calculations than Liu's scheme. In the listing of the faces, which is common in computer graphics applications, Liu's scheme does not require the assembling of the complete one-ring neighborhood and works with mixed quadrilaterals and triangles. This makes Liu's scheme more attractive

in the processing of graphical surfaces. Neither scheme has been proven to converge over such meshes and with additional design work it is possible to obtain results with our scheme on listings of faces.

Future work should include a method for dealing with finite boundaries, which have not been discussed, but which are an important issue.

REFERENCES

[1] M. Alazah, S. N. Chandler-Wilde, and S. La Porte. Computing Fresnel Integrals via Modified Trapezium Rules. *ArXiv e-prints*, September 2012.

[2] D. A. Calhoun, C. Helzel, and R. J. LeVeque. Logically rectangular grids and finite volume methods for PDEs in circular and spherical domains. *SIAM Review*, 50(4):723–752, 2008.

[3] Donna A. Calhoun and Christiane Helzel. A finite volume method for solving parabolic equations on logically cartesian curved surface meshes. *SIAM J. Sci. Comput.*, 31(6):4066–4099, 2009.

[4] M. Desbrun, M. Meyer, P. Schröder, and A. Barr. Implicit fairing of irregular meshes using diffusion and curvature flow. In *SIGGRAPH*, pages 317–324. ACM, 1999.

[5] Manfredo P. do Carmo. *Differential Geometry of Curves and Surfaces*. Prentice-Hall, Englewood Cliffs, NJ, 1976.

[6] F. Kälberer, M. Nieser, and K Polthier. Quadcover - surface parameterization using branched coverings. *Computer Graphics Forum*, 26:375–384., 2007.

[7] D. Liu, G. Xu, and Q. Zhang. A discrete scheme of Laplace-Beltrami operator and its convergence over quadrilateral meshes. *Computers and Mathematics with Applications*, 55:1081–1093, 2008.

[8] M. Meyer, M. Desbrun, P. Schröder, and A.H. Barr. Discrete differential geometry operators for triangulated 2-manifolds. International workshop on visualization and mathematics, Berlin, Germany 2002.

[9] G. Xu. Discrete Laplace-Beltrami operators and their convergence. *Computer Aided Geometric Design*, 21(8):767–784, 2004.

[10] Zhiqiang Xu, Guoliang Xu, and Jia-Guang Sun. Convergence analysis of discrete differential geometry operators over surfaces. *11th IMA international conference on Mathematics of Surfaces*, pages 448–457, 2005.