ISSN 1213-6972 (print)
ISSN 1213-6964 (online)
Journal of WSCG
http://www.wscg.eu
Vol.29, No-1-2, 2021

New Methods and Novel Framework for Hypersurface Curvature Determination and Analysis

Jacob D. Hauenstein
The University of Alabama in Huntsville
301 Sparkman Drive
Huntsville, AL 35899
jacob.hauenstein@uah.edu

Timothy S. Newman
The University of Alabama in Huntsville
301 Sparkman Drive
Huntsville, AL 35899
newmant@uah.edu

ABSTRACT

New methods for hypersurface (that is, 3-dimensional manifold) curvature determination in volumetric data are introduced. One method is convolution-based. Another method is spline-based. Method accuracy is also analyzed, with that analysis involving comparison of the methods with each other as well as against two existing convolution-based methods. The accuracy analysis utilizes a novel framework that enables curvature determination method accuracy analysis via dynamically generated synthetic test datasets formed from continuous trivariate functions. Such functions enable accuracy analysis versus ground truth. The framework is also described here.

Keywords

Curvature, Volumetric Data, 3D Manifold, Hypersurface, Imaging

1 INTRODUCTION

Curvature, a translation- and rotation-invariant descriptor, describes the shape of the underlying data (e.g., a continuous function or discrete dataset) at a given point within the data. It is commonly used for many tasks in areas including quality control, healthcare, visualization, computer graphics, etc. Because curvature is a shape descriptor, uses in such areas often exploit curvature for model analysis [Hul12], object recognition, and general shape-based analysis tasks or tasks that benefit from its translation- and rotation-invariant shape descriptive capabilities, e.g., [Wan16, Lef18, Bib16, Bes86, Bel12, Sou16]. Curvature's use also extends to less obvious shape-based tasks, such as in initializing streamlines in vector field visualization [Wei02] and image denoising [Myl15], as well as in improving the accuracy of neural networks used for object detection [Wan16], surface reconstruction [Lef17], biometrics [Sya17], etc.

While it is possible to compute and utilize curvature within 3-dimensional (3D) manifolds, many tasks, including the ones just described, instead compute curvature within surfaces (2-dimensional manifolds), even when the underlying data from which curvature is measured is 3D (e.g., volumetric data). Methods to find curvatures on surface intersection curves also exist, with new methods recently reported for intersection curves of surfaces of high dimension [Özc21]. Other recent work considering curvature of 3D manifolds (also known as *hypersurfaces* [Mon92]) has shown promise for shape-descriptive tasks within volumetric data, especially in healthcare [Suz18, Hir18] and seismological applications [Ald14].

Varied measures of surface and hypersurface curvatures exist. The *principal curvatures* are among the most commonly used, including in aforementioned tasks. In the case of surfaces, there are two principal curvatures (denoted κ_1 and κ_2), and, for each point on the surface, these curvatures represent the minimum and maximum curvature at the point (ordered such that $\kappa_1 > \kappa_2$). In contrast, for hypersurfaces, there are three principal curvatures (denoted κ_1, κ_2, and κ_3) for each point on the hypersurface (ordered such that $\kappa_1 > \kappa_2 > \kappa_3$).

While many curvature tasks utilize the principal curvatures, these curvatures may not be computed exactly, because the data on which curvature is to be computed is real, sensed data (such as from a medical scanner). As a second derivative quantity, exact computation of the principal curvatures requires knowledge of the underlying continuous form of the surface or hypersurface to compute the necessary derivatives. In the case of real, sensed data, the continuous form is typically unknown and only discrete data is available; the principal curvatures must then be determined by other means (e.g., by estimating the necessary derivative quantities from the discrete data). Determination of curvature from discrete data is often inexact. Different determination methods may produce differing levels of accuracy on a given data

set, as recently noted in two studies of surface curvature determination methods [Hau20, Kro19].

Our focus in this work is to determine hypersurface principal curvatures from discrete volumetric datasets (henceforth simply *volumes*). We describe two new methods for determination of such principal curvatures. We also describe a framework that can be used to dynamically generate volumes, along with ground truth curvature values, to evaluate and compare the accuracy of each method's curvature determination. Furthermore, we use this framework to evaluate the accuracy of our two new methods for principal curvature determination (as well as two existing methods).

In the paper, Section 2 provides background on the mathematics of hypersurface curvature and related methods for determining hypersurface and surface curvature in volumes. Section 3 discusses our new methods for determining hypersurface curvature in volumes. Section 4 describes our testing and evaluation framework. Section 5 describes the volumes we generated using the framework. Section 6 details results obtained from these volumes. Section 7 concludes the work.

2 BACKGROUND / PREVIOUS WORK

First, we provide a brief overview of the mathematics for determining curvature of hypersurfaces within continuous volumetric data. The presentation here closely follows the formulations in both [Mon92] and [Ham94]. Next, we briefly describe the two existing methods for determining hypersurface curvature in volumes considered in our studies presented later as well as an existing study of the two methods. Finally, we present some details on methods for determining conventional surface curvature in volumes, as our two new hypersurface curvature determination methods, presented later, are inspired by these methods.

The mathematics notation we use is as follows. A grid (or sample) point within the volume is denoted as (x,y,z), where $0 \leq x < N_x$, $0 \leq y < N_y$, $0 \leq z < N_z$ for a volume of size $N_x \times N_y \times N_z$. The value at point (x,y,z) is denoted $f(x,y,z)$; f represents the underlying continuous function that generates the discrete volume. f_x represents the partial derivative of f in the x direction.

2.1 Hypersurface Curvature Mathematics

When the continuous form of the function f that generates the volume is known, the three principal curvatures of the hypersurface can be computed at each point within the volume fairly easily. They are simply, at each point in the volume, the eigenvalues of the matrix:

$$\frac{1}{l}\begin{bmatrix} f_{xx} & f_{xy} & f_{xz} \\ f_{xy} & f_{yy} & f_{yz} \\ f_{xz} & f_{yz} & f_{zz} \end{bmatrix} \begin{bmatrix} 1+f_x^2 & f_x f_y & f_x f_z \\ f_x f_y & 1+f_y^2 & f_y f_z \\ f_x f_z & f_y f_z & 1+f_z^2 \end{bmatrix}^{-1}, \tag{1}$$

where

$$l = \sqrt{1+f_x^2+f_y^2+f_z^2}. \tag{2}$$

We denote these eigenvalues λ_1, λ_2, and λ_3. As mentioned previously, computation of κ_1, κ_2, and κ_3 at any point requires knowledge of the first and second derivatives of f at the point, and, for many curvature-based tasks, the continuous form of f is unknown because the data is not continuous. Consequently, the first and second derivatives often must be estimated to determine κ_1, κ_2, and κ_3. All of the hypersurface curvature determination methods discussed in this work operate by estimating the necessary derivatives using varying approaches and then using those estimated derivatives to evaluate Eq. (1) (i.e., compute λ_1, λ_2, and λ_3).

2.2 Existing Hypersurface Curvature Methods and Comparative Studies

In our studies we compare our new methods with two existing methods for determining hypersurface curvature in volumes. First, we describe these two existing methods. The first, denoted **TEF**, uses convolution with kernels based on the Taylor Expansion in order to estimate derivatives (from which curvature is then computed). The second, denoted as **OPF**, uses convolution with a set of orthogonal polynomials in order to estimate derivatives.

2.2.1 TEF

The **TEF** method uses convolution filters derived from the Taylor Expansion. It determines hypersurface curvatures by first estimating all derivatives necessary for computing hypersurface curvature. These estimates are made through doing a series of convolutions along each axis of the volume. Once these convolutions are completed, the three principal curvatures are computed as λ_1, λ_2, and λ_3 using the estimated derivatives.

The convolution filters used by **TEF** were developed using a framework described by Möller et al. [Möl98]. The same framework was also previously used in a surface curvature method for volumes [Kin03].

For the **TEF** method, convolution filters are sampled from the Taylor Expansion using specified accuracy and continuity parameters. Previous studies of both hypersurface curvature and surface curvature methods that utilize the Möller et al. framework have used filters with C^3 continuity and fourth order accuracy [Hau19, Hau20], and we follow that usage in our own studies. When no noise or other errors are present, such filters allow for exact reconstruction of functions of degree 3 or lower.

When configured as described, the **TEF** first derivative filter is:

$$\{-\frac{1}{12}, \frac{2}{3}, 0, -\frac{2}{3}, \frac{1}{12}\}, \tag{3}$$

and the second derivative filter is:

$$\{-\frac{1}{24},\frac{1}{6},\frac{17}{24},-\frac{5}{3},\frac{17}{24},\frac{1}{6},-\frac{1}{24}\}. \quad (4)$$

2.2.2 OPF

The **OPF** method uses convolution filters derived from orthogonal polynomials to estimate all the necessary derivatives for determination of hypersurface curvature. Convolution with such filters is equivalent to a least squares fitting, and it thus gives identical results to that of a linear regression-based approach. Like the **TEF** approach, the **OPF** approach uses these derivative estimates to compute the three principal curvatures at each point in the volume as λ_1, λ_2, and λ_3 of Eq. (1).

OPF uses convolution filters of size N, where N is an odd number. The filters are generated by sampling orthogonal polynomials, denoted b_0, b_1, b_2, at N sample locations. b_0, b_1, and b_2 are given by:

$$b_0(\theta) = \frac{1}{N}, \quad (5)$$

$$b_1(\theta) = \frac{3}{M(M+1)(2M+1)}\theta, \quad (6)$$

$$b_2(\theta) = \frac{1}{P(M)}(\theta^2 - \frac{M(M+1)}{3}), \quad (7)$$

where $M = \frac{N-1}{2}$ and $P(M)$ is given by:

$$P(M) = \frac{8}{45}M^5 + \frac{4}{9}M^4 + \frac{2}{9}M^3 - \frac{1}{9}M^2 - \frac{1}{15}M. \quad (8)$$

θ denotes the locations at which each polynomial is sampled, where $\theta \in \{-\frac{N-1}{2},...,-1,0,1,...,\frac{N-1}{2}\}$. The b_0 filter smooths; b_1 estimates the first derivative; and b_2 estimates the second derivative.

Previous works have recommended $N = 7$ [Hau20], but, in our studies, we test varying values of N including $N = 3, N = 5, N = 7$, and $N = 9$. When considering **OPF** at a specific N value, we will use the notation **OPF***N*, where N is one of 3, 5, 7, or 9.

2.2.3 Comparative Studies

We are aware of only one existing study that comparatively evaluated methods for determination of hypersurface curvature in volumes. In that study [Hau19], two hypersurface curvature methods were compared. That study compared performance of the **TEF** and **OPF** methods and concluded that (1) **TEF** exhibits relatively low error (i.e., high accuracy) in curvature when no noise is present in the data, and (2) **OPF** generally is more accurate in the presence of noise.

2.3 Related Methods for Surface Curvature in Volumes

The two new methods for determining hypersurface curvature later described in this paper are analogous to two existing methods for determining surface curvature in volumes. Here, we describe those two existing surface curvature methods. Both of these methods determine surface curvature by first estimating derivatives at every point in the volume and then computing the curvature at every point in the volume using the estimated derivatives. The first described existing method uses B-Spline fitting to estimate surface curvatures. The second described method uses convolution with Deriche filters.

2.3.1 B-Spline

Soldea et al. [Sol06] described a method for determining surface curvature in volumes using B-Spline fitting. The approach used by Soldea et al. directly uses the volume data as the coefficients for a trivariate B-Spline. Since the trivariate B-Spline formed from the data is a continuous function that approximates the volume, the necessary derivatives of the B-Spline can be computed at each point, and these derivatives are used to compute curvature.

2.3.2 Deriche Filters

Monga et al. [Mon92] described a method for determining surface curvature in volumes using convolution with 3D Deriche filters. Such filters are of special interest because they are formulated such that they can be implemented as infinite impulse response (IIR) filters, making them conducive to implementation on some hardware [Der90]. The Monga et al. strategy uses three filters. One filter is used to smooth, one filter is used to estimate the first derivative, and one filter is used to to estimate the second derivative. From these estimated derivatives, curvature is determined.

3 NOVEL METHODS FOR HYPERSURFACE CURVATURE DETERMINATION

Here, we describe the two new methods we introduce here for determining hypersurface curvature in volumes.

3.1 B-Splines (BS)

Our first new method for determining hypersurface curvature in volumes, denoted **BS**, forms a B-Spline that provides a continuous form approximation of the input volume f. It is analogous to the approach used by Soldea et al. [Sol06] for determining surface curvature in volumes. The method determines curvature by first forming a continuous trivariate B-Spline, denoted $\hat{f}$, from the discrete volume. $\hat{f}$ is then used to compute derivatives from which curvature is calculated, and $\hat{f}$ uses a volume of form:

$$\hat{f}(x,y,z) = \quad (9)$$
$$\sum_{i=0}^{N_x}\sum_{j=0}^{N_y}\sum_{k=0}^{N_z} f(i,j,k)B_{i,s_x,\tau_x}(x)B_{j,s_y,\tau_y}(y)B_{k,s_z,\tau_z}(z),$$

where each $B_{l,s,\tau}$ term denotes the l-th B-Spline blending function of degree s over a knot sequence τ. Since $\hat{f}$ is continuous, its first and second derivatives can be calculated at each point, and they are used to determine curvature (as λ_1, λ_2, and λ_3) at each point.

In our experiments on the **BS** method, reported later, τ is a uniformly spaced knot vector, and we test several values of s, including $s = 2$, $s = 3$, $s = 4$, and $s = 5$. To our knowledge, these experiments are the first reports on the use of splines of higher than cubic degree in any variety of curvature determination. When considering **BS** at a specific s value, we will use the notation **BS***s*, where s is one of 2, 3, 4, or 5.

3.2 Deriche Filters (DF)

Our second new method for determining hypersurface curvature in volumes, denoted **DF**, uses convolution with the Deriche filters. It is analogous to the approach used by Monga et al. [Mon92] for determining surface curvature in volumes. The method determines curvature by first convolving with the three Deriche filters to estimate derivatives from which curvature is calculated. We denote the three filters as $\hat{f}_0$, $\hat{f}_1$, and $\hat{f}_2$. The filters are defined as continuous functions, and they are sampled to produce the parameters for the convolution with the volume. Convolution with the sampling of $\hat{f}_0$ provides smoothing, with the sampling of $\hat{f}_1$ provides estimates for the first derivative, and with the sampling of $\hat{f}_2$ provides estimates for the second derivative. Convolution is always applied along all three axes at each point, with the directional derivatives estimated using the relevant derivative estimation filters along any axis on which derivatives are to be estimated. In this convolution, the smoothing filter is applied along all other axes (i.e., other than the axis for the directional derivative). $\hat{f}_0$, $\hat{f}_1$, and $\hat{f}_2$ are defined as [Mon92]:

$$\hat{f}_0(t) = c_0(1+\alpha|t|)e^{-\alpha|t|}, \tag{10}$$

$$\hat{f}_1(t) = -c_1 t\alpha^2 e^{-\alpha|t|}, \tag{11}$$

$$\hat{f}_2(t) = c_2(1-c_3\alpha|t|)e^{-\alpha|t|}, \tag{12}$$

where c_0, c_1, c_2, and c_3 are defined by

$$c_0 = \frac{(1-e^{-\alpha})^2}{1+2e^{-\alpha}\alpha-e^{-2\alpha}}, \tag{13}$$

$$c_1 = \frac{-(1-e^{-\alpha^3})}{2\alpha^2 e^{-\alpha}(1+e^{-\alpha})}, \tag{14}$$

$$c_2 = \frac{-2(1-e^{-\alpha 4})}{1+2e^{-\alpha}-2e^{-3\alpha}-e^{-4\alpha}}, \text{ and} \tag{15}$$

$$c_3 = \frac{(1-e^{-2\alpha})}{2\alpha e^{-\alpha}}. \tag{16}$$

The α term controls the level of smoothing when estimating derivatives, with smaller values providing more smoothing. The optimal value for it depends on a number of factors, including the amount and type of noise in the volume. In our experiments on the **DF** method, reported later, we test a variety of α values, including $\alpha = 1.0$, $\alpha = 3.0$, $\alpha = 5.0$, and $\alpha = 10.0$. When considering **DF** at a specific α value, we will use the notation **DF**α, where α is one of 1.0, 3.0, 5.0, or 10.0.

4 VOLUME GENERATION AND EVALUATION FRAMEWORK

This section describes our framework for generating volumes and evaluating curvature determination method accuracy. This framework is used to generate volumes (including application of rotations, shifts, and scalings) from continuous form trivariate expressions, compute the ground truth curvatures in such volumes, and evaluate the accuracy of the hypersurface curvatures determined by each determination method on the generated volumes.

4.1 Overview

Our testing and evaluation framework enables rapid evaluation of hypersurface curvature determination method accuracy. It is capable of dynamically generating volumes of a user-specified number of samples directly from continuous form trivariate expressions. These volumes can be generated at arbitrary, user specified rotations, shifts, and scales. Because these volumes are dynamically generated at run-time, the framework allows for rapid testing of accuracy on many volumes. This characteristic makes the framework particularly useful for generation of many closely related volumes, such as generation of many different rotations of volumes from the same continuous expression, e.g., in order to evaluate the sensitivity of hypersurface curvature determination methods to changes in rotation. Because all volumes, regardless of rotation, shift, or scale, are generated directly from the continuous form expression, ground truth values are always available to use for accuracy evaluation of all determination methods.

The framework maintains a list of all curvature determination methods for which accuracy is to be evaluated, including any relevant parameter settings for each method. Each combination of determination method and parameters defines a *determination method set*. Because a single determination method can be included on this list at differing parameter settings (i.e., in different determination method sets), the framework makes it possible to rapidly test curvature determination methods at a variety of possible parameter settings.

Upon generation of a volume, the framework runs each hypersurface curvature determination method set in the list on the volume, collects the determined curvatures from each method, and computes the absolute and relative error for each point of interest within the volume.

The framework thus enables rapid generation of volumes and rapid testing of curvature determination methods.

4.2 Generation of Test Volumes

To generate a volume used for testing a list of determination method sets via our testing framework, a number of characteristics that describe the volume to be generated must be specified (i.e., input). We describe here those characteristics and how the volume is generated.

The number of samples in each of the x, y, and z directions for the generated volume, N_x, N_y, and N_z, must be specified. For our experiments, described later, we have set $N_x = N_y = N_z = 32$.

The scale and shift parameters must be specified for each axis. At a shift of 0.0 and scale of 1.0 on each axis, the volume will be generated by sampling the continuous form expression f at unit spacing along each axis, resulting in samples at locations ranging from 0 to $N_x - 1$ along the x-axis, from 0 to $N_y - 1$ along the y-axis, and from 0 to $N_z - 1$ along the z-axis. That is, the sample at location (x_s,y_s,z_s) within the discrete volume will contain the value $f(x_s, y_s, z_s)$. In contrast, using the same shift but a scaling of 0.5 on each axis, the volume will be generated by sampling the continuous form expression at a spacing of 0.5 along each axis starting from 0. That is, the sample at location (x_s,y_s,z_s) within the discrete volume will contain the value $f(0.5x_s, 0.5y_s, 0.5z_s)$. If a shift of -1 is also applied on each axis, the volume will be generated by sampling the continuous form starting at -1 on each axis. That is, the sample at location (x_s,y_s,z_s) within the discrete volume will contain the value $f(0.5x_s - 1.0, 0.5y_s - 1.0, 0.5z_s - 1.0)$. Thus, the scaling parameter selects the spacing used for sampling, and the shift parameter selects the region where the sampling takes place.

The continuous form expression f must be specified. It is specified using a standard C-style mathematical notation understood by the *libmatheval* library [Fre14], but with placeholder variables *XFIX*, *YFIX*, and *ZFIX* used in place of the x, y, and z variables in the expression. These placeholder variables are replaced by the framework with appropriately scaled versions of the x, y, and z variables using a string find-and-replace within the expression. For example, if all scales along each axis are set at 0.5, the continuous form expression specified as $XFIX + YFIX + ZFIX$ would be replaced with $(0.5{*}x) + (0.5{*}y) + (0.5{*}z)$. In general, *XFIX* is replaced with $(\rho_x{*}x)$, where ρ_x is the scaling along the x-axis. *YFIX* and *ZFIX* are replaced analogously.

The rotation parameters must also be specified. By default, no rotation is applied, and sampling of the continuous expression is done exactly along the x-, y-, and z-axis as described in the prior two paragraphs. However, the framework allows for arbitrary rotation through the origin by specifying the angle of rotation and the axis of rotation. When a rotation is specified, the framework generates a rotation matrix describing the requested rotation, and the rotation is applied by multiplying each sample point's coordinates by the rotation matrix prior to sampling the continuous expression. For example, if a rotation angle of 90 degrees and a rotation axis of $(0,0,1)$ is specified, the continuous expression's x-axis will be mapped to the resulting volume's y-axis (e.g., location (0,1,0) in the volume will contain the value sampled via $f(1,0,0)$). When generating the volume, rotation is applied prior to shifting or scaling.

Additionally, the level of Gaussian noise must be configured. It is specified by standard deviation, and the standard deviation value is used to generate Gaussian noise of mean zero and specified standard deviation.

Finally, the volume is generated by sampling the replaced expression, resulting in a volume generated from the continuous expression at the appropriate scale and shift. Because the continuous form expression is known for the underlying volume, the framework is able to compute the ground truth curvature at each point in the volume (utilizing *libmatheval*'s symbolic derivative capabilities [Fre14]).

4.3 Evaluation of Accuracy of Test Volumes

For evaluation of accuracy, a list of *points of interest* (POI) within each test volume is also specified for the framework. These POIs specify which sample locations within the volume that the framework will include when calculating the accuracy of each determination method set. The use of POIs, as opposed to calculating accuracy using every sample point in the volume, allows for finer control over the accuracy evaluation process. For example, determined curvatures are often very inaccurate at volume edges due to insufficient support for convolution kernels. But, POIs can be selected such that the edges of the volume are not considered in accuracy evaluation. Each POI may optionally be used as the center of rotation when a rotation is applied. This results in the rotated volumes and POIs being equal in number, where each volume is rotated about a different POI. These volumes can then be used to evaluate the sensitivity of a method set to rotation, because, for each rotation, the value of curvature at the POI about which the volume is rotated should be unchanged. If the determined curvature value, or the error in determined curvature value, changes at the POI about which the volume is rotated, then the method set is sensitive to rotation.

Once the POIs are specified, the framework computes the absolute difference between the expected and actual curvature results at each POI. Using the results from each POI within the volume, the framework produces a number of aggregate results across all POIs in the volume for each determination method set. These aggregate

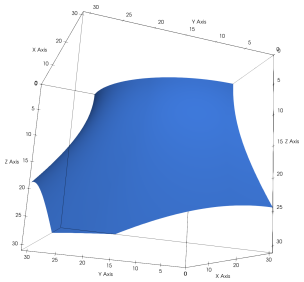


Figure 1: A rendering of a (2-D) isosurface of Genus3.

results include the arithmetic average of both the absolute errors and relative errors in each of κ_1, κ_2, and κ_3 for each determination method set on the volume.

5 GENERATED VOLUMES

Here, we describe the volumes generated by the framework and considered in our accuracy experiments. For all volumes, the framework is configured to generate volume datasets of size $32 \times 32 \times 32$, and the POIs selected are all the points within a $3 \times 3 \times 3$ neighborhood centered at the sample location $(15, 15, 15)$. This set of POIs ensures that none of the points used in accuracy evaluation are near the edge of the dataset.

The first volume is one generated from the quadratic polynomial expression:

$$x^2 + y^2 + z^2. \tag{17}$$

This expression has been used previously to generate volumes in other curvature studies (e.g., [Hau19, Hau20]). Because the 2D surfaces contained within the volume are spherical in shape, we will denote this expression (and the volumes it generates) *Spheres*. For volumes generated from this expression, we use a scaling of 1.0 and a shift of 0.0 on each axis. We consider the Spheres volumes at a variety of rotations (from -45 degrees to +45 degrees through the axis $(0, 0, 1)$), and both with and without noise. For each of these rotations, one rotated volume per POI is generated as described in Section 4.3.

The second volume is one generated from a higher degree polynomial expression:

$$(1 - (\frac{x}{6})^2 - (\frac{y}{3.5})^2)((x-3.9)^2 + y^2 - 1.44)$$
$$(x^2 + y^2 - 1.44)[(x+3.9)^2 + y^2 - 1.44] - 256z^2. \tag{18}$$

Because this expression exhibits a genus three 2D surface [Woo10], we will denote this expression (and the volumes it generates) *Genus3*. A rendering of a 2-D isosurface of Genus3 is shown in Fig. 1. For volumes generated from this expression, we use a scaling of $\frac{1}{64}$ and a shift of 0.0 on each axis. We consider the Genus3 volumes at a variety of rotations (from -45 degrees to +45 degrees through the axis $(0, 0, 1)$) and both with and without noise. For each of these rotations, one rotated volume per POI is generated as described in Section 4.3.

6 EXPERIMENTS AND RESULTS

In this section, we describe our experiments and results comparing the accuracy of our new **BS** and **DF** methods as well as the existing **TEF** and **OPS** methods. These results were all obtained using the framework described in Section 4, and the volumes generated for testing are as described in Section 5. We start with experiments on noise-free volumes.

6.1 Noise-Free Accuracy Results

First, we present an evaluation of how each method performs on the volumes when no noise is present. For these, the volumes were considered at 91 rotations (angles) (as described in Section 5), allowing simultaneous evaluation of the relative performance of each method on noise-free data and evaluation of how sensitive each method is to rotation (which should not change the curvature values or amount of error).

6.1.1 Spheres

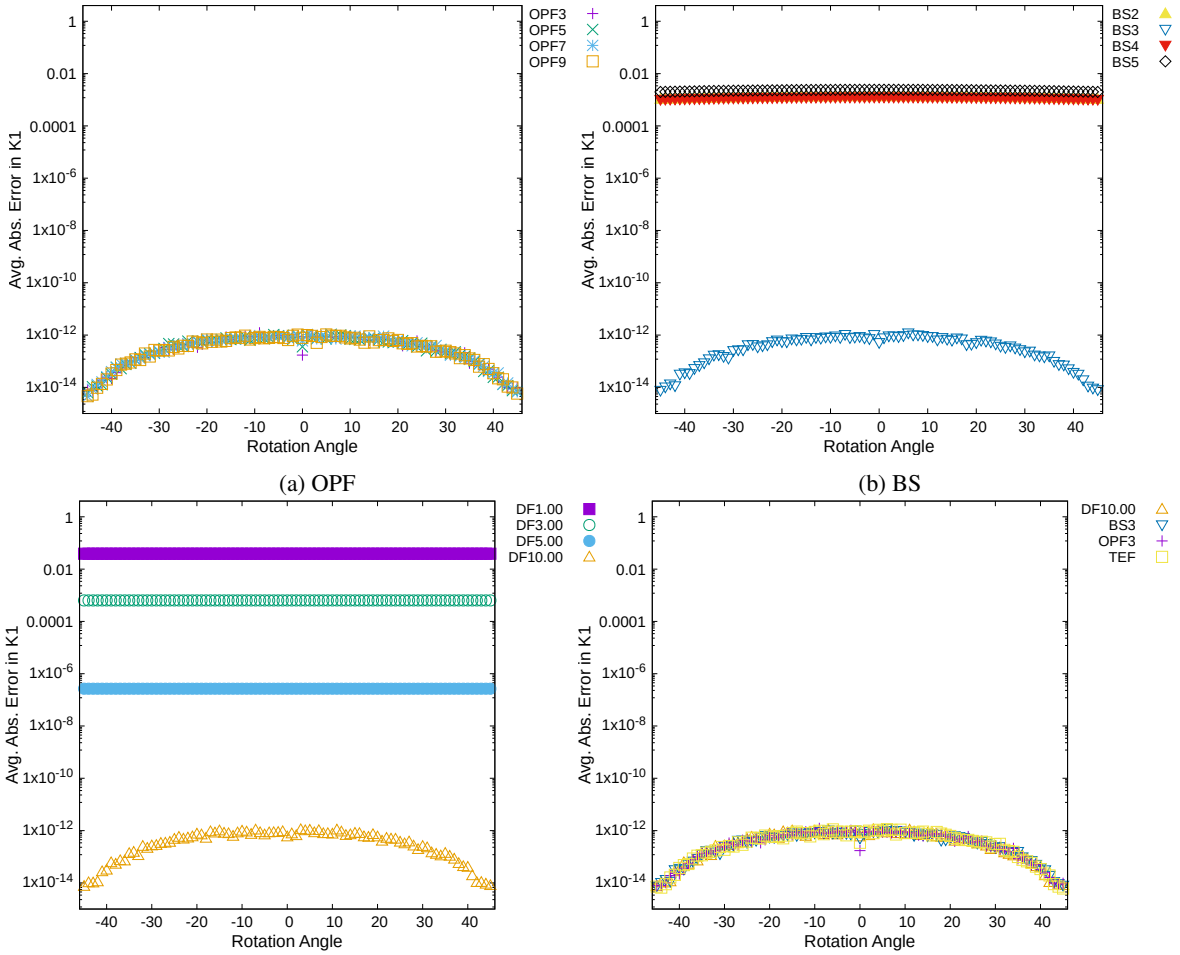
For the relatively simple Spheres volumes, most methods exhibit little error in determined curvatures. Fig. 2(a) shows the average absolute error in κ_1 for the **OPF** method at 91 rotations of Spheres volumes. While the amount of error in the determined curvatures does vary with the rotation, the amount of error at each rotation is extremely small and likely attributable to the limitations of floating point precision. Additionally, for these volumes, the filter size does not seem to have any appreciable impact on accuracy. This is likely because the hypersurface shapes within the dataset do not exhibit much variation over any of these filter sizes.

Fig. 2(b) shows the average absolute error in κ_1 for **BS** at the same 91 rotations. Here, **BS3** exhibits much lower error than the other degrees of **BS**. Like **OPF**, rotational sensitivity is low enough that it can be attributed to floating point precision.

Fig. 2(c) shows the average absolute error in κ_1 for the **DF** method at the same 91 rotations. Unlike **OPF**, here, accuracy strongly depends on the parameter settings. At lower α values, the error is much higher than at the higher α values, with $\alpha = 10.0$ exhibiting the smallest error. Only at $\alpha = 10.0$, where the errors are lowest, is it possible to see any change in error with rotation. Like in the **OPF** case, those changes are sufficiently small that they are likely explained by limitations of floating point precision.

Fig. 2(d) shows **OPF3**, **BS3**, **DF10.00** (the most accurate of each of **OPF**, **BS**, and **DF**) versus **TEF**. Here, the **BS3**, **OPF3**, **DF10.00**, and **TEF** methods all perform nearly identically and extremely accurately.

Due to space constraints, only plots of error in κ_1 are shown, but the results for κ_2 and κ_3 errors are very similar to these in relative accuracy of the methods and



(a) OPF (b) BS

(c) DF (d) OPF3, BS3, DF10.00, and TEF

Figure 2: Error levels in κ_1 on 91 different rotations of noise-free Spheres volumes

rotational sensitivity. For κ_2, the average absolute error across all the rotations is about 4.6×10^{-13} for **OPF** at all parameter settings, about 4.8×10^{-13} for **DF10.00**, about 4.6×10^{-13} for **BS3**, and about 4.8×10^{-13} for **TEF**. For κ_3, the average absolute error across all the rotations is nearly identical for **OPF** at all parameter settings, **DF10.00**, **BS3**, and **TEF**, with all of them having error of about 2.8×10^{-16}.

6.1.2 Genus3

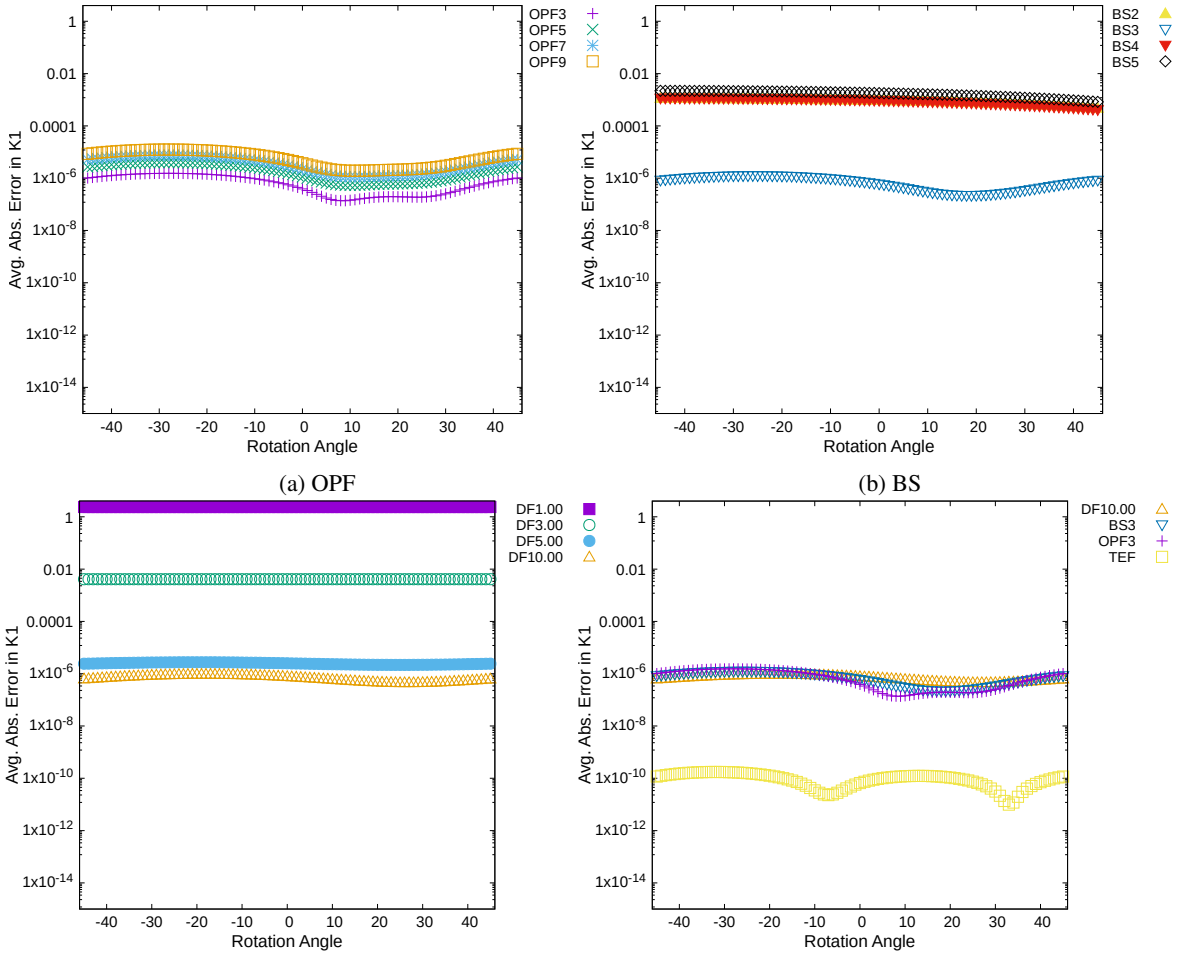
For Genus3 volumes, errors are notably higher than in the simpler Spheres volume. Fig. 3(a) shows the average absolute error in κ_1 for **OPF** at 91 rotations. Compared to Spheres, the change in error across rotation angles is much more pronounced. Whereas, in the case of Spheres, the changes in error with rotation were small enough that they could be explained by floating point limitations, the errors here are larger and likely indicate some amount of rotational sensitivity within **OPF**. Also unlike the Spheres, for these volumes, the filter size notably impacts the results, with **OPF3** exhibiting the least error.

Fig. 3(b) shows the average absolute error in κ_1 for **BS** at the same 91 rotations. Here, **BS3** again exhibits the lowest levels of error. **BS3** exhibits similar accuracy to **OPF3**, but it is more consistent in terms of exhibiting a similar error level across all rotations.

Fig. 3(c) shows the average absolute error in κ_1 for the **DF** method at the same 91 rotations. Again, accuracy strongly depends on the parameter settings for the **DF** method. $\alpha = 10.0$ again exhibits the smallest errors. **DF10.0** exhibits very small changes in error as rotation changes, suggesting that the **DF** methods are not particularly sensitive to rotation.

Fig. 3(d) shows **OPF3**, **BS3**, **DF10.00** (the most accurate of each of **OPF**, **BS**, and **DF**) versus **TEF**. Here, all methods exhibit low error, but **TEF** is much more accurate than the others.

Due to space constraints, only plots of error in κ_1 are shown, but the results for κ_2 and κ_3 errors are very similar to these in relative accuracy of the methods and rotational sensitivity. For κ_2, the average absolute error across all the rotations is about 7.5×10^{-7} for



(a) OPF (b) BS

(c) DF (d) OPF3, BS3, DF10.00, and TEF

Figure 3: Error levels in κ_1 on 91 different rotations of noise-free Genus3 volumes

OPF3, about 5.5×10^{-7} for **DF10.00**, about 6.5×10^{-7} for **BS3**, and about 1.9×10^{-11} for **TEF**. For κ_3, the average absolute error across all the rotations is about 1.2×10^{-7} for **OPF3**, about 1.0×10^{-7} for **DF10.00**, about 1.1×10^{-7} for **BS3**, and about 5.2×10^{-12} for **TEF**.

6.2 Noise-Added Accuracy Results

Next, we present an evaluation of how each method performs on the volumes when noise is added. For each of these studies, the same level of random (Gaussian) noise was added 30 times to both Spheres and Genus3, resulting in 30 volumes (or *trials*) of each with the same level of (different) random noise.

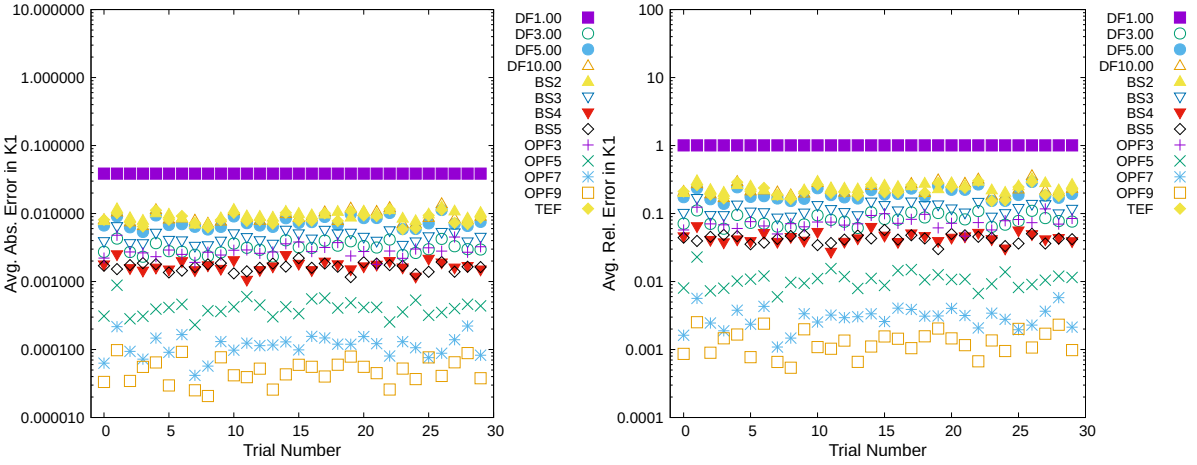
6.2.1 *Spheres*

Fig. 4(a) shows the average absolute error in κ_1 for the **OPF** method for 30 trials of noise with standard deviation of 0.25. In the presence of noise, **OPF** benefits from larger filter sizes, with **OPF9** exhibiting the lowest level of error in all trials, followed by **OPF7** and **OPF5**. The methods with the next lowest level of error are the B-Spline-based methods, with either **BS4** or **BS5** most accurate in most trials. This is a change from the noise-free cases, where **BS3** was consistently the most accurate, hinting that it may be beneficial to utilize higher degree splines in the presence of noise. Fig. 4(b) shows relative errors for the same volumes.

Again, due to space constraints, only results for κ_1 are shown, but relative performances are similar for κ_2 and κ_3. Because κ_3 is the smallest of the curvatures its relative error tends to be larger. For κ_2, the average absolute error across all the rotations is about 4.9×10^{-5} for **OPF9**, about 0.004 for **DF3.00**, about 0.003 for **BS4**, and about 0.008 for **TEF**. For κ_3, the average absolute error across all the rotations is about 1.3×10^{-8} for **OPF9**, about 1.2×10^{-6} for **DF3.00**, about 1.4×10^{-6} for **BS4**, and about 3.1×10^{-6} for **TEF**.

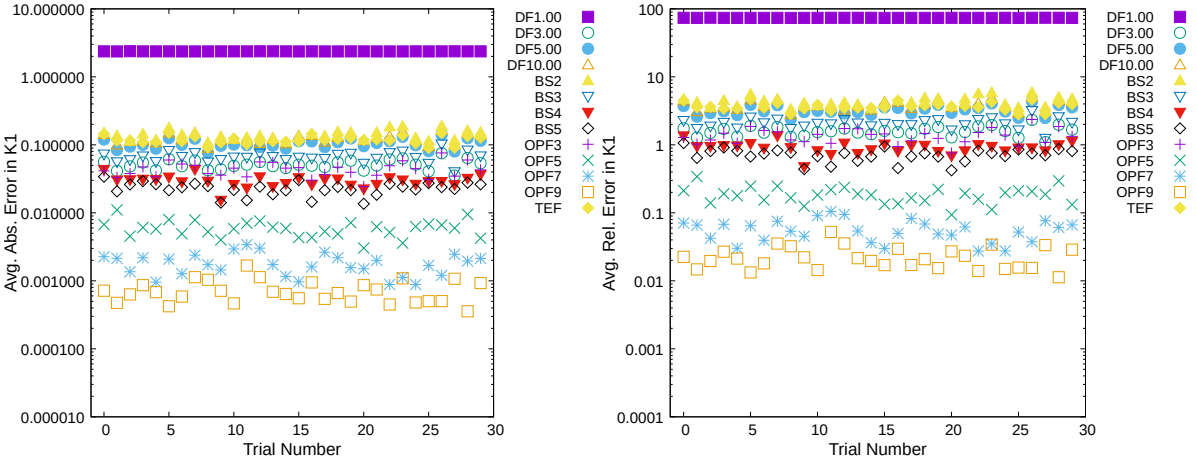
6.2.2 *Genus3*

Fig. 5(a) shows the average absolute error in κ_1 for the **OPF** method for 30 trials of noise with standard deviation of 0.25. Here, the results are similar to the Spheres



(a) Absolute Errors (b) Relative Errors

Figure 4: Error levels in κ_1 on noise-added Spheres



(a) Absolute Errors (b) Relative Errors

Figure 5: Error levels in κ_1 on noise-added Genus3

case. **OPF9** is best followed by **OPF7** and **OPF5**. Again, either **BS4** or **BS5** is next most accurate. Fig. 5(b) shows relative errors for the same volumes.

Again, due to space constraints, only results for κ_1 are shown, but relative performances are similar for κ_2 and κ_3. Due to the fact that κ_3 is the smallest of the curvatures, though, its relative error tends to be larger. For κ_2, the average absolute error across all the rotations is about 0.0002 for **OPF9**, about 0.1 for **DF2.00**, about 0.01 for **BS5**, and about 0.07 for **TEF**. For κ_3, the average absolute error across all the rotations is about 0.0005 for **OPF9**, about 0.03 for **DF2.00**, about 0.02 for **BS5**, and about 0.1 for **TEF**.

7 CONCLUSION

In this paper we have exhibited a framework for generating volumes and testing the accuracy of hypersurface curvature determination methods. In addition, we have introduced two new methods for determining such curvature. We have also compared these methods against two existing methods. One of these new methods, **BS**, uses B-Splines to estimate derivatives and then these estimated derivatives are used to determine the three hypersurface curvatures. The other method, **DF**, uses convolution with Deriche filters to estimate derivatives.

In summary, in our tests we found that **TEF** often exhibits the lowest error levels when no noise is present. When noise is present, **OPF** tends to exhibit much lower error levels than **TEF**, especially at larger filter sizes like **OPF9**. The new **BS3** method exhibits competitive accuracy on noise-free data.

In future works, we plan to introduce additional curvature determination methods and perform evaluations on a larger number of volumes and a larger number of parameter settings.

8 REFERENCES

[Ald14] G. Aldrich, A. Gimenez, M. Oskin, et al. Curvature-based crease surfaces for wave visualization. *Vision, Modeling & Vis.*, pp. 39–46, 2014.

[Bel12] A. Belyaev. Focal surfaces, skeletons, and ridges for shape analysis and processing. *Proc., 2012 Joint Int'l Conf. on Human-Centered Comput. Environ.*, HCCE '12, pp. 58–58, 2012.

[Bes86] P. J. Besl and R. C. Jain. Invariant surface characteristics for 3d object recognition in range images. *Comput. Vision, Graphics, Image Processing*, 33(1):33 – 80, 1986.

[Bib16] P. Bibiloni, M. González-Hidalgo, and S. Massanet. A survey on curvilinear object segmentation in multiple applications. *Pattern Recognition*, 60:949–970, Dec 2016.

[Der90] R. Deriche. Fast algorithms for low-level vision. *IEEE T-PAMI*, 12(1):78–87, Jan 1990.

[Fre14] Free Software Foundation. libmatheval (version 1.1.11), 2014. https://www.gnu.org/software/libmatheval/.

[Ham94] B. Hamann. Curvature approximation of 3D manifolds in 4D space. *Comput. Aided Geometric Design*, 11(6):621 – 632, 1994.

[Hau19] J. D. Hauenstein and T. S. Newman. Exhibition and evaluation of two schemes for determining hypersurface curvature in volumetric data. *J. of WSCG*, 27(2):121–129, 5 2019.

[Hau20] J. D. Hauenstein and T. S. Newman. Descriptions and evaluations of methods for determining surface curvature in volumetric data. *Computers & Graphics*, 86:52–70, 2 2020.

[Hir18] Y. Hirano. Categorization of lung tumors into benign/malignant, solid/GGO, and typical benign/others. K. Suzuki and Y. Chen, editors, *Artif. Intel. in Decision Support Systems for Diag. in Medical Imaging*, pp. 193–208. 2018.

[Hul12] R. Hulìk and P. Kršek. Local projections method and curvature approximation on 3D polygonal models. *Communic. Papers Proc., WSCG 2012*, pp. 223–230, 2012.

[Kin03] G. Kindlmann, R. Whitaker, T. Tasdizen, and T. Möller. Curvature-based transfer functions for direct volume rendering: methods and applications. *Proc., Vis. '03*, pp. 513 – 520, 2003.

[Kro19] M. Kronenberger, O. Wirjadi, and H. Hagen. Empirical comparison of curvature estimators on volume images and triangle meshes. *IEEE T-Vis. Comput. Graphics*, 25(10):3032–3041, 2019.

[Lef17] D. Lefloch, M. Kluge, H. Sarbolandi, et al. Comprehensive use of curvature for robust and accurate online surface reconstruction. *IEEE T-PAMI*, 39(12):2349–2365, Dec 2017.

[Lef18] L. Lefkovits and S. Lefkovits. Two-phase MRI brain tumor segmentation using random forests and level set methods. *Short Papers Proc., WSCG 2018*, pp. 152–159, 2018.

[Möl98] T. Möller, K. Mueller, Y. Kurzion, et al. Design of accurate and smooth filters for function and derivative reconstruction. *Proc., IEEE Symp. Volume Vis. 1998*, pp. 143 – 151, 1998.

[Mon92] O. Monga and S. Benayoun. Using partial derivatives of 3D images to extract typical surface features. Research Report RR-1599, INRIA, 1992.

[Myl15] M. Myllykoski, R. Glowinski, T. Karkkainen, and T. Rossi. A gpu-accelerated augmented lagrangian based l1-mean curvature image denoising algorithm implementation. *Full Papers Proc., WSCG 2015*, pp. 119–128, 2015.

[Özc21] B. Özcetin and M. Düldül. Curvature computations for the intersection curves of hypersurfaces in euclidean n-space. *Computer Aided Geometric Design*, 84(101954), 2021.

[Sol06] O. Soldea, G. Elber, and E. Rivlin. Global segmentation and curvature analysis of volumetric data sets using trivariate b-spline functions. *IEEE T-PAMI*, 28(2):265 – 278, 2006.

[Sou16] M. Soufi, H. Arimura, K. Nakamura, et al. Feasibility of differential geometry-based features in detection of anatomical feature points on patient surfaces in range image-guided radiation therapy. *Int'l J. of Comput. Assisted Radiology and Surgery*, 11(11):1993–2006, 2016.

[Suz18] H. Suzuki, Y. Kawata, N. Niki, et al. Automated assessment of aortic and main pulmonary arterial diameters using model-based blood vessel segmentation for predicting chronic thromboembolic pulmonary hypertension in low-dose ct lung screening. *Medical Imaging 2018: Comput.-Aided Diagnosis*, volume 10575, 2018.

[Sya17] M. A. Syarif, T. S. Ong, A. B. J. Teoh, and C. Tee. Enhanced maximum curvature descriptors for finger vein verification. *Multimedia Tools and Appl.*, 76(5):6859–6887, Mar 2017.

[Wan16] C. Wang and K. Siddiqi. Differential geometry boosts convolutional neural networks for object detection. *Proc., Diff. Geo. in Comp. Vision and Machine Learn. Work. (CVPRW Proc.)*, pp. 1006–1013, June 2016.

[Wei02] T. Weinkauf and H. Theisel. Curvature measures of 3d vector fields and their applications. *J. of WSCG*, 10(2):507–517, 2002.

[Woo10] B. A. Wood, J. K. Lee, M. Maskey, and T. S. Newman. Higher order approximating normals and their impact on isosurface shading accuracy. *Machine Graphics & Vision Intl. J.*, 19(2):201–221, 2010.