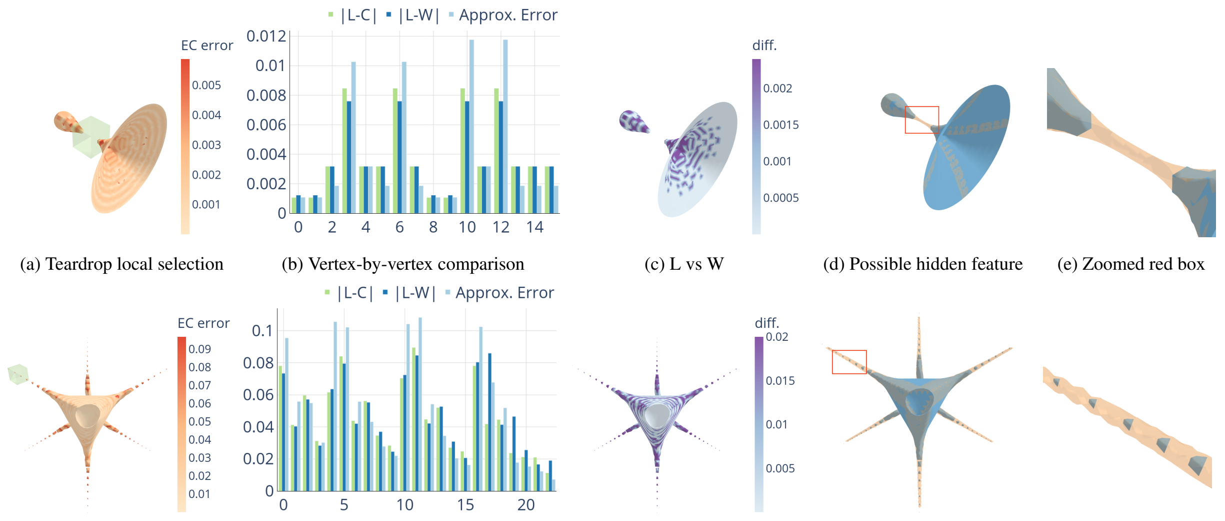
Estimation and Visualization of Isosurface Uncertainty from Linear and High-Order Interpolation Methods

Timbwaoga A. J. Ouermi [*]
SCI Institute, University of Utah

Jixian Li [†]
SCI Institute, University of Utah

Tushar M. Athawale [‡]
Oak Ridge National Laboratory

Chris R. Johnson [§]
SCI Institute, University of Utah



(a) Teardrop local selection (b) Vertex-by-vertex comparison (c) L vs W (d) Possible hidden feature (e) Zoomed red box

(f) Tubey local selection (g) Vertex-by-vertex comparison (h) L vs W (i) Possible hidden feature (j) Zoomed red box

Figure 1: Our proposed visualization system highlights the geometric and topological errors introduced by linear interpolation methods and allows users to query local vertex differences between interpolation methods. The first column (Fig. 1a and Fig. 1f) shows the approximated isosurface uncertainty and local selection using the colormap and transparent box, respectively. The second column (Fig. 1b and Fig. 1g) shows the differences between linear and cubic, linear and WENO methods and the approximated error for each vertex inside the transparent boxes. The third column shows a global comparison between linear and WENO interpolation methods. The fourth and fifth columns (Fig. 1d, Fig. 1e, Fig. 1i, and Fig. 1j) show a comparison between isosurfaces with (transparent orange) and without (opaque blue) possible hidden features that indicates isosurface feature uncertainty.

Abstract

Isosurface visualization is fundamental for exploring and analyzing 3D volumetric data. Marching cubes (MC) algorithms with linear interpolation are commonly used for isosurface extraction and visualization. Although linear interpolation is easy to implement, it has limitations when the underlying data is complex and high-order, which is the case for most real-world data. Linear interpolation can output vertices at the wrong location. Its inability to deal with sharp features and features smaller than grid cells can lead to an incorrect isosurface with holes and broken pieces. Despite these limitations, isosurface visualizations typically do not include insight into the spatial location and the magnitude of these errors. We utilize high-order interpolation methods with MC algorithms and interactive visualization to highlight these uncertainties. Our visualization tool helps identify the regions of high interpolation errors. It also allows users to query local areas for details and compare the differences between isosurfaces from different interpolation methods. In addition, we employ high-order methods to identify and reconstruct possible features that linear methods cannot detect. We showcase how our visualization tool helps explore and understand the extracted isosurface errors through synthetic and real-world data.

Index Terms: Marching cubes, linear interpolation, high-order interpolation isosurface uncertainty, weighted essentially non-oscillatory method

[*]e-mail: touermi@sci.utah.edu
[†]e-mail: jixianli@sci.utah.edu
[‡]e-mail: athawaletm@ornl.gov
[§]e-mail: crj@sci.utah.edu

1 Introduction

Isosurface visualization is fundamental for the exploration and analysis of 3D volumetric data. Several domains, including medical imaging [30, 52, 43, 19], dynamic simulations [37, 15], and crystallography [45] rely on isosurface visualization to explore and analyze important features that provide key insight for decision-making. The marching cubes (MC) [31] algorithm is widely used for extracting and visualizing these isosurfaces. The MC algorithms are commonly coupled with linear interpolation. Although local and fast, linear interpolation can introduce significant errors that lead to undesirable visual artifacts and degrade the isosurface approximation [17, 13]. In addition, the standard MC algorithms with linear interpolation fail to detect and reconstruct hidden and sharp features not captured by the mesh resolution [29, 26, 22, 23, 38, 53, 14]. Sharp features correspond to edges and corners, while hidden features are isosurface patches ignored by MC topological cases and linear interpolation. Despite the MC algorithms' well-known errors and limitations, most visualizations don't offer insights into how those errors could impact the extracted isosurface. Instead of treating the extracted isosurface as a complete piece, we observe that some regions are more (or less) trustworthy than others. Characterizing the quality of extracted isosurfaces and the type of errors can help better understand the reliability and limitations of the extracted isosurface and its features. However, obtaining the true surface specification is often impossible due to insufficient information. Scalar fields are commonly stored on a sampled uniform grid. Linear interpolation directly computes the surface's geometry using those values, ignoring some of the important information, such as gradients or high-order coefficients. Therefore, it is important to highlight the regions with high errors and variations and allow the users to determine the correct surfaces based on their expertise and domain-specific knowledge.

To investigate the isosurface uncertainty, one considers the errors caused by noisy input data (data uncertainty) and/or the isosurface extraction procedure (model uncertainty). Here, we investigate the latter in the context of interpolation methods. Current approaches have focused on characterizing the uncertainty from noisy data applied to MC with linear interpolation [2, 44, 41, 3, 4, 20]. These approaches apply probabilistic methods to the noisy data to estimate the isosurface uncertainty. Other approaches compare the extracted isosurface against a high-resolution target isosurface [11]. Many real-world examples lack finer resolution datasets or target isosurface to compare against. Several studies propose quadratic and cubic interpolation methods to improve the accuracy of level-crossing at each edge, and therefore the isosurface accuracy [17, 13, 12, 32, 33, 8, 46]. However, these methods don't provide sufficient insight into the accuracy improvement from linear to higher-order interpolation.

To address the issue of feature recovery, several studies extend the original MC algorithm to recover sharp features by inserting additional points or refining the cells containing the sharp features [29, 26, 22, 23, 38, 53, 14]. These methods require access to normals or finer-resolution data to identify and recover sharp edges. In practice, finer resolution data are often unavailable and normals at the interior of the cell edges are approximated using linear interpolation, which has large errors as previously indicated. Moreover, these methods do not identify features in cells where all node values are above or below the provided isovalue.

In this paper, we provide insights into uncertainties caused by interpolation methods through interactive visualization and address the aforementioned limitations. We summarize our contribution as follows: (1) We construct an analytical error approximation of the edge-crossing vertex in the MC algorithms for visualizing the reconstructed isosurface error. This method effectively approximates the edge-crossing error and is computationally more efficient than the isosurface comparison methods. In addition, the method is applicable to any volumetric data as it directly computes the estimated error from the volume data and doesn't require sampling or additional information such as ensemble or high-resolution data. (2) We introduce a method for detecting and reconstructing possible hidden features missed with the MC algorithm with linear interpolation. We use the divided differences (slopes) of each cell's edges and its neighbors to identify the cell with possible hidden features. The target cells are fitted with local cubic Lagrange polynomials that are then used to divide the cell and reconstruct the hidden features. The current approaches for sharp feature reconstruction don't recover hidden features because they don't consider cells where all node values are above or below the provided isovalue. (3) We present a visualization tool that employs error approximation, isosurface positional variation for different interpolation methods, and possible feature reconstruction with other techniques to provide a platform for the exploration and analysis of isosurface uncertainty. The framework can effectively highlight edge-crossing errors and isosurface feature uncertainty.

This manuscript has been authored by UT-Battelle, LLC under Contract No. DE-AC05-00OR22725 with the U.S. Department of Energy. The publisher, by accepting the article for publication, acknowledges that the U.S. Government retains a non-exclusive, paid up, irrevocable, world-wide license to publish or reproduce the published form of the manuscript, or allow others to do so, for U.S. Government purposes. The DOE will provide public access to these results in accordance with the DOE Public Access Plan (`http://energy.gov/downloads/doe-public-access-plan`).

2 Related Work

2.1 Edge-Crossing Uncertainty

The approximation and visualization of isosurface uncertainty is a challenging problem [7, 25]. Statistical methods for parametric [2] and nonparametric [41, 3] models provide measure metrics that can be used to visualize the most probable isosurface and its uncertainty. For instance, Athawale et al. provided a closed form for computing the expected position and variance of the level-crossing in the MC algorithm for parametric [2] and nonparametric [3] distributions. Topology case count and entropy-based methods can resolve ambiguity in MC algorithm and visualize isosurface uncertainty [4]. The statistical approaches may require solving the level-set crossing problem for each cell many times or sampling methods such as Monte Carlo sampling algorithms that are computationally expensive [20, 48]. The closed forms in [2, 3, 4] improve the computational performance for independent noise models, however, no closed forms are available for more complex noise models such as multivariate Gaussian noise models. The isosurface uncertainty characterized by the statistical methods relies on ensemble data and doesn't explicitly account for the uncertainty from the interpolation method (model uncertainty) which is the focus of this work. When the target isosurface is accessible, the uncertainty can be derived by computing the error between the target and approximated isosurfaces [11, 1]. However, the error computation is computationally expensive as it relies on isosurface sampling, and the target isosurface is often unavailable.

2.2 Feature Uncertainty

We consider the isosurface variation from MC with and without feature-preserving methods. These feature uncertainties impact the overall isosurface structure. Several studies have extended the MC algorithms to incorporate feature-preserving techniques that can recover these sharp features [29, 26, 22, 23, 38, 53, 14, 5]. These methods use information about the cell derivatives to better represent the underlying sharp features. Recently, machine learning-based approaches have been proposed for more accurate MC with feature preservation [10, 16, 9, 42, 18].

Kobelt et al. [29] propose a surface extraction method from directed distance field and surface normals of a geometric object that preserves sharp features. The normals are used to detect the sharp features and new sample points are added inside the cell to recover the hidden features. The Dual contouring algorithm proposed in [26] uses the edges intersection and the normals at those intersections to process the cells with sharp features. This method doesn't explicitly require identifying the cell with sharp features because it uses a quadratic error function to automatically place the additional points. Ho et al. [23] propose sampling the edges normals to detect the cell with sharp features in volumetric data. These cells are then subdivided to represent the sharp features. The adaptive refinement of the cell requires access to a finer-resolution version of the data.

3 Technical Background

3.1 Marching Cubes Algorithm

The MC algorithm [31] extracts the isosurface as it steps through each cubical cell of the uniform grid. For a single cell, the algorithm first determines the topological configuration based on the relationship between the value on each vertex and the isovalue k. The values on the vertices could be either larger or smaller than the isovalue. On each edge, the isosurface crosses the edge if one of the vertex has a value larger than the isovalue while the other is smaller. We connect the edge-crossing points to form surfaces contributing to the final output surface. Comparing vertex values only shows whether there is an edge-crossing point. To determine the exact location of the edge-crossing point, we need to identify the point on the edge where the value is equal to our isovalue. Therefore, we need a method for interpolation between the two vertices of an edge. We will introduce different methods of interpolation in the rest of this section.

3.2 Linear Interpolation

Estimating edge-crossing points using linear approximation has advantages in terms of speed and simplicity of the mathematical model. Let $f(x_0)$ and $f(x_1)$ be the scalar values sampled at vertex positions x_0 and x_1 denoting ends of a cell edge, respectively. The crossing position for the isovalue k on this cell edge is determined by finding x such that $k = f(x_0) + \frac{f(x_1)-f(x_0)}{x_1-x_0}(x - x_0)$. The solution is $x = x_0 + \frac{x_1-x_0}{f(x_1)-f(x_0)}(k - f(x_0))$. To take advantage of vector arithmetic the solution can written as follows: $x = \alpha * x_1 + (1-\alpha) * x_0$, where $\alpha = \frac{k-f(x_0)}{f(x_1)-f(x_0)}$.

3.3 Cubic Interpolation

Although linear interpolation is efficient, it may lead to significant approximation errors that degrade the quality of the extracted isosurface. Many studies have proposed higher order interpolation methods to improve the accuracy of the level crossing at each edge, and therefore the isosurface accuracy [17, 13, 12, 32, 33, 8, 46].

For the same edge (or interval) considered in Sec. 3.2 the cubic interpolant is $q(x) = c_0 + c_1 x + c_2 x^2 + c_3 x^3$ with $x \in [x_0, x_1]$. The cubic polynomial has four degrees of freedom and requires solving a 4×4 system of linear equations to compute the coefficients $c_i,\ i = 0, \cdots, 3$. A common approach is to use the sampled data values and derivatives at the edge endpoints to build the system of linear equations and find the coefficients. Let $(f(x_0), f'(x_0))$ and $(f(x_1), f'(x_1))$ be the data values at x_0 and x_1, respectively. The coefficients are obtained by solving

$$q(x_0) = f(x_0) \quad q(x_1) = f(x_1) \quad q'(x_0) = f'(x_0) \quad q'(x_1) = f'(x_1).$$

The crossing position for isovalue k on this edge is obtained by finding the roots to $q(x) = k$. We note that the derivatives $f'(x_0)$ and $f'(x_1)$ are often not available and therefore approximated using finite difference methods.

3.4 WENO Interpolation

The weighted essentially non-oscillatory (WENO) method [54] is a high-order polynomial reconstruction method developed for solving hyperbolic and convection-diffusion equations. WENO achieves high-order accuracy in smooth regions and provides a better representation of regions with sharp gradients compared to standard Lagrange interpolation [21].

Let's consider the 1D mesh $\mathscr{M} = \{\cdots, x_{i-2}, x_{i-1}, x_i, x_{i+1}, \cdots\}$ where $i \in \mathbb{N} \cup \{0\}$. For each edge E_i where the isovalue k lies between f_i and f_{i+1}, a high-order polynomial $p_i(x)$ is used to approximate the function inside the interval defined by the edge boundaries. The edge-crossing is obtained by finding the roots of the implicit equation $p_i(x) = k$. The final interpolant $p_i(x)$ is a convex combination of the third-order polynomials $p^{(1)}(x)$, $p^{(2)}(x)$, and $p^{(3)}(x)$.

$$p_i(x) = w_1 p^{(1)}(x) + w_2 p^{(2)}(x) + w_3 p^{(1)}(x), \tag{1}$$

where w_1, w_2, and w_3 are nonlinear weights such that $w_1 + w_2 + w_3 = 1$. The nonlinear weights are obtained using the "smoothness" indicator β_j in [24] that can be approximated as follows:

$$\begin{aligned}
\beta_1 &= 13/12\big(f_{i-2} - 2f_{i-1} + f_i\big)^2 + 1/4\big(f_{i-2} - 4f_{i-1} + 3u_i\big)^2,\\
\beta_2 &= 13/12\big(f_{i-1} - 2f_i + f_{i+1}\big)^2 + 1/4\big(f_{i-1} - f_{i+1}\big)^2,\\
\beta_3 &= 13/12\big(f_i - 2f_{i+1} + f_{i+2}\big)^2 + 1/4\big(3f_i - 4f_{i+1} + f_{i+2}\big)^2.
\end{aligned} \tag{2}$$

The nonlinear weights are dependent on the constants $\gamma_1 = 1/10$, $\gamma_2 = 3/5$, $\gamma_3 = 3/10$. Using the "smoothness" indicator in Eq. (2) and the constants, the nonlinear weight can be expressed as follows:

$$w_j = \alpha_j/(\alpha_1 + \alpha_2 + \alpha_3), \quad \alpha_j = \gamma_j/(\varepsilon + \beta_j)^2, \quad j = 1,2,3. \tag{3}$$

The parameter ε, typically set to 10^{-6}, is introduced to avoid division by zero. Jiang and Shu [24] proved that in smooth regions the approximation $p_i(x)$ in Eq. (1) is fifth-order accurate. Solving $p_i(x) = k$ is equivalent to a root-finding problem for a cubic polynomial. The roots for the WENO polynomial can be found using the cubic formula in [51]. Similar to [17], the median solution is selected in the cases where multiple valid roots are found.

4 Method

4.1 Edge-Crossing Error Approximation

We propose an edge-crossing error approximation for MC algorithms derived from polynomial interpolation error, which is used to visualize isosurface discrepancies and highlight regions with significant errors. Taylor series expansion is widely used for local function and error approximation function. For instance, in the context of visualization, Moller et al. [36, 35, 34] use it to develop smooth filters for volume data approximation and estimate their local error. We distinguish our approach by using the Taylor series expansion to estimate the edge-crossing error, which has the advantage of providing insight into the isosurface reconstruction error without the need to solve a linear system. The interpolation error from the linear approximation $\ell(x)$ of the function $f(x)$ is

$$e(x) = \ell(x) - f(x) = \frac{f''(\xi)}{2}(x - x_i)(x - x_{i+1}), \tag{4}$$

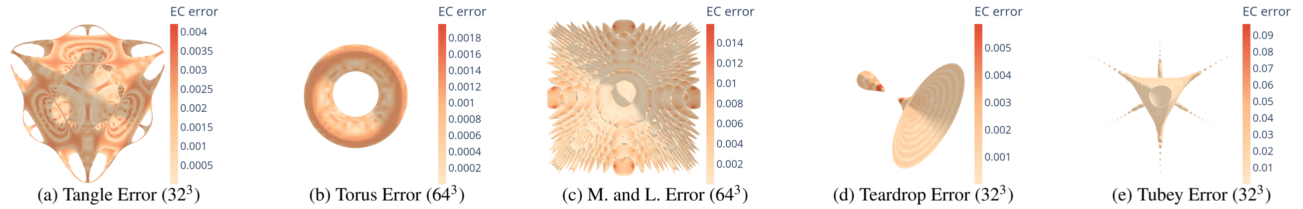
where $\xi \in (x_{i-1}, x_{i+2})$. The MC algorithms solve the implicit problem $\ell(x) = k$ where $k \in \mathbb{R}$ is the isovalue. Let x_* and $\bar{x}_*$ the solutions to $f(x) = k$ and $\ell(x) = k$. Substituting the solutions x_* and $\bar{x}_*$ into $f(x)$ and $\ell(x)$ gives.

$$\ell(\bar{x}_*) = f_i + U[i, i+1](\bar{x}_* - x_i) = k, \text{ and} \tag{5}$$

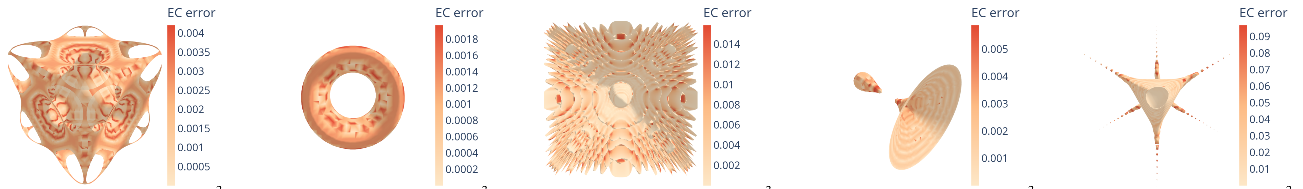
$$f(x_*) = f_i + U[i, i+1](x_* - x_i) + \frac{f''(\xi)}{2}(x_* - x_i)(x_* - x_{i+1}) = k. \tag{6}$$

The divided difference $U[i, i+1]$ is recursively defined as

$$\begin{aligned}
U[i] = f(x_i) = f_i, \quad U[i, i+1] &= \frac{U[i+1] - U[i]}{x_{i+1} - x_i},\\
U[i, i+j] &= \frac{U[i+1, \cdots, i+j] - U[i, \cdots, i+j-1]}{x_{i+j} - x_i},
\end{aligned} \tag{7}$$

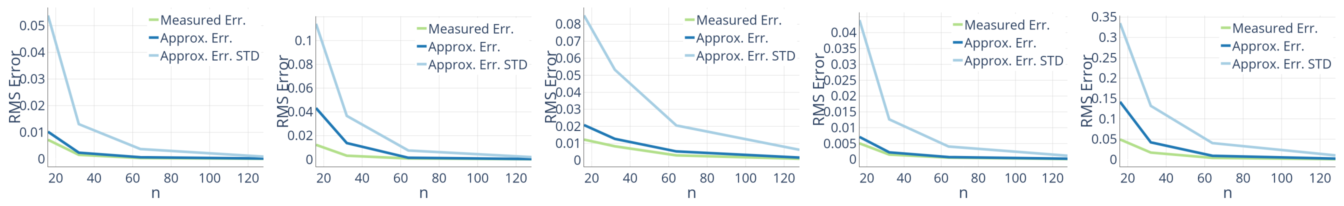


(a) Tangle Error (32^3) (b) Torus Error (64^3) (c) M. and L. Error (64^3) (d) Teardrop Error (32^3) (e) Tubey Error (32^3)



(f) Tangle Approx. Error (32^3) (g) Torus Approx. Error (64^3) (h) M. and L. Approx. Error (64^3) (i) Teardrop Approx. Error (32^3) (j) Tubey Approx. Error (32^3)

Figure 2: Comparison between measured and approximate error. The first row corresponds to the measured error and the second to the approximated error. Each column from left to right corresponds the **Tangle** (Fig. 2a and Fig. 2f with $k = 0.1$), **Torus** (Fig. 2b and Fig. 2g with $k = 0.0$), **Marschner and Lobb** (Fig. 2c and Fig. 2h with $k = 0.5$), and **Teardrop** (Fig. 2d and Fig. 2i with $k = -0.001$) and **Tubey** (Fig. 2e and Fig. 2j with $k = 0.0$) examples. Our approximated errors show similar patterns to the measured errors. In most cases, it slightly overestimates the errors.



(a) Tangle (b) Torus (c) Marschner and Lobb (d) Teardrop (e) Tubey

Figure 3: Comparison of measured (in green), our estimated (in blue), and standard approach for error approximation (light blue). The columns from left to right show errors for the **Tangle** (Fig. 3a), **Torus** (Fig. 3b), **Marschner and Lobb** (Fig. 3c), **Teardrop** (Fig. 3d), and Tubey (Fig. 3e). Our error estimation in Eq. (11) is much closer to the measured error compared to the standard approach in Eq. (10).

with j being an integer. Subtracting Eq. (5) from Eq. (6) gives the edge-crossing error

$$|x_* - \bar{x}_*| = \Big|\frac{1}{U[i,i+1]}\frac{f''(\xi)}{2}(x_* - x_i)(x_* - x_{i+1})\Big|. \qquad (8)$$

The edge-crossing approximation error can be bounded by

$$|x_* - \bar{x}_*| \leq \frac{\max_{\xi\in[x_i,x_{i+1}]}|f''(\xi)|}{2U[i,i+1]}(x_* - x_i)(x_* - x_{i+1}) \qquad (9)$$

In practice, ξ, f'', and x_* are not available. Typically the term $\max_{\xi\in[x_i,x_{i+1}]}|f''(\xi)|$ is approximation using finite difference $max\big(|U[i-1,i+1]|,|U[i,i+2]|\big)$. The product $(x_* - x_i)(x_* - x_{i+1})$ is approximate with the interval size $(x_i - x_{i+1})^2$. The error bound on the right side of Eq. (9) is approximated as follows:

$$\bar{e}_b = \frac{max\Big(|U[i-1,i+1]|,|U[i,i+2]|\Big)}{U[i,i+1]}(x_i - x_{i+1})^2 \qquad (10)$$

The approximation $\bar{e}_b$ in Eq. (10) tends to overestimate the edge-crossing errors $|x_* - \bar{x}_*|$. We estimate the vertex approximation error $\bar{e} \approx |x_* - \bar{x}_*|$ as follows:

$$\bar{e} = \frac{max\big(|U[i-1,i+1]|,|U[i,i+2]|\big)}{U[i,i+1]}(\bar{x}_* - x_i) * (\bar{x}_* - x_{i+1}) \quad (11)$$

Eq. (11) provides a much tighter approximation of the error compare to Eq. (10). Fig. 4 shows the difference between the underlying function (black curve) and its linear approximation (orange line). The black horizontal line indicates the edge considered and the blue line shows the isovalue position of the underlying function (black curve) and the linear interpolation (orange line).

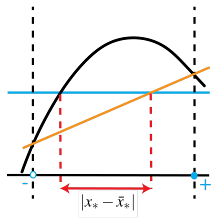


Figure 4: Edge-crossing uncertainty. The underlying function and the linear interpolation are shown in black and orange, respectively. The black line segment with the positive and negative nodes is the edge and the blue line indicates the target isovalue. The red double arrow indicates the approximation error. The isovalue is indicated by the blue horizontal line.

We use several datasets to evaluate the approximated edge-crossing error introduced in Eq. (11). The volume datasets are sampled from a **Tangle**, **Torus**[28], **Marschner and Lobb**[32], **Teardrop** [28], and **Tubey** [6] functions. Their equations are provided in the appendix.

The functions are sampled on a 512^3 uniform mesh to construct the high-resolution data from which the target isosurfaces are extracted. The isosurface error is obtained by calculating the difference between the high- (target) and coarse-resolution isosurfaces using METRO [11]. The results in Fig. 2 and Fig. 3 evaluate and validate the edge-crossing error introduced in Eq. (11). The first and second rows in Fig. 2 show the measured and approximated errors, respectively. The label “EC error” represents the edge-crossing error. In each example, the measured and approximated errors exhibit similar patterns, demonstrating that the edge-crossing error approximation offers a fast and reliable estimate of the isosurface error arising from linear interpolation. This method is computationally more efficient because the approximated error is directly computed using Eq. (11) and doesn’t require additional data, computation, or sampling algorithm as in the case of statistical and isosurface comparison methods. For instance, the isosurface extraction, and error estimation for the tangle example in Fig. 2f takes less than a millisecond ($1ms$) whereas the measured error in Fig. 2a takes a few seconds. This cost is significantly magnified with the increase in grid resolution which can hinder the interactivity of visualization. Our approach provides a much more efficient way of visualizing linear interpolation uncertainty with the proposed approximation. The estimated isosurface uncertainty provides quick insight into the quality of the isosurface that can guide decisions about using higher-order interpolation, higher-resolution data, and/or other methods to improve the isosurface quality.

The results in Fig. 3 show the root mean squares (RMS) errors. The green line corresponds to the measured errors, the blue to our introduced method, and the light blue to the standard approach for polynomial error approximation. The standard approach (in light blue) [21] significantly overestimates the edge-crossing errors. Our method provides a better approximation of the measured error that can be utilized for visual analysis of isosurface error.

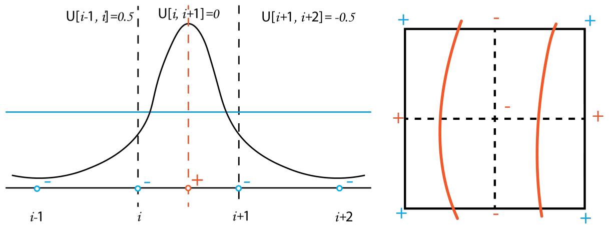
4.2 Hidden Features Detection and Reconstruction

The MC algorithms with and without sharp feature recovery fail to identify and reconstruct hidden features not captured by the mesh resolution. These hidden features are isosurface patches not detected by linear interpolation and the topological cases considered in MC algorithms. The hidden features can alter the isosurface connectivity and its overall structure. We propose a method for hidden feature detection and reconstruction that relies on the slopes (divided differences) on cell edges and high-order interpolation. We utilize this method to offer the user two possible isosurface reconstructions: one with hidden feature recovery and one without. Both isosurfaces are visualized together to highlight the differences and provide insight into the feature differences.

A cell might have a hidden feature if for any of its edges two of the three slopes $U[i-1,i]$, $U[i,i+1]$, and $U[i,i-1]$ of neighboring edges have opposite signs. The detected cell is divided into smaller subcells using tri-cubic Lagrange polynomial interpolation. We note that using linear interpolation instead for the cell refinement does not recover the missing hidden features. The MC algorithm is then applied to each subcell to reconstruct the hidden features.

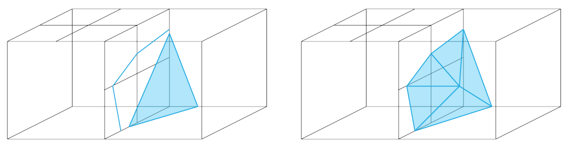
Fig. 5a shows a 1D example where the vertex crossings on the middle edge are detected using the slopes (divided differences) of the neighboring edges. The middle edge is split into two new edges that can then be used to detect and approximate the edge-crossing. The scalar value at the split location is obtained using a cubic Lagrange interpolation. Fig. 5b provides an illustration using a marching square in the 2D case. The MC algorithms with and without sharp feature-preserving methods don’t reconstruct the hidden feature in orange. This cell is considered above the desired isoline because all its node values are larger than the target isovalue. The cell outlined with black lines is divided into smaller cells, indicated by the dashed lines, using cubic Lagrange interpolation. These new cells reveal new edge-crossings that are used to represent the hidden feature and better approximate the overall isosurface.

These cell refinements lead to cracks in the reconstructed isosurface. Crack-free isosurface extraction techniques have been introduced in the context of adaptive mesh refinement [50, 49, 47]



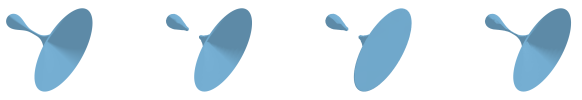
(a) 1D hidden feature recovery (b) 2D hidden feature recovery

Figure 5: Hidden feature recovery in 1D and 2D.Fig. 5a shows an example of a hidden feature between i and $i+1$ that can be detected by noting that $U[i-1,i+1]*U[i+1,i+2]<0$, meaning the slopes surrounding the hidden features have a different sign. Our method subdivides the cell at the orange dotted line to recover the hidden feature. The isovalue is indicated by the blue horizontal line. Fig. 5b shows a 2D example with hidden features that can be recovered by refining the cell. The orange curves are the isocontour inside the cell. MC will miss the contour because all four corners have the same sign. Our method identifies the hidden feature and subdivides the cell at the dotted black lines.



(a) (b)

Figure 6: Crack patching. Fig. 6a shows an isosurface crack caused by the refined cell. The cell on the left is divided according to our algorithm, while the cell on the right is from the original Marching Cubes. The extracted vertices on the interface of two cells are mismatched. In Fig. 6b the crack is fixed by (1) matching the boundaries of the blue triangle in Fig. 6a to the boundary of the blue line in Fig. 6a, (2) connecting the edges of the new polygon to the center of the triangle in Fig. 6a to form the crack-free triangulated patch..

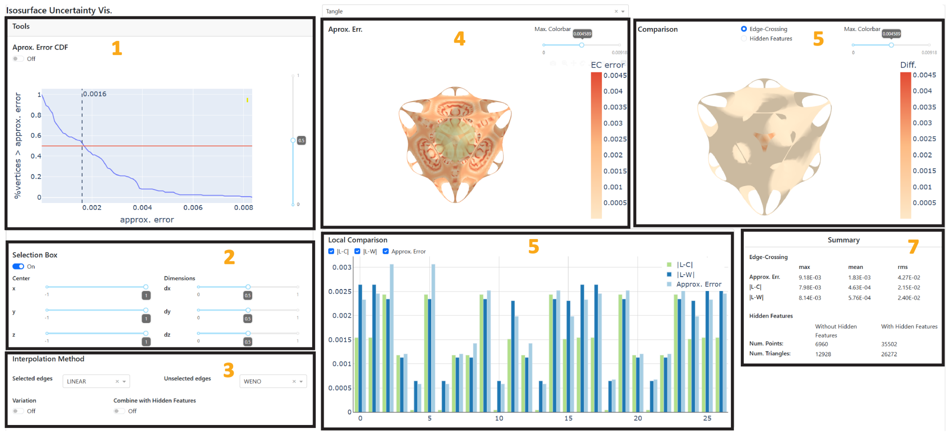


(a) target (b) standard MC (c) dual contouring (d) hidden feature

Figure 7: Comparison between the target, MC, dual contouring, and hidden feature reconstruction method. This example is based on the teardrop example with a resolution of 32^3 and an isovalue $k=-0.001$. The hidden feature recovery method can detect and reconstruct the missing feature that connects the two broken pieces.

Here, we resolve this issue by connecting the edges at hidden feature face boundaries with edges at the boundaries of the reaming isosurface. Additional edges indicated by the interior blue lines shown in Fig. 6b are introduced to reduce discontinuities at the cell interface. The cells at the boundaries of the isosurface with no hidden features are modified during the polygon extraction to ensure a crack-free isosurface, as shown in Fig. 6.

The interface is designed to highlight high errors and isosurface feature differences based on queries. Several filter tools (sliders, switches, checklists, radio buttons) are introduced for flexibility and to facilitate exploration and analysis. The different views are used to enable simultaneous visualization of different isosurface error



Aprox. Error CDF: Off

Selection Box: On

Center		Dimensions	
x	1	dx	0.5
y	1	dy	0.5
z	1	dz	0.5

Interpolation Method

Selected edges: LINEAR — Unselected edges: WENO

Variation: Off — Combine with Hidden Features: Off

Comparison: [✓ Edge-Crossing] Hidden Features

Edge-Crossing

	max	mean	rms
Approx. Err.	9.18E-03	1.83E-03	4.27E-02
|L-C|	7.98E-03	4.63E-04	2.15E-02
|L-W|	8.14E-03	5.76E-04	2.40E-02

Hidden Features

	Without Hidden Features	With Hidden Features
Num. Points:	6960	35502
Num. Triangles:	12928	26272

Figure 8: Our visualization framework has seven components. It shows the approximated error cumulative distribution function(CDF), the local selection box specifier, the interpolation method specifier, the approximated error overview, the local vertices comparison, the surface comparison, and the summary.

metrics. The interface design provides insight into the confidence of the extracted isosurface. Fig. 7 shows the ground truth, standard MC, dual contouring, and our hidden feature recovery method isosurfaces for the **Teardrop** dataset. The standard MC and dual contouring do not detect and reconstruct the missing piece. The dual contouring method inserts additional triangles to enforce closed surfaces. Our method successfully recovers the missing piece and yields an isosurface similar to the target solution.

4.3 Isosurface Uncertainty Visual Analysis Tool

Here, we introduce a framework for visualizing and analyzing MC isosurface uncertainty using C, Python, and Dash [1]. Our framework employs the error approximation in Sec. 4.1, the hidden feature-preserving method in Sec. 4.2, and several other techniques to enable visual analysis of isosurfaces uncertainty obtained from using different interpolation methods along with MC algorithms. The interface is designed to highlight high errors and isosurface feature differences based on queries. Several filter tools (sliders, switches, checklists, radio buttons) are introduced for flexibility and to facilitate exploration and analysis. The different views are used to enable simultaneous visualization of different isosurface error metrics. The interface design provides insight into the confidence of the extracted isosurface. Our framework has seven components indicated by the outlined rectangles and the corresponding component number shown in Fig. 8

The first component C_1 shows a plot of the cumulative percentage of vertices (y-axis) with respect to the approximated error (x-axis) introduced in Sec. 4.1. The vertical slider to the right of the plot in the first component is used to select a cumulative percentage. The switch below “Approx. Error CDF” turns on and off a binary color map on the isosurface shown in the fifth component. This component provides insight into the percentage of isosurface vertices below and above a selected threshold.

The second component C_2 is used to insert a transparent box inside the domain of the fifth component to show a selected local region. The switch below “Selection Box” must be on to activate the box feature. The position and size of the box are adjusted using “Center” and “Dimension”, respectively. This component facilitates detailed inspection of local isosurface uncertainty based on a selected region of interest.

The third component C_3 is used to select different interpolation methods for selected and unselected edges based on the error threshold in C_1 or local box selection in C_2. This feature enables the comparison between different interpolation methods.

[1]https://dash.plotly.com/

The fourth component C_4 visualizes the estimated isosurface uncertainty obtained from Sec. 4.1 and the local selection based on the parameters selected in C_2. This component highlights regions with high and low edge-crossing errors.

The fifth component C_5 shows a bar plot that provides a local comparison of the different interpolation methods for selected local regions. This enables a vertex-by-vertex comparison of the edge-crossing error and the difference between interpolation methods.

The sixth component C_6 visualizes the comparison between two selected interpolation methods using the “Edge-Crossing” radio button. The ”Hidden Feature” radio button enables the simultaneous visualization of both the isosurface with and without hidden features reconstruction. This component provides insight into the difference and accuracy gain between linear and higher interpolation methods. It also highlights the feature uncertainty between standard MC and hidden feature recovery.

The seventh component C_7 provides a summary of the isosurface uncertainties.

5 Results

5.1 Synthetic Examples

The synthetic examples are based on the **Tangle**, **Torus**, **Marschner and Lobb**, **Teardrop**, and **Tubey**. Figure 9 shows an exploration/analysis pipeline using the visualization framework to gain insight into the isosurface uncertainties from interpolation methods. The first column in Fig. 9 shows a selected error threshold (vertical dashed line) and the corresponding percentage of vertices with errors larger than the specified error threshold is indicated with the red horizontal line. The results in the second column in Fig. 9 are constructed from measured error obtained METRO [11]. These results show the isosurfaces with a binary colormap indicating the regions with isosurface errors that are above (in reg) and below (in light orange) the specified error threshold. The third column shows the maximum variation across all three interpolation methods. The fourth, fifth, and sixth columns in Fig. 9 show a comparison between linear and cubic (L vs. C), linear and WENO (L vs. W), and cubic and WENO (C vs. W) in the regions with errors larger than the selected threshold. The similar patterns between the second, third, fourth, and fifth columns in Fig. 9 indicate that the approximated error effectively identifies regions with large errors which corresponds to the regions with large variation between interpolation methods indicated in purple. The fourth and fifth columns of Fig. 9 show the possible accuracy improvement from linear to cubic and WENO. The linear, cubic, WENO interpolation are $O(h^2)$, $O(h^4)$, and $O(h^5)$ accurate. The fifth column highlights the differ-

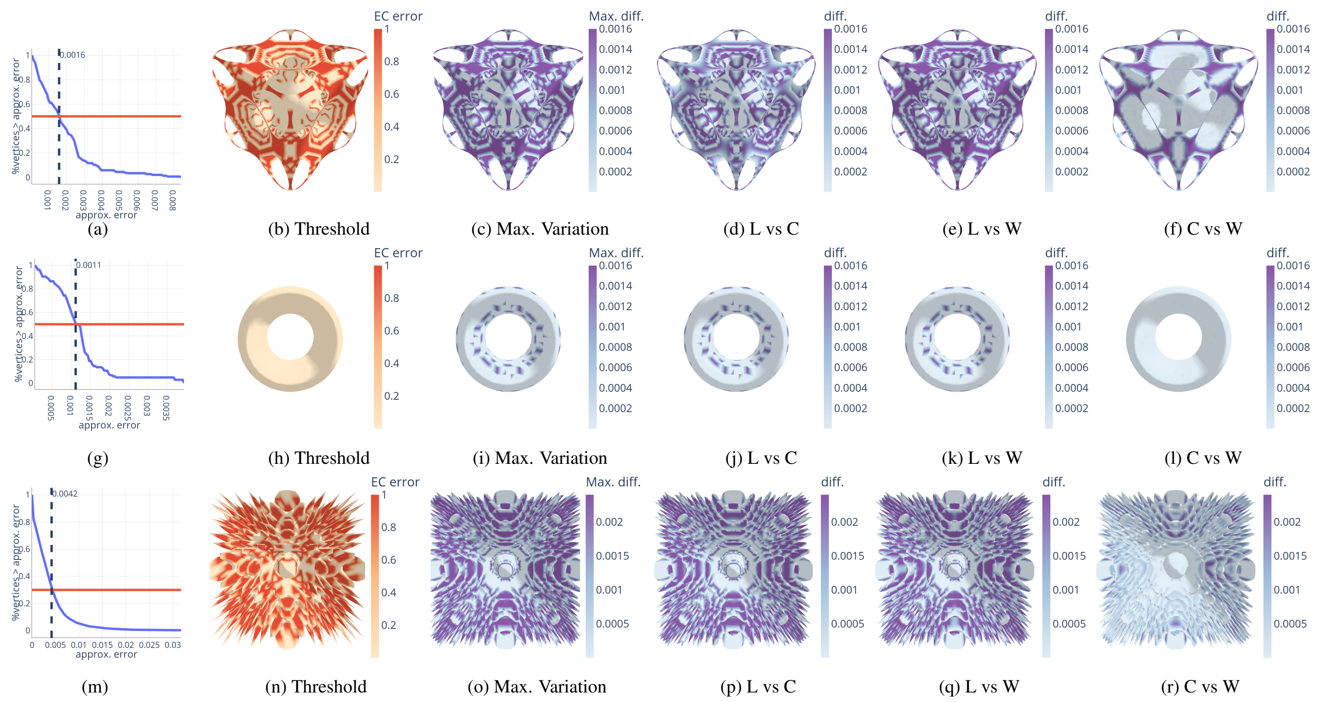


Figure 9: Isosurface comparison based on selected threshold value and interpolation methods (Components C_1, C_3, C_4 and C_6). The first column shows the selected threshold using the plot of the cumulative percentage of vertices (y-axis) with respect to the approximated error (x-axis). The second column shows the binary colormap based on the selected threshold. The third column shows the maximum variation. The fourth, fifth, and sixth columns show a comparison of L vs. C, L vs. C, and C vs. W.

ence between cubic and WENO. WENO further improves the cubic interpolation accuracy from $O(h^4)$ to $O(h^5)$. The improvement from cubic to WENO can be minor as shown in Fig. 9l.

The visualization tool in Fig. 8 uses C_2 for local uncertainty exploration , as shown in the first and second columns of the teaser image Fig. 1. The transparent boxes in Fig. 1a and Fig. 1f indicate the selected local region of interest. These selections correspond to the regions with broken pieces and hidden features. Fig. 1b and Fig. 1g show a local vertex-by-vertex comparison of the approximate edge-crossing error, L vs. C, and L vs. W. Fig. 1c and Fig. 1h show a comparison of L vs. W. This local vertex-by-vertex comparison enables a detailed comparison of the magnitude of the approximated error and the difference between interpolation methods. The fourth and fifth columns show a comparison of the isosurface with (transparent orange) and without (opaque blue) hidden feature recovery. The zoomed-in versions in Fig. 1e and Fig. 1j demonstrate that our proposed feature-reconstructions methods introduced in Sec. 4.2 successfully constructs possible connection among the broken components (the transparent orange isosurface) and propose an alternate isosurface to be considered.

5.2 Real-World Examples

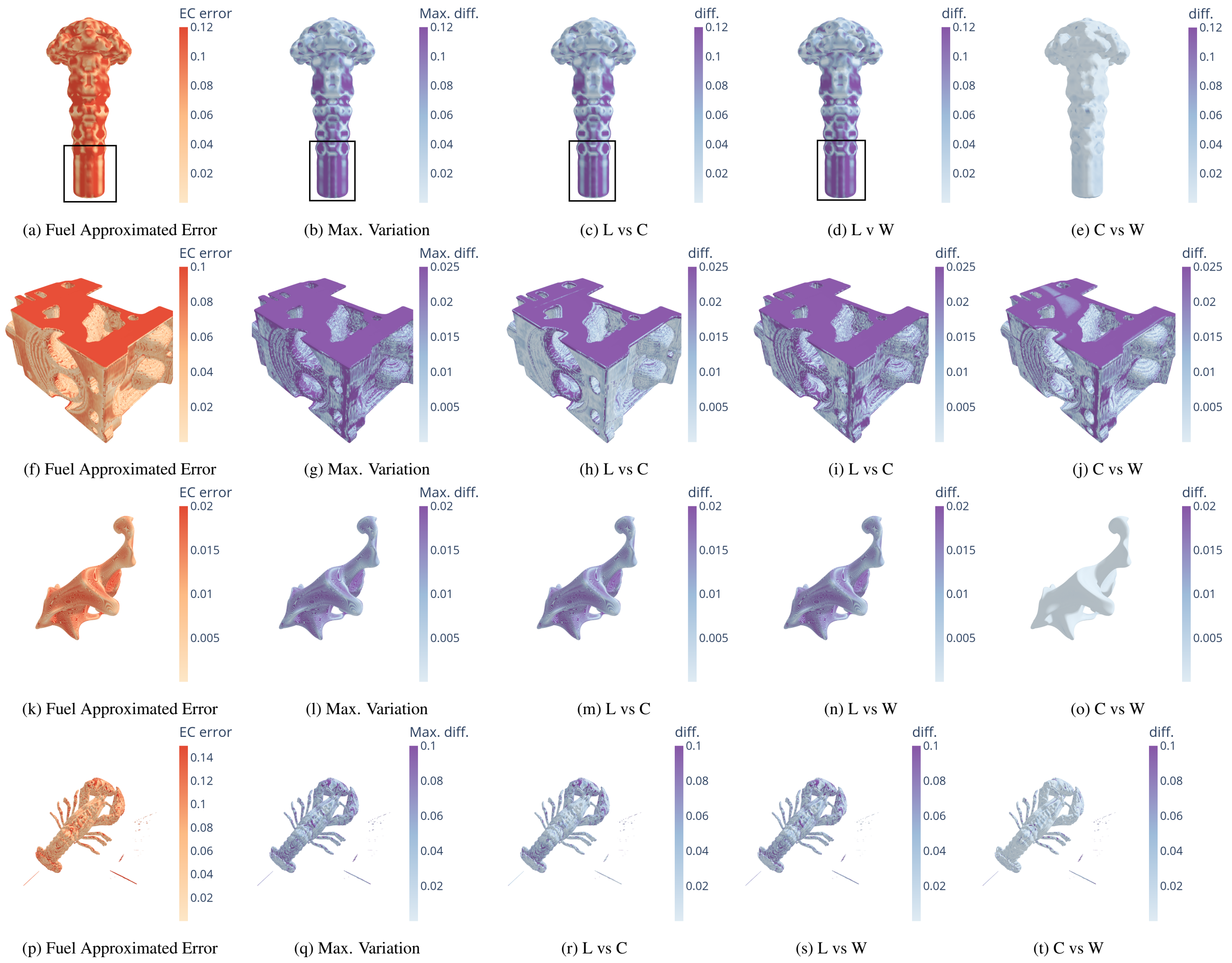
The datasets obtained from [27] include a simulation of fuel injection into a combustion chamber (64^3), a CT scan of an engine ($256 \times 256 \times 128$), a CT scan of a lobster ($301 \times 324 \times 56$), a simulation of a homogeneous charge compression ignition (560^3), a rotational C-arm x-ray scan of the arteries of the right half of a human head (256^3), a CT scan of a Bonsai tree (256^3), and a CT scan of a carp fish ($256 \times 256 \times 512$).

5.2.1 Edge-Crossing Uncertainty

The isosurface uncertainty shown in Fig. 10 is based on our edge-crossing error estimation introduced in Sec. 4.1, and the difference between interpolation methods. The approximated isosurface uncertainty shown in the first column (Fig. 10a, Fig. 10f, Fig. 10k, Fig. 10q) is larger than the difference between interpolation methods shown in the remaining columns. The regions with high errors (red regions in the first column) correspond to the regions with large differences (uncertainty) between interpolation methods. For instance, in the fuel example, the black rectangle delineates a region of high error detected by our method in Fig. 10a, corresponding to the same region with significant differences between linear and higher-order methods in Fig. 10c and Fig. 10d. These results show that the edge-crossing error estimation in Sec. 4.1 identifies regions with large uncertainty in the context of practical datasets. In addition, the similarity observed demonstrates that our uncertainty estimation methods efficiently indicate the positions (red regions), where high-order interpolation can improve isosurface accuracy compared to linear interpolation. In the case of the Engine dataset, the improvements from linear in Fig. 10h, Fig. 10i, and Fig. 10j are considerably smaller than the approximated errors in Fig. 10f. These results indicate that using higher-order interpolation methods in the case of the engine dataset doesn't significantly improve the accuracy. For the fuel, combustion simulation, and lobster examples, the differences between cubic and WENO are smaller compared to the differences between linear and high-order interpolation methods, as shown in Fig. 10e, Fig. 10o, and Fig. 10t.

5.2.2 Hidden Features Comparison

The uncertain isosurface feature recovery using our proposed methods (Sec. 4.2) in the first, second, and fourth columns from the left of Fig. 11 improves the reliability of results. State-of-the-art techniques miss these important features, as shown in the third and fifth columns from left in Fig. 11, which can lead to less reliable data analysis. The targeted regions of interest are shown with the rectangles in the first column. The zoomed-in regions shown in the remaining columns correspond to the red rectangles. The second and fourth columns show a comparison between the standard MC



(a) Fuel Approximated Error (b) Max. Variation (c) L vs C (d) L v W (e) C vs W

(f) Fuel Approximated Error (g) Max. Variation (h) L vs C (i) L vs C (j) C vs W

(k) Fuel Approximated Error (l) Max. Variation (m) L vs C (n) L vs W (o) C vs W

(p) Fuel Approximated Error (q) Max. Variation (r) L vs C (s) L vs W (t) C vs W

Figure 10: Comparison of approximated error and different interpolation methods (Components C_1, C_3, C_4, and C_6). The first column corresponds to the approximated edge-crossing error. The second column shows the maximum variation. The remaining third, fourth, fifth, and sixth columns correspond to the comparison L vs. C, L vs. W, and C vs. W, respectively. These results show that using our methods in the first column from left effectively identifies regions with large errors and variations between linear and higher-order interpolation methods.

(opaque blue) and the hidden feature recovery method introduced in Sec. 4.2 (transparent orange). Showing the standard MC and hidden feature recovery in the framework enables a visual comparison of isosurface features. The third and fifth columns show results for sharp feature recovery based on dual contouring [26].

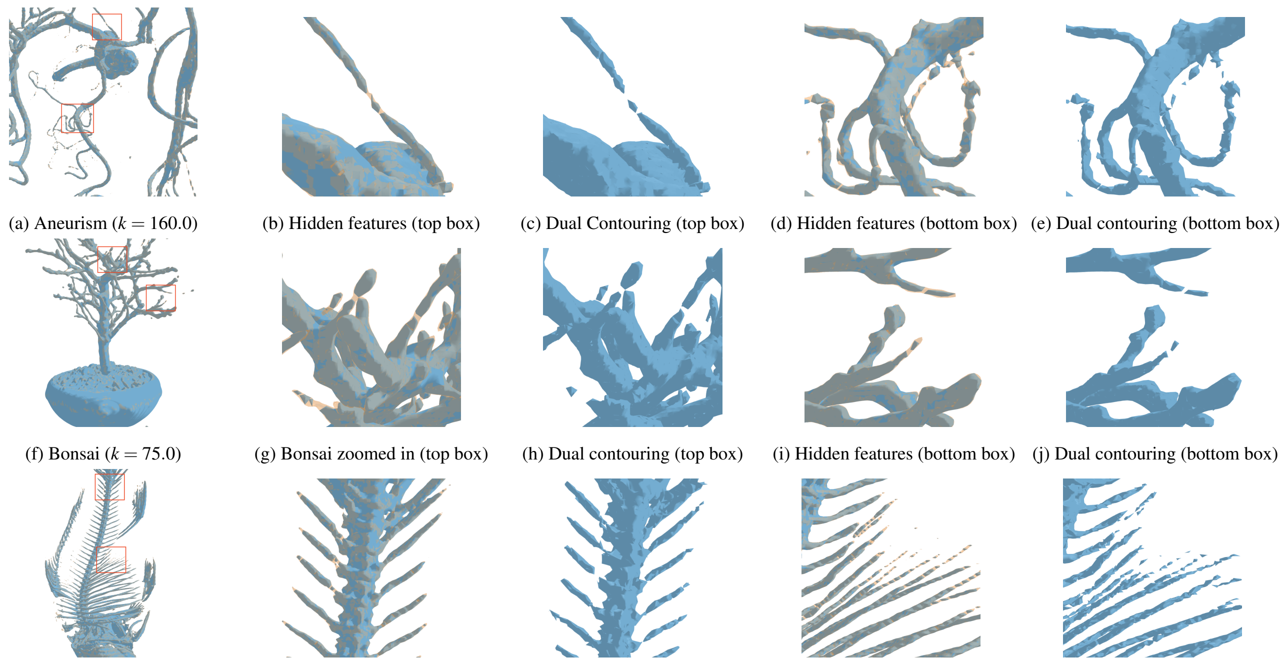
The zoomed-in results show that many broken pieces in the isosurface without hidden feature recovery shown with the opaque blue in the second and fourth columns (Fig. 11b, Fig. 11d, Fig. 11g, Fig. 11i, Fig. 11l, and Fig. 11n) are connected in the isosurface with hidden feature recovery shown in transparent orange in the same figures. The dual contouring method connects the broken pieces but introduces sharp corners and edges.

6 Discussion

We presented an integrated interactive visualization system that provides valuable insights into the uncertainties inherent in the MC algorithm with linear interpolation. Our error estimation method introduced in Sec. 4.1, provides a more accurate approximation of the edge-crossing errors compared to standard error approximation, as demonstrated in Fig. 2 and Fig. 3. The estimated error can be computed for any 3D volumetric data without the need for additional information, such as high-resolution data which underscores its broad applicability. In addition, the error estimation is computationally efficient compared to measured error and other sampling methods because the errors are directly calculated using the local neighborhood cells.

Figure 9 and Fig. 10 show examples of how the framework in Fig. 8 can be used to visualize edge-crossing errors and compare different interpolation methods. The results in these figures further demonstrate that the isosurface uncertainty estimation, shown in the first column of both figures, successfully identifies regions with large errors (red) and variations between linear and higher-order interpolation methods. The orange regions in Fig. 9 and Fig. 10 show the accuracy improvement from linear to higher order methods.

Moreover, we introduced a hidden feature reconstruction method in Sec. 4.2 that successfully identifies and reconstructs possible features that are ignored by the MC algorithms. The extraction of these features is based on fitting and refining the target cells using cubic interpolation. The polygon extraction at the boundary of the refined and unrefined cells causes cracks that are resolved using a similar approach in [29], illustrated in Fig. 6. we visualize the isosurfaces with (transparent orange) and without (opaque blue) to enable vi-


(a) Aneurism ($k = 160.0$) (b) Hidden features (top box) (c) Dual Contouring (top box) (d) Hidden features (bottom box) (e) Dual contouring (bottom box)

(f) Bonsai ($k = 75.0$) (g) Bonsai zoomed in (top box) (h) Dual contouring (top box) (i) Hidden features (bottom box) (j) Dual contouring (bottom box)

(k) Carp ($k = 1270.0$) (l) Hidden features (top box) (m) Dual contouring (top box) (n) Hidden features (bottom box) (o) Dual contouring (bottom box)

Figure 11: Comparison of isosurfaces from MC using our uncertain-feature recovery methods in C_6 (first, second, and fourth columns from left) vs. sharp feature recovery (third and fifth columns from left). The first column shows isosurfaces with selected regions indicated with red boxes. The second and fourth rows show a zoomed-in isosurface with (transparent orange) and without (blue) hidden feature recovery. The third and fifth columns show a zoomed-in isosurface from dual counting. The results indicate possible new interesting features and topological connections by semitransparent orange surfaces that are missed by the state-of-the-art MC feature extraction method.

sual comparison of two possible isosurfaces from MC with different features and topological structures, as shown in Fig. 1. Figure 11 shows that our method for hidden feature reconstruction leads to a smoother connection between broken pieces compared to dual contouring which introduces sharp edges and corners.

It is important to note that the techniques introduced have some limitations that we plan to address as this work continues. Even though we didn't observe these issues in the datasets used, the cubic, WENO, and other high-order interpolation methods may introduce undesirable oscillations. These oscillations may be reduced without comprising the possible hidden features with bounded interpolation methods [39, 40]. In addition, non-polynomial-based methods could provide more accurate error approximation for data that are generated from processes that don't rely on polynomials.

7 Conclusion

In this paper, we presented an efficient method for estimating and visualizing isosurface uncertainty from Marching Cubes (MC) algorithms. We introduce a closed-form approximation of edge-crossing error using polynomial interpolation and develop a technique for detecting and reconstructing uncertain hidden features. These approaches provide valuable insights into isosurface uncertainty and highlight the limitations of linear interpolation. In addition, we developed an integrated visualization system for the exploration and analysis of these uncertainties. Our examples and results demonstrate the effectiveness of our methods in estimating and visualizing isosurface uncertainty associated with linear interpolation. This work focused on error estimations for linear, cubic, and WENO interpolation methods. Extending the error analysis to include higher-order polynomial and non-polynomial interpolation techniques in future work could further improve error characterization across a broader range of interpolation models. Additionally, the current framework visualizes errors and potential hidden features separately. Integrating these into a unified visualization could provide a more comprehensive analysis of isosurface uncertainty.

8 Appendix

List of Synthetic examples used in this paper.

Tangle:

$$f_1(x,y,z) = x^4 + y^4 + z^4 - (x^2 + y^2 + z^2 - 0.4),\ x,y,z \in [-1,1]. \quad (12)$$

Torus:

$$f_2(x,y,z) = r_1 - \left(\sqrt{x^2+y^2}\right)^2 + z^2 - r_0^2, x,y,z \in [-1,1],$$
$$r_0 = 0.1,\ r_1 = 0.3. \quad (13)$$

Marschner and Lobb:

$$f_3(x,y,z) = \frac{\left(1 - sin(\frac{\pi z}{2}) + \alpha(1 + \rho_r(\sqrt{x^2+y^2})\right)}{2(1+\alpha)},\ x,y,z \in [-1,1]$$
$$\text{where: } \rho_r(r) = cos\left(2\pi f_M cos(\frac{\pi r}{2})\right),\ f_M = 6 \text{ and } \alpha = 0.25. \quad (14)$$

Teardrop:

$$f_4(x,y,z) = 0.5x^5 + 0.5x^4 - y^2 - z^2,\ x,y,z \in [-1,1] \quad (15)$$

Tubey:

$$f_5(x,y,z) = -3x^8 - 3y^8 - 2z^8 + 5x^4y^2z^2 + 3x^2y^4z^2$$
$$-4(x^3+y^3+z^3+1) + (x+y+z+1)^4 + 1,\ x,y,z \in [-3,3] \quad (16)$$

Acknowledgments

This work was partially supported by the Intel OneAPI CoE, the Intel Graphics and Visualization Institutes of XeLLENCE, the DOE Ab-initio Visualization for Innovative Science (AIVIS) grant 2428225, and the U.S. Department of Energy (DOE) RAPIDS-2 SciDAC project under contract number DE-AC0500OR22725.

References

[1] N. Aspert, D. Santa-Cruz, and T. Ebrahimi. Mesh: measuring errors between surfaces using the hausdorff distance. In *Proceedings. IEEE International Conference on Multimedia and Expo*, vol. 1, pp. 705–708 vol.1, 2002. doi: 10.1109/ICME.2002.1035879 2

[2] T. Athawale and A. Entezari. Uncertainty quantification in linear interpolation for isosurface extraction. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2723–2732, 2013. doi: 10.1109/TVCG.2013.208 2

[3] T. Athawale, E. Sakhaee, and A. Entezari. Isosurface visualization of data with nonparametric models for uncertainty. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):777–786, 2016. doi: 10.1109/TVCG.2015.2467958 2

[4] T. M. Athawale, S. Sane, and C. R. Johnson. Uncertainty visualization of the marching squares and marching cubes topology cases. In *2021 IEEE Visualization Conference (VIS)*, pp. 106–110, 2021. doi: 10.1109/VIS49827.2021.9623267 2

[5] B. Bagley, S. P. Sastry, and R. T. Whitaker. A marching-tetrahedra algorithm for feature-preserving meshing of piecewise-smooth implicit surfaces. *Procedia Engineering*, 163:162–174, 2016. 25th International Meshing Roundtable. doi: 10.1016/j.proeng.2016.11.042 2

[6] P. Bourke. Tubey, 2003. https://paulbourke.net/geometry/tubey/. 4

[7] K. Brodlie, R. Allendes Osorio, and A. Lopes. *A Review of Uncertainty in Data Visualization*, pp. 81–109. Springer London, London, 2012. doi: 10.1007/978-1-4471-2804-5_6 2

[8] E. Catmull and R. Rom. A class of local interpolating splines. In R. E. BARNHILL and R. F. RIESENFELD, eds., *Computer Aided Geometric Design*, pp. 317–326. Academic Press, 1974. doi: 10.1016/B978-0-12-079050-0.50020-5 2, 3

[9] Z. Chen, A. Tagliasacchi, T. Funkhouser, and H. Zhang. Neural dual contouring. *ACM Trans. Graph.*, 41(4), jul 2022. doi: 10.1145/3528223.3530108 2

[10] Z. Chen and H. Zhang. Neural marching cubes. *ACM Trans. Graph.*, 40(6), dec 2021. doi: 10.1145/3478513.3480518 2

[11] P. Cignoni, C. Rocchini, and R. Scopigno. Metro: Measuring Error on Simplified Surfaces. *Computer Graphics Forum*, 1998. doi: 10.1111/1467-8659.00236 2, 4, 6

[12] D. Cohen-Or, A. Kadosh, D. Levin, and R. Yagel. Smooth boundary surfaces from binary 3D datasets. In M. Chen, A. E. Kaufman, and R. Yagel, eds., *Volume Graphics*, pp. 71–78. Springer London, London, 2000. doi: 10.1007/978-1-4471-0737-8_4 2, 3

[13] B. Csébfalvi. Beyond trilinear interpolation: Higher quality for free. *ACM Trans. Graph.*, 38(4), jul 2019. doi: 10.1145/3306346.3323032 1, 2, 3

[14] C. A. Dietrich, C. E. Scheidegger, J. Schreiner, J. L. Comba, L. P. Nedel, and C. T. Silva. Edge transformations for improving mesh quality of marching cubes. *IEEE Transactions on Visualization and Computer Graphics*, 15(1):150–159, 2009. doi: 10.1109/TVCG.2008.60 2

[15] M. Edmunds, R. S. Laramee, G. Chen, N. Max, E. Zhang, and C. Ware. Surface-based flow visualization. *Computers and Graphics*, 36(8):974–990, 2012. Graphics Interaction Virtual Environments and Applications 2012. doi: 10.1016/j.cag.2012.07.006 1

[16] Y.-F. Feng, L.-Y. Shen, C.-M. Yuan, and X. Li. Deep shape representation with sharp feature preservation. *Computer-Aided Design*, 157:103468, 2023. doi: 10.1016/j.cad.2022.103468 2

[17] S. Fuhrmann, M. Kazhdan, and M. Goesele. Accurate isosurface interpolation with hermite data. In *2015 International Conference on 3D Vision*, pp. 256–263, 2015. doi: 10.1109/3DV.2015.36 1, 2, 3

[18] J. Gao, W. Chen, T. Xiang, A. Jacobson, M. McGuire, and S. Fidler. Learning deformable tetrahedral meshes for 3D reconstruction. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20. Curran Associates Inc., Red Hook, NY, USA, 2020. 2

[19] C. Gillmann, D. Saur, T. Wischgoll, and G. Scheuermann. Uncertainty-aware visualization in medical imaging - a survey. *Computer Graphics Forum*, 40(3):665–689, 2021. doi: 10.1111/cgf.14333 1

[20] M. Han, T. M. Athawale, D. Pugmire, and C. R. Johnson. Accelerated probabilistic marching cubes by deep learning for time-varying scalar ensembles. In *2022 IEEE Visualization and Visual Analytics (VIS)*, pp. 155–159, 2022. doi: 10.1109/VIS54862.2022.00040 2

[21] F. B. Hildebrand. *Introduction to numerical analysis*. Courier Corporation, 1987. 3, 5

[22] C.-C. Ho, Y.-H. Lu, H.-T. Lin, S.-H. Guan, S.-Y. Cho, R.-H. Liang, B.-Y. Chen, and M. Ouhyoung. Feature refinement strategy for extended marching cubes: Handling on dynamic nature of real-time sculpting application. In *2004 IEEE International Conference on Multimedia and Expo (ICME) (IEEE Cat. No.04TH8763)*, vol. 2, pp. 855–858 Vol.2, 2004. doi: 10.1109/ICME.2004.1394335 2

[23] C.-C. Ho, F.-C. Wu, B.-Y. Chen, Y.-Y. Chuang, and M. Ouhyoung. Cubical Marching Squares: Adaptive Feature Preserving Surface Extraction from Volume Data. *Computer Graphics Forum*, 2005. doi: 10.1111/j.1467-8659.2005.00879.x 2

[24] G.-S. Jiang and C.-W. Shu. Efficient implementation of weighted ENO schemes. *Journal of Computational Physics*, 126(1):202–228, 1996. doi: 10.1006/jcph.1996.0130 3

[25] C. Johnson. Top scientific visualization research problems. *IEEE Computer Graphics and Applications*, 24(4):13–17, 2004. doi: 10.1109/MCG.2004.20 2

[26] T. Ju, F. Losasso, S. Schaefer, and J. Warren. Dual contouring of hermite data. *ACM Trans. Graph.*, 21(3):339–346, jul 2002. doi: 10.1145/566654.566586 2, 8

[27] P. Klacansky. Open scivis datasets, December 2017. `https://klacansky.com/open-scivis-datasets/`. 7

[28] A. Knoll, Y. Hijazi, C. Hansen, I. Wald, and H. Hagen. Interactive ray tracing of arbitrary implicits with simd interval arithmetic. In *2007 IEEE Symposium on Interactive Ray Tracing*, pp. 11–18, 2007. doi: 10.1109/RT.2007.4342585 4

[29] L. P. Kobbelt, M. Botsch, U. Schwanecke, and H.-P. Seidel. Feature sensitive surface extraction from volume data. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '01, p. 57–66. Association for Computing Machinery, New York, NY, USA, 2001. doi: 10.1145/383259.383265 2, 8

[30] T.-Y. Lee and C.-H. Lin. Growing-cube isosurface extraction algorithm for medical volume data. *Computerized Medical Imaging and Graphics*, 25(5):405–415, 2001. doi: 10.1016/S0895-6111(00)00084-7 1

[31] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. *SIGGRAPH Comput. Graph.*, 21(4):163–169, aug 1987. doi: 10.1145/37402.37422 1, 3

[32] S. Marschner and R. Lobb. An evaluation of reconstruction filters for volume rendering. In *Proceedings Visualization '94*, pp. 100–107, 1994. doi: 10.1109/VISUAL.1994.346331 2, 3, 4

[33] D. P. Mitchell and A. N. Netravali. Reconstruction filters in computer-graphics. *SIGGRAPH Comput. Graph.*, 22(4):221–228, jun 1988. doi: 10.1145/378456.378514 2, 3

[34] T. Moller, R. Machiraju, K. Mueller, and R. Yagel. Classification and local error estimation of interpolation and derivative filters for volume rendering. In *Proceedings of 1996 Symposium on Volume Visualization*, pp. 71–78, 1996. doi: 10.1109/SVV.1996.558045 3

[35] T. Moller, R. Machiraju, K. Mueller, and R. Yagel. Evaluation and design of filters using a taylor series expansion. *IEEE Transactions on Visualization and Computer Graphics*, 3(2):184–199, 1997. doi: 10.1109/2945.597800 3

[36] T. Moller, K. Mueller, Y. Kurzion, R. Machiraju, and R. Yagel. Design of accurate and smooth filters for function and derivative reconstruction. In *IEEE Symposium on Volume Visualization (Cat. No.989EX300)*, pp. 143–151, 1998. doi: 10.1109/SVV.1998.729596 3

[37] M. Müller. Fast and robust tracking of fluid surfaces. In *Proceedings*

of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, SCA '09, p. 237–245. Association for Computing Machinery, New York, NY, USA, 2009. doi: 10.1145/1599470.1599501 1

[38] T. S. Newman and H. Yi. A survey of the marching cubes algorithm. *Computers and Graphics*, 30(5):854–879, 2006. doi: 10.1016/j.cag.2006.07.021 2

[39] T. A. Ouermi, R. M. Kirby, and M. Berzins. Eno-based high-order data-bounded and constrained positivity-preserving interpolation. *Numerical Algorithms*, 92(3):1517–1551, 2023. 9

[40] T. A. J. Ouermi, R. M. Kirby, and M. Berzins. Algorithm 1041: Hippis – a high-order positivity-preserving mapping software for structured meshes. *ACM Trans. Math. Softw.*, 50(1), mar 2024. doi: 10.1145/3632291 9

[41] K. Pöthkow and H.-C. Hege. Nonparametric models for uncertainty visualization. *Computer Graphics Forum*, 32(3pt2):131–140, 2013. doi: 10.1111/cgf.12100 2

[42] E. Remelli, A. Lukoianov, S. R. Richter, B. Guillard, T. Bagautdinov, P. Baque, and P. Fua. MeshSDF: Differentiable iso-surface extraction. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20. Curran Associates Inc., Red Hook, NY, USA, 2020. 2

[43] G. Ristovski, T. Preusser, H. K. Hahn, and L. Linsen. Uncertainty in medical visualization: Towards a taxonomy. *Computers and Graphics*, 39:60–73, 2014. doi: 10.1016/j.cag.2013.10.015 1

[44] S. Schlegel, N. Korn, and G. Scheuermann. On the interpolation of data with normally distributed uncertainty for visualization. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2305–2314, 2012. doi: 10.1109/TVCG.2012.249 2

[45] P. R. Spackman, M. J. Turner, J. J. McKinnon, S. K. Wolff, D. J. Grimwood, D. Jayatilaka, and M. A. Spackman. *CrystalExplorer*: a program for Hirshfeld surface analysis, visualization and quantitative analysis of molecular crystals. *Journal of Applied Crystallography*, 54(3):1006–1011, Jun 2021. doi: 10.1107/S1600576721002910 1

[46] H. Theisel. Exact isosurfaces for marching cubes. *Computer Graphics Forum*, 21(1):19–32, 2002. doi: 10.1111/1467-8659.00563 2, 3

[47] I. Wald, S. Zellmann, W. Usher, N. Morrical, U. Lang, and V. Pascucci. Ray tracing structured amr data using exabricks. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):625–634, 2021. doi: 10.1109/TVCG.2020.3030470 5

[48] Z. Wang, T. M. Athawale, K. Moreland, J. Chen, C. R. Johnson, and D. Pugmire. $FunMC^2$: A Filter for Uncertainty Visualization of Marching Cubes on Multi-Core Devices. In R. Bujack, D. Pugmire, and G. Reina, eds., *Eurographics Symposium on Parallel Graphics and Visualization*. The Eurographics Association, 2023. doi: 10.2312/pgv.20231081 2

[49] G. H. Weber, H. Childs, and J. S. Meredith. Efficient parallel extraction of crack-free isosurfaces from adaptive mesh refinement (amr) data. In *IEEE Symposium on Large Data Analysis and Visualization (LDAV)*, pp. 31–38, 2012. doi: 10.1109/LDAV.2012.6378973 5

[50] G. H. Weber, O. Kreylos, T. J. Ligocki, J. M. Shalf, H. Hagen, B. Hamann, and K. I. Joy. Extraction of crack-free isosurfaces from adaptive mesh refinement data. In D. S. Ebert, J. M. Favre, and R. Peikert, eds., *Data Visualization 2001*, pp. 25–34. Springer Vienna, Vienna, 2001. 5

[51] E. W. Weisstein. Cubic formula. *https://mathworld. wolfram. com/*, 2002. 3

[52] I. Wolf, M. Vetter, I. Wegner, T. Böttger, M. Nolden, M. Schöbinger, M. Hastenteufel, T. Kunert, and H.-P. Meinzer. The medical imaging interaction toolkit. *Medical Image Analysis*, 9(6):594–604, 2005. ITK. doi: 10.1016/j.media.2005.04.005 1

[53] Z. Yu, M. J. Holst, Y. Cheng, and J. McCammon. Feature-preserving adaptive mesh generation for molecular shape modeling and simulation. *Journal of Molecular Graphics and Modelling*, 26(8):1370–1380, 2008. doi: 10.1016/j.jmgm.2008.01.007 2

[54] Y.-T. Zhang and C.-W. Shu. Chapter 5 - ENO and WENO schemes. In R. Abgrall and C.-W. Shu, eds., *Handbook of Numerical Methods for Hyperbolic Problems*, vol. 17 of *Handbook of Numerical Analysis*, pp. 103–122. Elsevier, 2016. doi: 10.1016/bs.hna.2016.09.009 3