Shape Diameter Computation on Surface Meshes and A Review of Shape Regularization Methods in LevelSet Topology Optimization

by

Dhyey Bhavsar

Master of Science, University of California San Diego, 2019

Bachelor of Technology, Nirma University, 2017

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

of the requirements for the degree of

Master of Science

Department of Aerospace Engineering

2025

Committee Members:

Dr. Kurt Maute, Chair

Dr. John A. Evans

Dr. Maryam Shakiba

Bhavsar, Dhyey (M.S., Aerospace Engineering)

Shape Diameter Computation on Surface Meshes and A Review of Shape Regularization Methods in LevelSet Topology Optimization

Thesis directed by Dr. Kurt Maute

At any given point on a closed surface mesh, the Shape Diameter, a scalar function measuring the diameter of the largest fit circle or sphere, provides insight for shape analysis and, in our case, enforces a minimum feature size for a shape optimization problem. The shape diameter of a surface mesh and/or point clouds can be computed using the conical ray-cast approach, offset curve method, a point collision test, etc. Among various approaches available to calculate the shape diameter for both surface meshes, the current work implements a conical ray-casting-based approach due to its robustness. In the present work, the shape diameter represents a local thickness measure that is used to construct an optimization problem with a penalty term for minimum thickness violation. The current implementation integrates the ArborX library for constructing a bounding volume hierarchy with in-house optimization code for multiphysics optimization. Various numerical experiments highlight the sensitivity of current implementation with various parameters such as the number of rays within the cone and the angle of the cone. However, the improvement in terms of measuring local thickness just in the direction of normal compared to this current shape diameter approach reinforces the potential of using shape diameter for feature size control and regularization in shape optimization problems.

Dedication

To Mom, Dad, Masi and Batu.

Acknowledgements

I would like to thank my advisor, Dr. Kurt Maute, for advising me and answering my trivial questions in the course of this thesis. His guidance has greatly expanded my understanding of computational geometry and its application to problems in mechanics. I'd also like to thank Dr. John A. Evans and Dr. Maryam Shakiba for agreeing to serve on my committee. Furthermore, I would like to thank Nils Wunsch, Mokhtarzadeh Mohammad, Conor Rowan, Frieder Frank, and Brendan Chong for their continuous support throughout my time with the MORIS group and for helping me with various questions during this thesis. Finally, I'd like to thank my family (my parents, my aunt, my cousins, and my girlfriend) for always believing in me and pushing me to do the next best thing that I can be proud of.

Contents

Figures

Figure

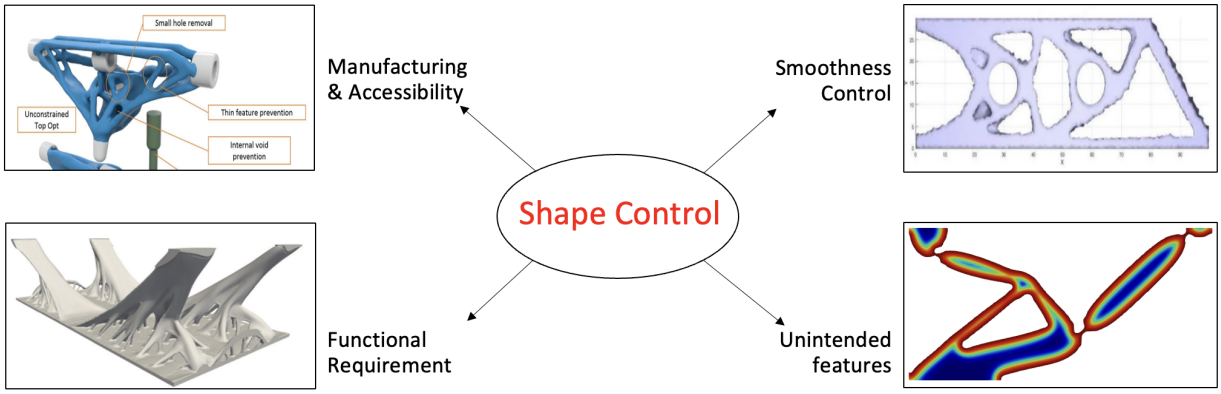
Chapter 1

Introduction

Shape control or shape regularization is aimed at achieving the desired outcome under constrained criteria for an evolving topology. It is a balance between parameters of shape control and design freedom. Shape regularization, within the topology optimization community, is studied through either the density method or the Level-set method each offering its advantages and disadvantages. Shape control can be broadly divided into four categories, (a) manufacturability, (b) smoothness control, (c) functional requirements, and (d) avoiding spurious numerical features. Figure 1.1 broadly categorizes shape control through various examples. Achieving shape regularization for various categories as mentioned here requires constructing a constrained optimization problem. Various approaches from manufacturing constraints to smoothness control are highlighted in Chapter 2. Furthermore, the smoothness control is broadly explored through either implicit thickness control by constructing some energy functional or through skeleton-based approaches or by using perimeter penalty-based approaches. The perimeter-area approach by embedding curvature into the area formulation is also studied for this thesis. Each of these methods has its advantages and limitations. The shape diameter broadly falling into an explicit shape regularization technique offers a new insight into the current optimization framework.

The shape diameter of a surface mesh represents a measure of volume through a neighborhood diameter. Mathematically, it is equivalent to the medial-axis transform in which a skeleton of a surface mesh is constructed. Shapira et al. [18] proposed the original idea of computing shape diameter for skeletonization. It suggests the construction of an inward-normal cone around every

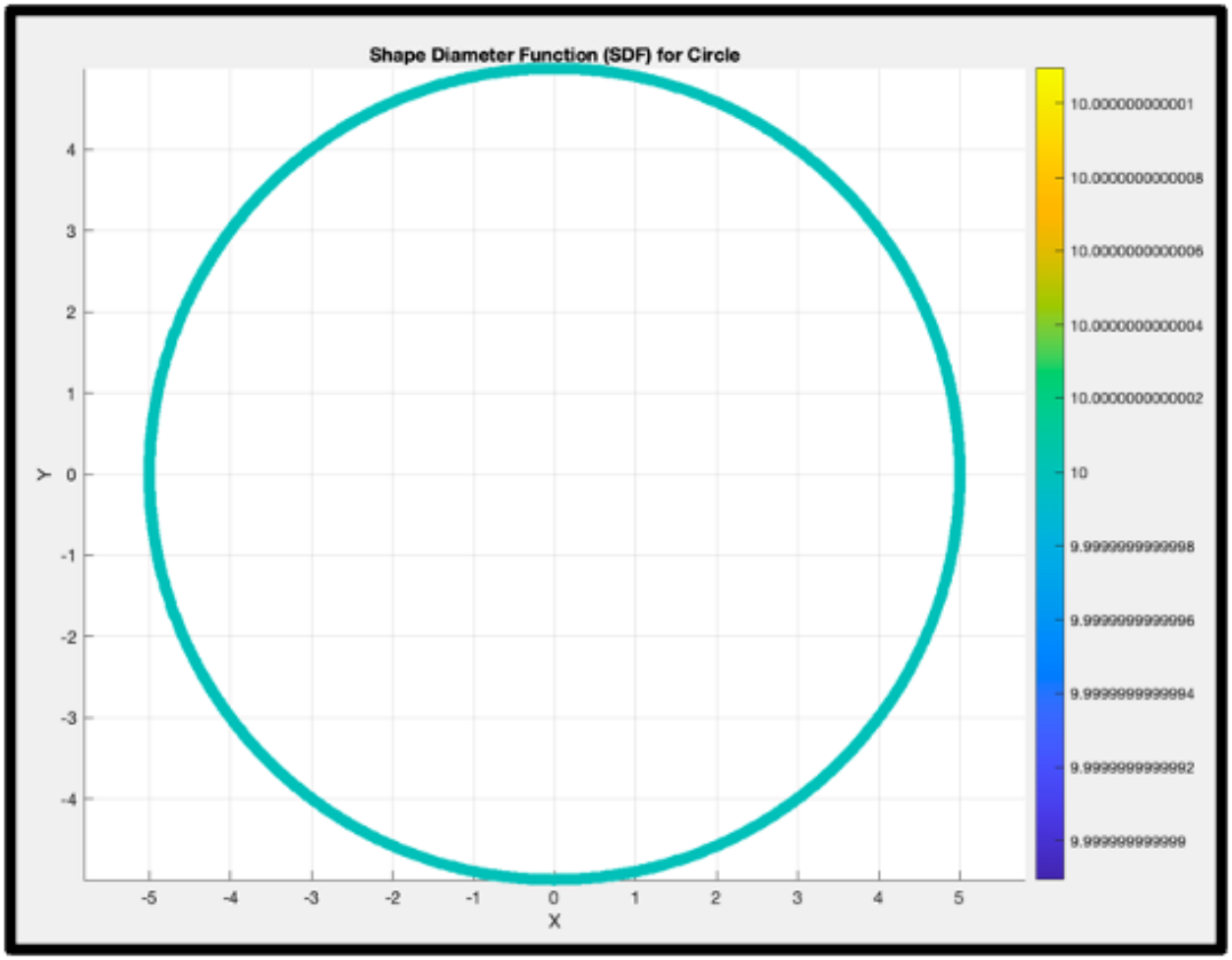
Figure 1.1: Various Aspects of Shape Control



point on the surface mesh from which the number of rays are cast such that they intersect the opposite side of the surface mesh. An inverse of the angle between the ray and the normal at the point is assigned as the weight associated with each ray. By using certain statistical tools the relevant rays are selected and a weighted sum of intersected distances provides a shape diameter measure at the given point. Consistent with the practice of embedding 3D objects into 2D surface mesh by the computer graphics community, the shape diameter encodes volumetric information of an object onto its boundary. To intuitively understand the shape diameter function, consider Figure 1.2 as shown below, which shows the shape diameter of a circle with a diameter of 5 units. For this circle, it can be easily said that the shape diameter of this circle at every point is its diameter. This simple idea, however, requires a lot of computational resources due to the need to cast multiple rays.

This original idea offers a new insight to the computational geometry community for shape analysis, but it comes with a few limitations the first being the massive need for parallelization. Therefore, alternative approaches to both CPU-based approaches, such as the offset curve by Chen et. al. [4] and the GPU architecture-based approach by Madaras et. al. [14] are being developed. Shape diameter with its pose-oblivious nature offers an advantage in many computational geometry applications. Although it suffers both in terms of accuracy and computational cost for a surface with

Figure 1.2: Shape Diameter Function of a Circle with Radius of 5 unit



rich geometric details, suitable methods such as averaging distances and offsetting methods open a wider domain of applications. Among the many approaches available to compute shape diameter as discussed in Chapter 2, the present work utilizes the original approach by Shapira et al. due to its robustness for implementation and handling a variety of geometric details with accuracy. The in-house research code, MORIS, has integrated the ArborX library by Lebrun-Grandie et. al. [11] for the construction of a bounding volume hierarchy to preselect potential intersection points for ray casting.

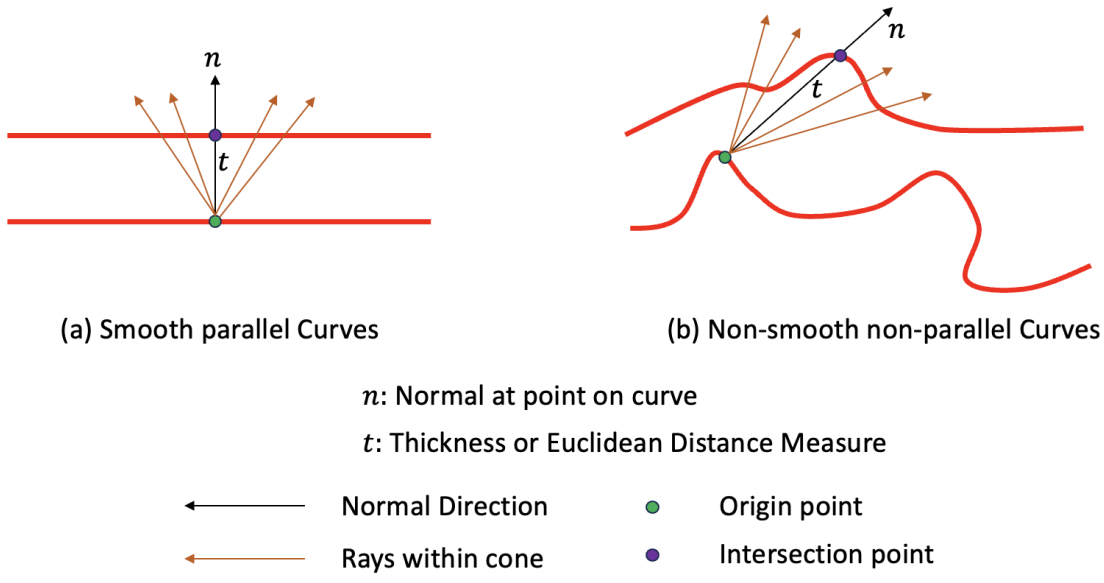
Shape regularization is a significant aspect in topology optimization to avoid undesired numerical artifacts. One of the problems studied extensively by the topology optimization community

is implementing a minimum feature size constraint such that the optimizer avoids creating features with a thickness smaller than what is intended. This is one of the important aspects of manufacturing limitations. Wang et al. [21] presented a curvature-based approach wherein the construction of a perimeter-based area and signed distance field-based area decides the violation of minimum thickness constraint. This approach, however, as explored in our numerical experiments, showed us certain limitations due to sensitivity to curvature and signed distance field. Various techniques such as Tikhonov-formulation-based curvature control [23], and least-square stabilization of curvature gradients, have been unsatisfactory in this attempt. Therefore, the central idea of this work is to construct an optimization problem with a constraint or penalty term invariant to topological changes. The volume of an object remains invariant under topological changes. Therefore, the selection of shape diameter as an indicator of volumetric information encoded in the surface mesh of the object offers a suitable alternative.

Although we have established a basic idea about the shape diameter formulation, there's no clear definition as to what makes a shape diameter measure different from Euclidean distance, other than its formulation. Figure 1.3 attempts to show a difference and the need for a measure like shape diameter. In case of smooth parallel curves and surfaces, the shape diameter is roughly equivalent to its Euclidean distance because in this case the shape diameter becomes a weighted average symmetrically distributed around a given normal direction. This yields the same result as computing the thickness measure along the direction of normal. However, when the surfaces are non-smooth and non-parallel, relying simply on the traditional thickness measure along the normal yields thickness values completely unrepresentative of their neighborhood. This leads to a scenario where the thickness measure, still representing a physical aspect of the volume, makes it hard for the optimizer to enforce the constraint efficiently. In addition to that, sudden changes in thickness values, as in the case of convex or concave curves and surfaces, create non-smooth gradients, leading to ineffective constraint enforcement. Therefore, the shape diameter measure, by opening a bigger viewing cone rather than a unidirectional Euclidean distance, offers an alternative that smoothly captures local neighborhood volumetric information. So, while there's a notable

discrepancy between thickness and the shape diameter for obvious cases, it is also important to create some kind of relationship between the "physical thickness" measure and the shape diameter at a given point from the constraint formulation standpoint. Details of this are presented in Chapter 3.

Figure 1.3: Difference in Thickness Measure for Various Curves



The next chapter will talk about various approaches for shape diameter construction in detail and the reason behind this work to implement the approach by Shapira et. al. The third chapter will talk about shape diameter computation in detail along with MORIS implementation. The fourth chapter will talk about an analytical aspect of sensitivity analysis followed by results and discussion in the fifth chapter.

Chapter 2

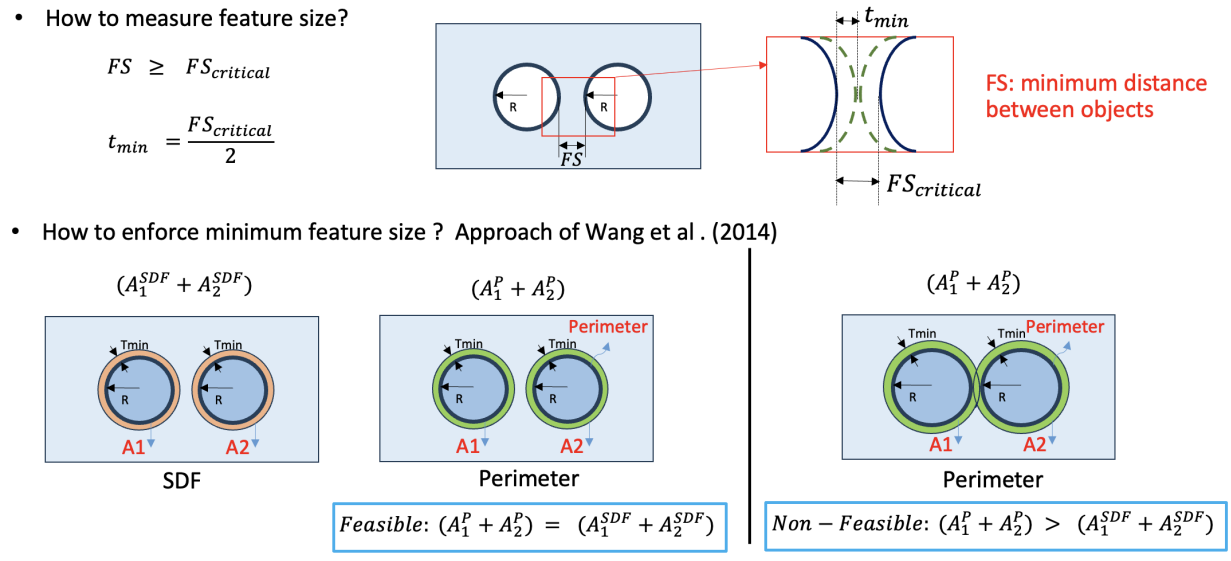
Literature Review

This chapter details various works on explicit and implicit thickness control with an eventual pivot to the computation of shape diameter for 2D and 3D surface meshes for various applications from segmentation and skeleton formation to a potential thickness control method for shape control. This chapter presents a comparative review discussing the accuracy of each method and the robustness of the implementation.

Shape regularization using thickness control can be broadly divided into explicit and implicit methods. In density-based formulations, various techniques have come up to control minimum feature size in the recent past. In Barrera et. al. [2], a minimum feature size control is achieved through a coupled density-LevelSet-based scheme by constructing dual penalty parameters. First to promote binary density distribution and second to regulate density field evolution in voids. In a density-based framework, the work by Guo et. al. [7] and Sigmund et. al. presents an explicit minimum length scale control. The work on overhang projection constraint for manufacturability by Gaynor et. al. is noteworthy. Despite, the strong community engagement from the density method community, these approaches mainly suffer due to inadequate shape representation and material interpolation. On the other hand, level-set methods despite being recent, offer advantages in terms of crisp geometry representation and smoother interface construction. Level-set approaches, despite being recent, still present strong arguments for feature size constraint formulations. Chen et. al. [3], proposed an approach by adding global quadratic energy functional into the objective which eliminates the development of small features. This approach, however, lacks explicit

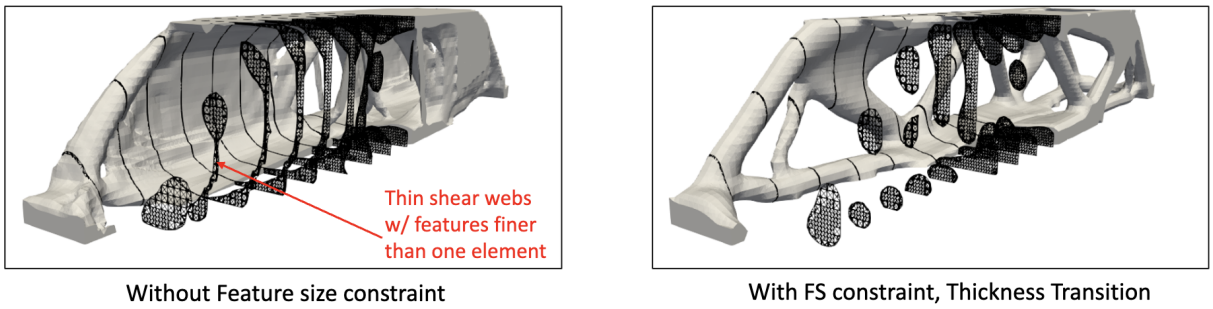
geometric representation in the form of providing the exact value of feature size. To overcome this, Guo et. al. proposed a signed distance field-based approach to impose minimum/maximum feature size. Liu et. al. [12], in their work on uniform thickness control proposed a scheme by adding thickness control functional with various weight factors into the objective function instead of explicitly constructing feature size constraint. This approach demonstrates the balance between structural performance and length scale control. Wang, Y. et. al. [21], in their paper on length scale control using level-sets demonstrated a novel approach by constructing offset curves of structural boundaries and level-set contour at the interface with a constant distance referring to the size of feature (minimum/maximum). Following this work a continuation strategy is developed to enforce minimum feature size for problems in 2D and 3D. It requires computing two area measures. The first one uses a signed distance field and the second area uses perimeter curvature information. By keeping the perimeter curvature area smaller than the signed distance field-based offset area, one can expect no overlap between thin regions as shown in Figure 2.1. The result of this work was presented at USNCCM17.

Figure 2.1: Schematic Diagram of Thickness Control Using Curvature Area Measure [21]



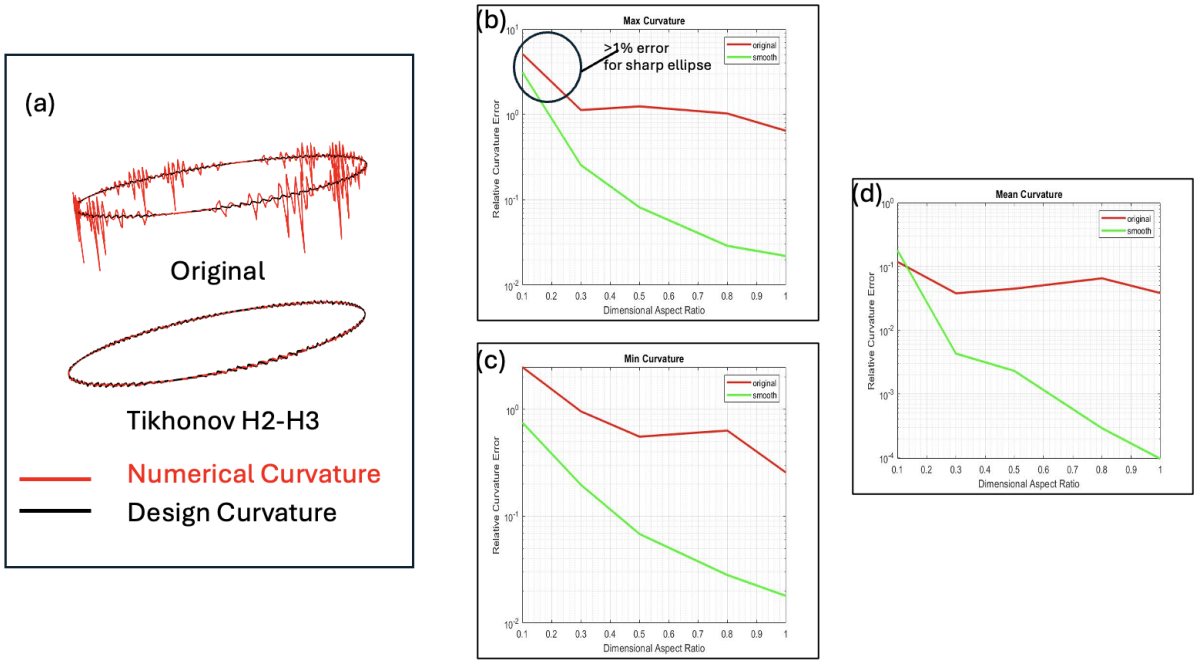
However, further exploration with this approach became limited due to sensitivity to cur-

Figure 2.2: Minimum Feature Size Enforcement on a 3D Simply Supported Beam



vature noise with certain problems. Among many approaches, a Tikhonov formulation [23] to stabilize the curvature by adding higher-order derivatives into the L^2 map was investigated. Figure 2.3 shows the curvature improvement for a circle with this approach, however, as the aspect ratio decreases such as for an extremely notched ellipse, the mean curvature error remains very high.

Figure 2.3: Tikhonov Formulation for Curvature Control (a) Curvature Stabilization for Circle, (b) Maximum Curvature Error, (c) Minimum Curvature Error, (d) Mean Curvature Error



In recent work, Zuo et al. [22], present an explicit thickness control by means of embedding maximum and minimum thickness constraints computed using Betti Number and Euler number which are topological invariants. Different approaches both for density and level-set methods require a deep understanding of problem-dependent parameters which makes them less suitable for a spectrum of problems.

Both shape diameter as a primary variable and the skeleton as a result of constructing shape diameter can be utilized for our problem to conduct shape regularization. Skeleton-based approaches by constructing Laplacian of the field are complex due to first construction of Laplacian and then extracting the skeleton from it. This skeleton which is a 1D line representation can then be used to apply thickness control through a maximum fit circle at each point on the skeleton. This approach is studied by Geiss et al. [6] in which the Laplacian of signed-distance field is constructed based on the work on heat method by Crane et al. [5] and then minimum feature size is achieved using the work of Guo et al. [7]. Due to the requirement of fine mesh resolution for accurate Laplacian construction, this method becomes computationally expensive. Therefore, efficient skeleton construction is well suited for the optimization problem that we aim to solve. Tierny et al. [19] present work on skeleton construction by topological and geometrical analysis. This work is divided into five segments, (a) feature point extraction, (b) geodesic distance function mapping, (c) Reeb Graph construction, (d) Constriction Approximation, and (e) Skeleton construction. Feature points on surface meshes are local minima and maxima using which geodesic distances are computed for all the points on surface mesh. Discrete Gaussian curvatures are computed using the contour lines from this geodesic distance mapping function. As shown in Figure 2.1, a constraint on this curvature with a low-pass filter yields a skeleton. This method, however, is sensitive to mesh resolution due to the requirement of accurately computing critical feature points.

Oscar et al. [1] introduced skeleton creation by successive mesh contraction approach which is insensitive to input mesh noise. It performs geometry contraction by successively solving discrete Laplacian equations followed by connectivity surgery which removes degenerated faces and edges. Figure 2.2 shows a schematic representation of this method.

Figure 2.4: Skeleton Construction Using Topological and Geometrical Analysis by Tierny et al. [19]

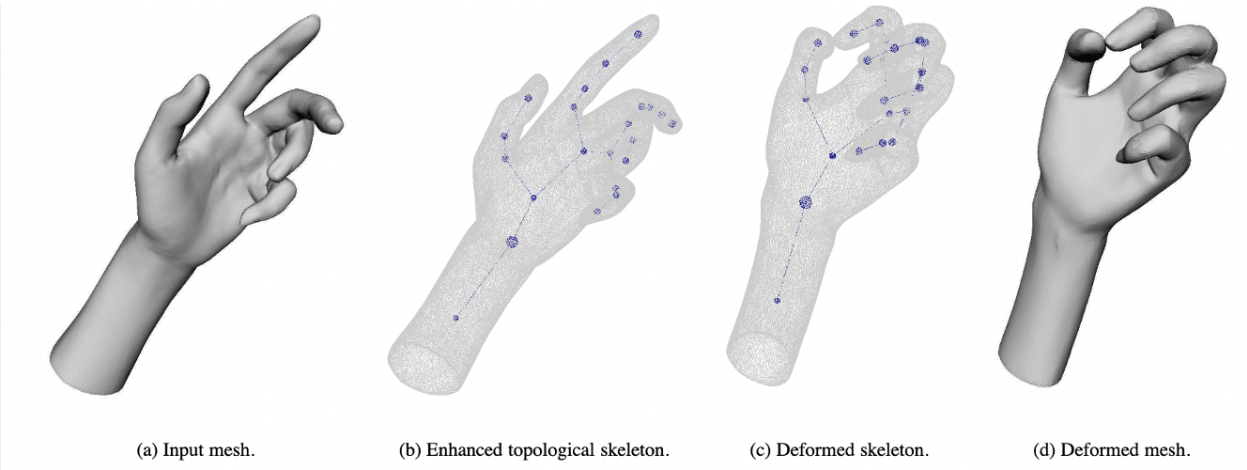
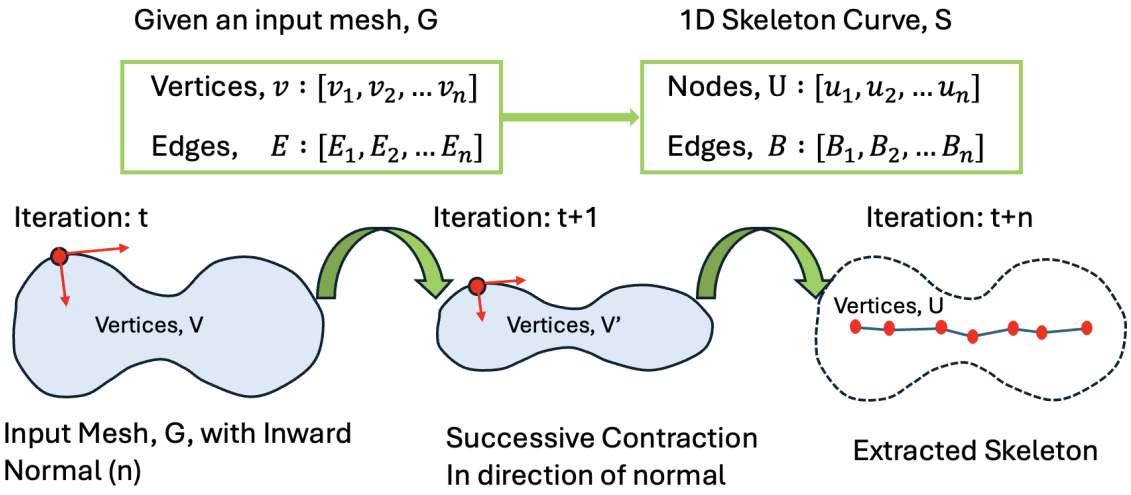
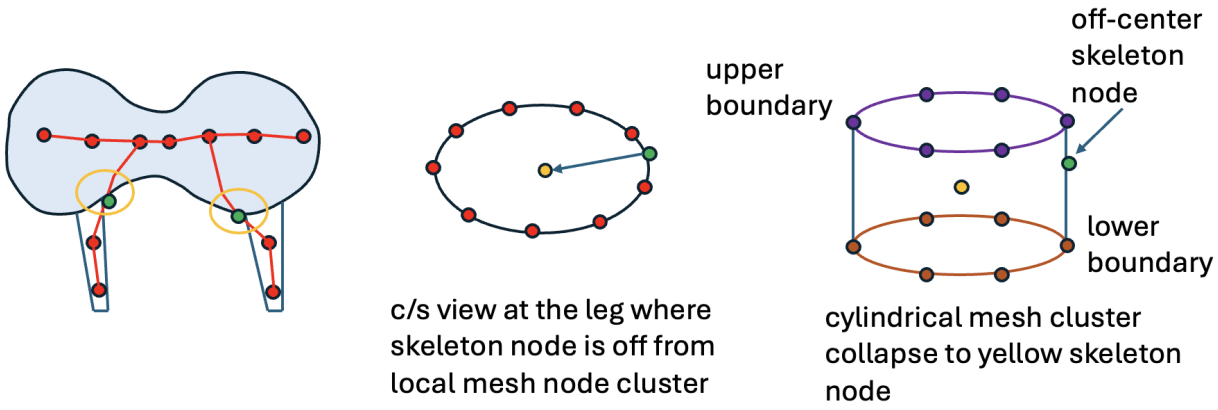


Figure 2.5: Schematic Representation of Mesh Contraction Method for Skeleton Extraction by Oscar et. al. [1]



At every iteration of solving the Laplacian, it checks whether the contracted geometry abstracts the original geometry well enough and adds the sampling and shape cost to the quadratic energy function during connectivity surgery. The quadratic energy function, a combination of contraction forces acting inward on the vertices of contracted geometry and attraction forces balanced such that contracted geometry abstracts the original geometry, is solved in a least-square setting.

Figure 2.6: Schematic Representation of Embedding Refinement by Oscar et. al. [1]



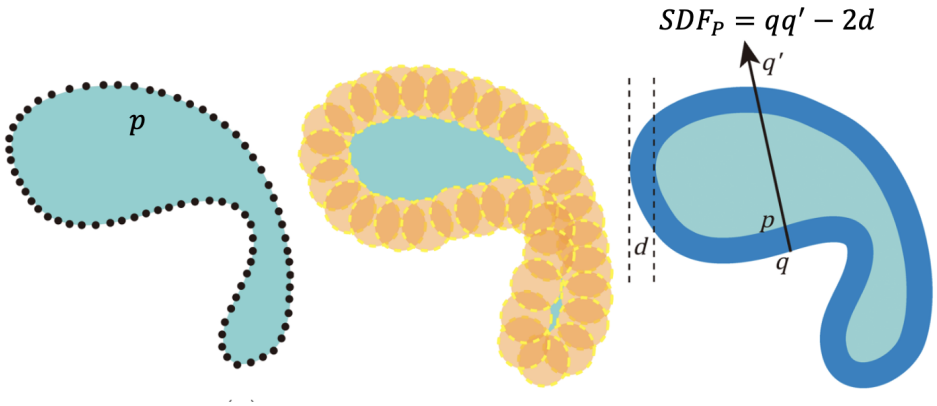
The final step of embedding refinement is performed for geometrically rich details such as the transition from thick to thin regions where the imbalance in contraction forces moves the node away from the expected skeleton. As shown in Figure 2.3, embedding refinement is a process of moving an off-center node in the skeleton to a potential center point in the mesh cluster by computing the average distances of the nodes on the upper boundary and lower boundary and subtracting the average of these two distances from the coordinate of the off-center node. Despite its advantage in terms of insensitivity to input mesh noise, pose-oblivious nature, and simple implementation it suffers from accuracy due to poor weights associated with contraction and attraction forces. Sometimes, due to inherence construction, it creates a zig-zag pattern of a skeleton which is unsuitable for further use.

Considering all these limitations for various approaches, a direct skeleton-based method to enforce shape regularization either becomes computationally expensive as shown by Geiss et al. [6] suffers from accuracy and robustness issues as shown in. However, skeleton construction through shape diameter function as achieved in [18] still possesses a potential for numerical study.

As discussed in Chapter 1, shape diameter is a representation of volumetric information which is topologically invariant, however, presents an opportunity to investigate its potential for

shape regularization. The shape diameter computation and its application have improved since the original work by Shapira et al. [18]. For example, the work of Chen et al. [4] requires only a single ray to be cast from each point corresponding to the surface mesh or point cloud. Considering the limitations of the work, mostly in terms of suitable cone angle for noisy geometry and scale-dependent problems, by Shapira et al. [18], an adaptive cone angle strategy was proposed by [17] for robust shape diameter computation.

Figure 2.7: Schematic Representation of Curve Offset Method by Chen et. al.[4]]

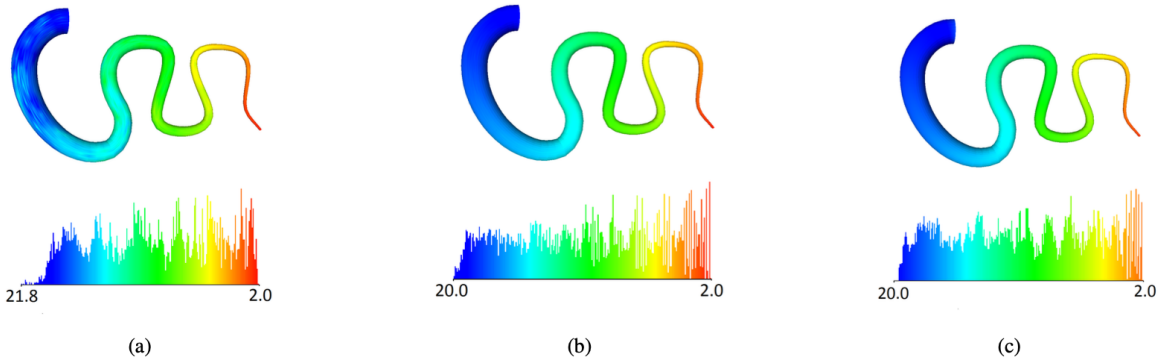


(a) Input mesh with vertices on boundary; (b) Bounding Box Generation; (c) SDF at a point P

Given a point on input surface mesh or point cloud, a shape diameter is computed by first constructing an offset curve to input mesh with certain constraints at specified offset distances. Then, the closest point is found for each point on the input surface mesh in the direction of its exterior normal. From this closest point on offset curves, a ray is cast and intersection distances on the opposite side are computed. The difference between this intersected distance and twice the offset distance provides the shape diameter at a given point on the input mesh. A very simple pictorial representation of this approach is shown in Figure 2.4. However, the complexity of this approach arises due to the need to construct offset curves. While constructing an offset curve for a closed and simple geometry is a trivial task, for open geometry or geometries with rich details such as extreme curvatures, it relies heavily on offset distances and therefore, does not yield satisfactory results.

This work, despite relying on accurate estimation of normal at each point, considers unit directional segment as an equivalent normal. MATLAB-based numerical investigation of this approach has shown the sensitivity of this method to geometric richness such as curvatures inherently due to the error in normal computation. The comparison of results is shown in chapter 5.

Figure 2.8: Comparison of Shape Diameter Values, (a) Shapira et al. [18] (b) OpenCL depth peeling, and (c) OpenCL ray casting]



(a) (b) (c)

A meshless approach for a point cloud developed by Huska et al. [9] requires evolving particle flow in the inward normal direction and at each time step a collision test is performed against any particle that comes in the direction of the normal. If the collision is detected, the distance traveled by colliding particles serves as the shape diameter value at that point.

All these approaches to compute shape diameter, thus far, consider implementation on CPU architecture. However, for an industry-grade problem, the solution using shape diameter construction commands a need for parallelization. The work by Madaras et al. [4] demonstrates the use of GPU architecture to compute shape diameter by parallel ray casting from each vertex. A prior hybrid approach by Carr et al. proposed generating rays on the CPU and performing ray-triangle intersection on the GPU. Despite being recent, a parallel ray-casting functionality offers an advantage in terms of computational cost with minimal deviation from the results obtained through traditional ray-casting approaches. Figure 2.5 highlights the difference between the shape diameter values of Shapira et al. [18] and various GPU-based schemes of Madaras et al. [4].

In the present study, a ray batching functionality is implemented to perform ray-triangle intersections which has improved timing performance by a considerable amount.

Chapter 3

Introduction to MORIS and Shape Diameter Implementation

This chapter details mainly two approaches that have been implemented in MORIS throughout this thesis, the first being (a) a perimeter-area-based approach for Level-set regularization, and (b) a shape-diameter-based approach for thickness control on surface meshes. While the first approach has shown its effectiveness both for 2D and 3D problems, it suffers if the geometry has noisy curvatures as this method explicitly utilizes the curvature into its area measure. The shape diameter approach while simple in theory possesses challenges both in terms of implementation and conducting sensitivity analysis.

3.1 An Introduction to Level-Set Topology Optimization

Level-set topology optimization was first introduced to model moving boundaries [16]. Due to boundary representation through an implicit level-set function, the geometric representation becomes crisp making this method suitable for topology optimization that requires dealing with evolving interfaces. Mathematically, a level-set function for a two-domain model with an interface is given by the following representation.

$$\phi(\mathbf{X}) < 0, \quad \forall\, \mathbf{X} \in \Omega_0^I,$$

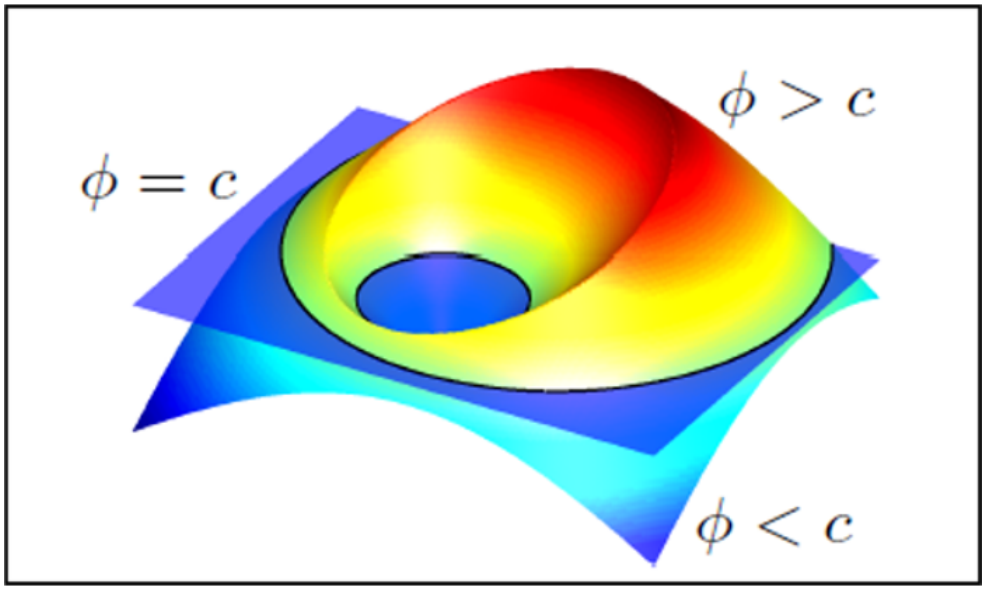
$$\phi(\mathbf{X}) > 0, \quad \forall\, \mathbf{X} \in \Omega_0^{II},$$

$$\phi(\mathbf{X}) = 0, \quad \forall\, \mathbf{X} \in \Gamma_0^{I,II}.$$

Where, the computational domain is represented as $\Omega_D = \Omega_0^I \cup \Omega_0^{II}$ and the interface between two domains is obtained as $\Gamma_0^{I,II}$.

The level-set method in the context of topology optimization is divided into implicit and explicit methods. The implicit level-set formulation requires solving the Hamilton-Jacobi equation at every design step the solution of which updates the material interface using nodal velocities as design variables. In the case of the explicit level-set method, the nodal level-set values act as a design variable which is updated using mathematical programming [15][13][10] [20] as shown in. A large dependency on initial design necessitates creating holes in the initial topology in the case of level-set topology optimization.

Figure 3.1: Schematic Representation of LevelSet Field on Two-Domain Model



3.2 Multi-physics Optimization Research and Innovation System (MORIS)

MORIS consists of mainly five modules, each with a specific role for a forward analysis and subsequent optimization process. HMR or hierarchical mesh refinement module creates a background Lagrange mesh or an interpolation mesh. GEN or geometry engine operates on user-defined geometries to construct explicit level-set fields. Each geometry can be assigned a material

phase. XTK is a mesh decomposition module that generates triangulation around the interfaces and creates sideset information. The triangulated mesh represents an integration mesh on which finite element problems are solved. FEM or finite element module is responsible for creating finite element problems depending on physics and boundary conditions associated with each material phase. OPT or optimization module is responsible for computing sensitivities and subsequently updating material interfaces as the optimizer iterates. In the current context, based on geometries defined in the geometry engine or GEN, the XTK module creates an integration mesh which is then passed onto the MTK module. This MTK module, in the current context, is responsible for creating surface meshes in 2D and 3D. This surface mesh consisting of nodes and nodal connectivity information is utilized for the computation of shape diameter.

3.3 An Explicit Thickness Control Using Shape Diameter Computation

In the current framework, the shape diameter is computed using the work of Shapira et al. [18] due to various advantages discussed in earlier chapters. In this context, shape diameter represents an explicit thickness measure at each vertex of a surface mesh.

3.3.1 Normal Computation

For two-dimensional surface meshes, the normal at a given vertex is computed using a weighted average of normals at adjacent facets or edges. In the current framework, the length of facets is taken as a weight associated with each normal. Figure 3.2 illustrates a schematic representation of normal computation for 2D surface mesh. For 3D surface meshes, the normal at a vertex is computed using normals of facets or cells and the area associated with corresponding cell as shown in Figure 3.3.

$$n_p = \frac{\sum_{i=1}^{5} n_i \, l_i}{\text{norm}\left(\sum_{i=1}^{5} n_i \, l_i\right)} \tag{3.2}$$

Where l_i is a length of an edge or facet and n_i is a facet normal.

Figure 3.2: Schematic Representation of Normal Computation at a Vertex in 2D

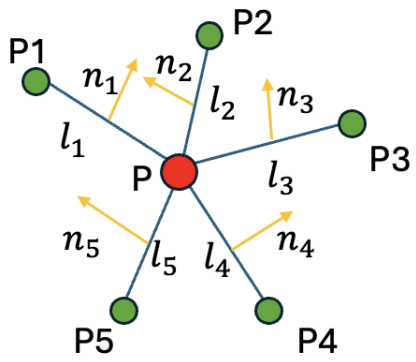
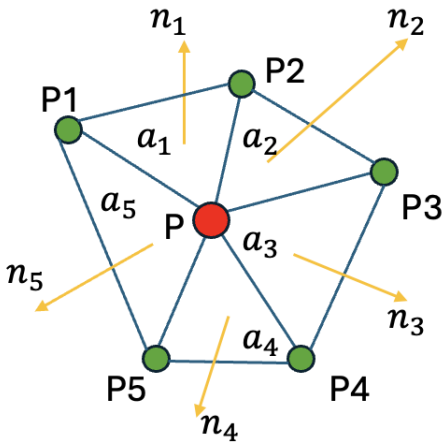


Figure 3.3: Schematic Representation of Normal Computation at a Vertex in 3D



In the case of 3D, equation 3.3 shows the trivial normal computation.

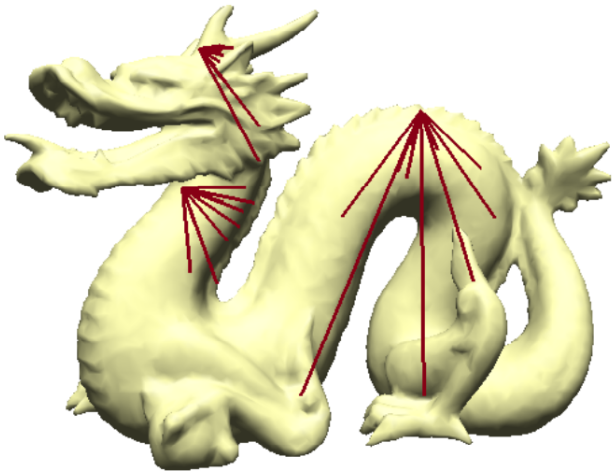
$$n_p = \frac{\sum_{i=1}^{5} n_i \, a_i}{\text{norm}\left(\sum_{i=1}^{5} n_i \, a_i\right)} \tag{3.3}$$

Where a_i is an area of a facet and n_i is a facet normal.

3.3.2 Construction of an Inward Cone

Construction of an inward cone for a two-dimensional problem is a trivial operation of creating a triangle with a certain angle aligned with a normal at a given vertex. Figure 3.4 shows a schematic representation of a normal aligned inward cone within which rays are cast. Then, rays within this triangle are computed by multiplying the two-dimensional rotation matrix with the direction of normal. For a three-dimensional problem, a slightly more rigorous process is required due to handling polar and azimuthal directions. Rays are equally distributed within the cone such that for a specified cone angle it covers most aspects of geometry in case of noisy details.

Figure 3.4: Schematic Representation of Inward Cone Construction and Ray Casting [14]



An inverse of the angle between a ray and the normal at a vertex is considered as a weight associated with each ray. In this context, it is important to highlight the sensitivity of this method to an extremely fine distribution of angles within a cone. The angle between the closest ray to the normal and the normal gets smaller, and the weight associated with that ray will explode yielding statistically skewed results at that vertex. Shapira et al. [18] therefore use a statistical screening of rays based on standard deviation criteria to remove outliers. A ray is considered an outlier if the difference between the intersected distance of that ray and the median of all distances falls beyond

one standard deviation. At a given point p for a ray r_1 for an intersected distance d_1^p,

$$(d_1^p - median(d_i^p) > 1.0 * \sigma \tag{3.4}$$

Where σ is a standard deviation computed based on distances from all rays d_i^p at a given vertex.

This is also advantageous for non-watertight meshes. However, as highlighted in [14] for some shapes this approach yields counter-intuitive results. A better strategy for constructing adaptive cones [17] can be implemented depending on the complexity of the problems. For the problems considered in this thesis, a static cone with criteria described by Shapira et al. [18] appears to be satisfactory.

The sensitivity associated with the smaller angle is numerically studied and a minimum unit angle criteria is set in the current implementation. Although Shapira et al. [18] suggested insensitivity of shape diameter values with respect to cone angle and number of rays, in our numerical experiments, we have found that for certain noisy details, the cone angle and number of rays are extremely important. The comparative study on various cone angles and number of rays along with mesh discretization is provided in Chapter 5. Our experiments suggest that a moderate cone angle (30 degrees to 60 degrees) with a denser distribution of rays (roughly 100 rays per vertex) generally yields satisfactory results. This implementation is verified at each successive step against results obtained through a MATLAB based implementation to avoid any errors. A comparative study on both MATLAB and MORIS based results is provided in Chapter 5. It is also important to highlight the importance of efficiently storing the information associated with each vertex in terms of relevant rays and their weights as it will be required later to compute sensitivities.

3.3.3 Ray Facet Intersection

After constructing a series of rays within a triangle (2D) or cone (3D), the ArborX library [11] is called to preselect potential intersection facets associated with each ray. The list of candidate facets obtained from the ArborX function is then passed onto the Moller-Trumbore algorithm to

find intersection points. Different cases such as a ray and 1D-edge, a ray and 2D triangle, and a ray and 2D-quadrilateral are implemented to compute intersection for different 2D and 3D cases. In the case of a ray and 2D-quadrilateral intersection, the 2D-quadrilateral is divided into two triangles, and the intersection of the ray is checked with each triangle. Please see the below algorithm that computes ray-triangle intersection using the Moller-Trumbore procedure.

(1) **Inputs:**

- **Ray:** Defined as $\mathbf{R}(t) = \mathbf{O} + t\mathbf{D}$, where:
  - $\mathbf{O}$ is the ray origin.
  - $\mathbf{D}$ is the ray direction (normalized).
  - t is the parameter along the ray.
- **Triangle:** Defined by its three vertices $\mathbf{V}_0$, $\mathbf{V}_1$, and $\mathbf{V}_2$.

(2) **Step 1: Compute Triangle Edges**

$$\mathbf{E}_1 = \mathbf{V}_1 - \mathbf{V}_0, \quad \mathbf{E}_2 = \mathbf{V}_2 - \mathbf{V}_0$$

(3) **Step 2: Compute Determinant**

- Compute the cross product:

$$\mathbf{P} = \mathbf{D} \times \mathbf{E}_2$$

- Compute the determinant:

$$\det = \mathbf{E}_1 \cdot \mathbf{P}$$

- If $|\det| < \epsilon$, the ray is parallel to the triangle, and there is no intersection.

(4) **Step 3: Compute Parameter** u

- Compute the vector from the triangle vertex $\mathbf{V}_0$ to the ray origin:

$$\mathbf{T} = \mathbf{O} - \mathbf{V}_0$$

- Compute the parameter:

$$u = \frac{\mathbf{T} \cdot \mathbf{P}}{\det}$$

- If $u < 0$ or $u > 1$, the intersection lies outside the triangle.

(5) **Step 4: Compute Parameter** v

- Compute the cross product:

$$\mathbf{Q} = \mathbf{T} \times \mathbf{E}_1$$

- Compute the parameter:

$$v = \frac{\mathbf{D} \cdot \mathbf{Q}}{\det}$$

- If $v < 0$ or $u + v > 1$, the intersection lies outside the triangle.

(6) **Step 5: Compute Parameter** t

$$t = \frac{\mathbf{E}_2 \cdot \mathbf{Q}}{\det}$$

- If $t < 0$, the intersection lies behind the ray origin.
- If $t \geq 0$, the ray intersects the triangle.

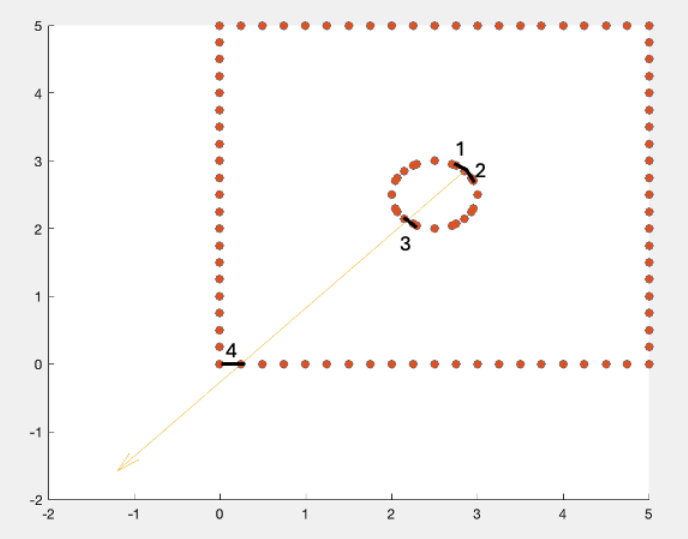
(7) **Step 6: Compute Intersection Point (if applicable)**

$$\mathbf{P}_{\text{intersection}} = \mathbf{O} + t\mathbf{D}$$

Based on the ArborX library [11] which returns a vector of potential candidate facets for intersection, the Moller-Trumbore algorithm computes the intersected distances for each candidate facet. The adjacent facet to the point from where the ray is originated removed based on a parallel facet check. For an interesting case, where the ArborX returns two facets in the direction of a ray, one which is closer while the other is further, the final shape diameter computation logic considers the closest distance as it is the most relevant case. However, it is necessary to remove those close to zero distances first since the logic takes the minimum value from the final vector. The figure below shows this particular case as it was noticed during implementation. As shown, the ArborX

library [11] returns candidate facets 1, 2, 3, and 4. The moller-trumbore algorithm removes facets 1 and 2 based on origin point criteria and computes the distance for facets 3 and 4. Now, for an inward-looking cone within the circle, it is necessary to have the distance corresponding to facet 3 instead of facet 4. Therefore, it is implemented to consider the closest distance between these two.

Figure 3.5: Relevant Case for Ray-Facet Intersection



3.3.4 Shape Diameter Computation

After constructing a data structure of vertex and the relevant ray weights and distances combination, the following equation is used to compute shape diameter. This equation represents a weighted sum of relevant distances. This approach inherently encodes neighborhood details while flattening out noisy details in many cases. The shape diameter at a given point p is computed as,

$$SDF(P) = \frac{\sum_{i=1}^{n} w_i^p d_i^p}{\sum_{i=1}^{n} w_i^p}$$

Where $w_i^p = 1/\theta_i^p$ is a weight associated with each relevant ray and d_i^p is an intersected distance with each relevant ray. For the purpose of this thesis, the computation of this distance

measure is enough. However, Shapira et al. [18] proposes a normalized logarithmic equation to create efficient segmentation and segregate various geometric details into groups.

$$N(P) = \log\left(\frac{SDF(P) - \min(SDF)}{\max(SDF) - \min(SDF)} \cdot \alpha + 1\right) / \log(\alpha + 1)$$

Where α is a constant with a value of 4 suggested based on a numerical experiment by Shapira et al. [18]. Minimum and maximum values of shape diameter are extracted after computing shape diameter values at all the nodes making it slightly expensive.

3.3.5 Shape Diameter Relation with Physical Thickness

While it's impossible to derive an analytical relationship between the unidirectional thickness measure and the multi-directional statistically quantified shape diameter, it is important to look at different scenarios where the minimum thickness constraint can be enforced through a constraint on the shape diameter value. In the absence of any analytical relation, rigorous analysis of the values of shape diameter for the entire surface mesh is required. To that extent, this thesis proposes evaluating shape diameter values using the cone approach and respective physical thickness as Euclidean distance at all the points on the surface mesh and then binning them based on higher, equal, and lower criteria compared with physical thickness values at respective points. Now, the following criteria can be considered to apply the minimum thickness constraint. (1) For the shape diameter values in equal or lower bins, the minimum value of shape diameter is set to the value of minimum physical thickness. (2) For the shape diameter values in the higher bin, a more rigorous unit test is required to calibrate the shape diameter value against the physical thickness measure. A factor α corresponding to penalty amplification can be introduced and multiplied by the shape diameter penalty term in the optimization framework. Numerical calibration of α needs to be performed using representative cases to understand the range of values that shape diameter occupies against its corresponding physical thickness measure.

Following the general criteria can be established for all points on surface mesh,

$\forall\, p \in S$,

(a) When the shape diameter value at node p is less than or equal to the corresponding thickness

If $SDF_p \leq t_p \Rightarrow SDF_{min} \leq t_{min}$

Therefore, by setting $SDF_{\min} = t_{\min}$; minimum requirement on physical thickness will be implicitly enforced.

(b) When shape diameter value at node p is greater than the corresponding thickness

If $SDF_p > t_p$

Required: (1) $t_p \geq t_{min} \Rightarrow t_p = \alpha_1 . t_{min}$; where $\alpha_1 \geq 1.0$

(2) $SDF_p \geq SDF_{min} \Rightarrow SDF_p = \alpha_2 . SDF_{min}$; where $\alpha_2 \geq 1.0$

By putting these required conditions into the original inequality,

$$SDF_{\min} > \frac{\alpha_1}{\alpha_2} t_{\min}$$

Conservatively, α_1 and α_2 can be computed as,

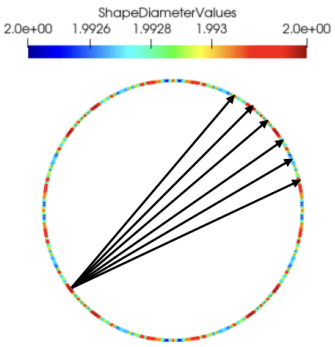
$$\alpha_1 = \frac{\max(t_p)}{t_{\min}} \quad \& \quad \alpha_2 = \frac{\min(SDF_p)}{t_{\min}}$$

A better approach to effectively enforce thickness control through minimum shape diameter value would be by doing a quantitative study on how α_1 and α_2 vary for a variety of geometrical cases within a given problem and calibrating it accordingly.

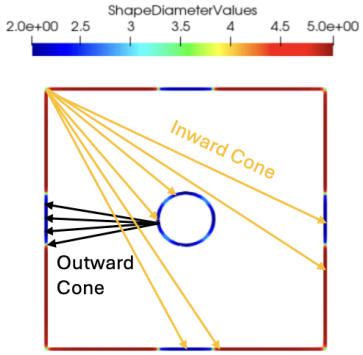
3.4 Relevant MORIS Implementation

The first step for this implementation to work is passing relevant sideset information such that the correct surface meshes are created. At present, based on surface mesh creation, it can compute shape diameter explicitly from a computer graphics standpoint or it can compute shape diameter as a thickness measure for a geometry embedded into the computational domain. Figure 3.5 shows that capability.

Figure 3.6: Closed and Hollow Surface Mesh Shape Diameter Computation



(a) Shape Diameter Computed using Inward Cone for a circular surface mesh



(b) Shape Diameter or "Local Thickness Measure" Computed using Inward/Outward Cone for a circle embedded into computational domain

After computing the shape diameter, it is necessary to efficiently store the values associated with each vertex. Now, since this entire computation is performed on integration mesh, it is required to store the shape diameter values as nodal coefficients that can be used to interpolate and compute the shape diameter on Gauss points in the Finite Element setting. As shown in Figure 3.7, to interpolate the shape diameter values at a Gauss point of a linear Lagrange bar, it is necessary to have a relevant nodal coefficient, computed as nodal shape diameter values from surface mesh, set on the correct field interpolator. To compute, the shape diameter values at any Gauss point,

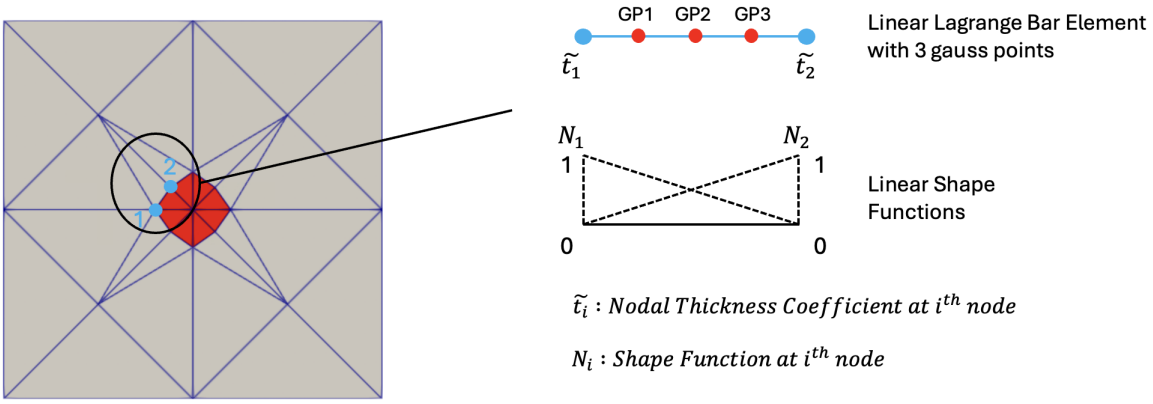
$$t_{gp_i} = w_{gp_i} \begin{bmatrix} N_1 & N_2 \end{bmatrix} \begin{bmatrix} \tilde{t}_1 \\ \tilde{t}_2 \end{bmatrix} \tag{3.7}$$

Where, w_{GP_i} is the weight on i^th gauss point.

$N_1 = 1/2(1-\xi)$ and $N_2 = 1/2(1+\xi)$ are the parametric shape functions on node-1 and node-2.

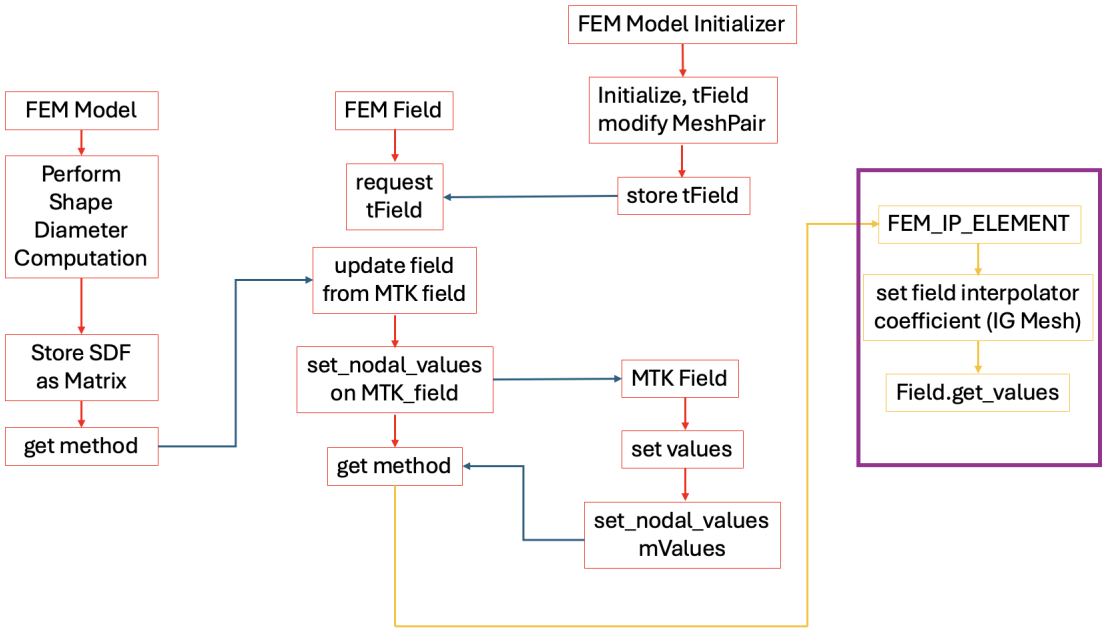
A field interpolator specifically designed to operate only on integration mesh in addition to a prior field interpolator operating on interpolation mesh is implemented. A brief workflow

Figure 3.7: Shape Diameter Evaluation at Gauss Point



highlighting necessary changes such that the shape diameter values are set on a field of the FEM module is shown in the figure below. It is necessary to store the values of shape diameter as a FEM field as all the later computations happen within the FEM module of MORIS.

Figure 3.8: MORIS workflow to store Shape Diameter Values Nodally



To summarize this chapter, it introduces the Levelset topology optimization in which we use the explicit method meaning the nodal level set values are design variables in the updating scheme. To compute shape diameter, the current thesis adopts the method presented by Shapira et al. [18]. The relevant implementation details within MORIS are also presented to highlight the current capabilities.

Chapter 4

Sensitivity Analysis

For the scope of this thesis, we intend to use the shape diameter values computed in the earlier section in the form of an integrated quantity (IQI) which can then be used in the optimization problem as a constraint or penalty term. This integration happens over the boundary nodes or sidesets. For that purpose, the minimum thickness penalty function is formulated below,

$$IQI = \int_{\Gamma} f(t)\, d\Gamma = \int_{\Gamma} \left(\frac{t}{t_{\min}} - 1 \right)^{n} d\Gamma \tag{4.1}$$

In finite-element setting, this integral is computed using Gauss quadrature as follows,

$$IQI = \sum_{p} w_{gp} \left(\frac{t_{gp}}{t_{\min}} - 1 \right)^{n} \tag{4.2}$$

Where,

w_{gp}: Weight at Gauss Point

t_{gp}: Shape Diameter at Gauss Point

Now, as we have highlighted in section 3.1 about Level-Set topology optimization, the nodal level-set values act as design variables that are abstract (ADVs). These abstract design variables, which are B-spline coefficients corresponding to the level-set field, are represented by s. Therefore, in sensitivity analysis, the goal is to compute the derivative of the IQI term with respect to these design variables. However, since there's no explicit dependency of IQI terms with respect to design variables, we only need to consider the implicit part. For the implicit part, the IQI depends on

thickness and the thickness values as shown in section 3.3.4 depend on physical nodal coordinates x_Γ of sidesets. Therefore,

$$\frac{\partial IQI}{\partial s} = \frac{dIQI}{ds}^{explicit} + \frac{\partial IQI}{\partial x_\Gamma}\frac{\partial x_\Gamma}{\partial s} \tag{4.3}$$

Since there's no explicit dependency,

$$\frac{\partial IQI}{\partial s} = \frac{\partial IQI}{\partial x_\Gamma}\frac{\partial x_\Gamma}{\partial s} \tag{4.4}$$

Now, computing the first part of the implicit term, that is, $\frac{\partial IQI}{\partial x_\Gamma}$ is a trivial task of further looking at IQI as a function of thickness measure,

$$\frac{\partial IQI}{\partial x_\Gamma} = \frac{\partial IQI}{\partial t}\frac{\partial t}{\partial x_\Gamma} = \sum_{gp}\left[w_{gp}\frac{\partial}{\partial t}\left(\frac{t_{gp}}{t_{min}} - 1\right)^n \frac{\partial t_{gp}}{\partial x_\Gamma}\right], \tag{4.5}$$

Where,

$$\frac{\partial}{\partial t}\left(\frac{t_{gp}}{t_{min}} - 1\right)^n = n\left(\frac{t_{gp}}{t_{min}} - 1\right)^{n-1}\frac{1}{t_{min}} \tag{4.6}$$

Now, to compute, the second term of the implicit formulation, we need to understand how the thickness value at a particular node is dependent on different nodes on the sideset. In a finite element discretized setting, thickness at a Gauss point depends on the thickness coefficients of the basis node. Therefore, the second implicit term can be written as,

$$\frac{\partial t_{gp}}{\partial x_\Gamma} = \frac{\partial}{\partial x_\Gamma}\left(\sum_i N_i\,\tilde{t_i}\right) = \sum_i N_i\frac{\partial \tilde{t_i}}{\partial x_\Gamma} \tag{4.7}$$

Where,

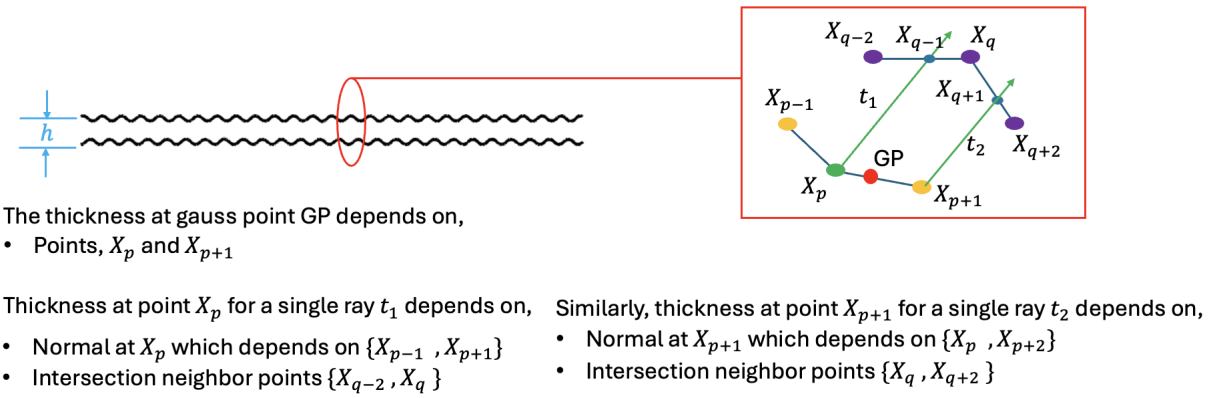
i: Number of nodes in a the discretized element

N_i : Finite element basis function at i^{th} node

$\tilde{t_i}$: Shape diameter coefficient at i^{th} node

To compute the last derivative, $\frac{\partial \tilde{t_i}}{\partial x_\Gamma}$, it is necessary to understand how this thickness value at the basis node depends on other nodes x_Γ. For simplicity, let's consider a single ray within a cone system as shown in Figure 4.1.

Figure 4.1: Single Ray System for Sensitivity Analysis



The shape diameter or thickness value at node X_p depends on nodes $(X_p, X_{p-1}, X_{p+1}, X_{q-2}, X_q)$. Therefore, we need to compute the sensitivity of thickness at node X_p with respect to all these nodes. For problems in 2D, the derivatives are computed for the x and y directions, and for 3D problems, the derivatives are computed for the x, y, and z directions.

For this chapter, the following analytical expressions in 2D are derived,

Following terms based on nodal positions are needed to write expressions for derivatives,

$$A = (R_x \ V_y \ - R_y \ V_x) \tag{4.8}$$

$$B = (R_x \ E_y \ - R_y \ E_x) \tag{4.9}$$

$$R_x = (y_{p+1} - y_{p-1})\cos\theta_p \ - (x_{p-1} - x_{p+1})\sin\theta_p \tag{4.10}$$

$$R_y = (y_{p+1} - y_{p-1})\sin\theta_p \ + (x_{p-1} - x_{p+1})\cos\theta_p \tag{4.11}$$

$$V_x = (x_{q-2} - x_p)\,;\ V_y = (y_{q-2} - y_p) \tag{4.12}$$

$$E_x = (x_q - x_{q-2})\,;\ E_y = (y_q - y_{q-2}) \tag{4.13}$$

Where θ_p represents the angle of a ray with respect to normal at node X_p. $(x_p\,, y_p)$ are nodal coordinates correspond to point X_p. $(x_{p-1}\,, y_{p-1})$ are nodal coordinates correspond to point X_{p-1}. $(x_{p+1}\,, y_{p+1})$ are nodal coordinates correspond to point X_{p+1}. $(x_{q-2}\,, y_{q-2})$ are nodal coordinates correspond to point X_{q-2}. $(x_q\,, y_q)$ are nodal coordinates correspond to point X_q.

Therefore, the derivative of thickness measure at node X_p with respect to x-direction is given by,

$$\frac{\partial t_p}{\partial x_p} = \frac{R_y}{B} \tag{4.14}$$

$$\frac{\partial t_p}{\partial x_{p-1}} = \frac{-(\sin\theta_p\ V_y\ +\ \cos\theta_p\ V_x)}{B} + \frac{A\ (\sin\theta_p\ E_y\ +\ \cos\theta_p\ E_x)}{B^2} \tag{4.15}$$

$$\frac{\partial t_p}{\partial x_{p+1}} = \frac{(\sin\theta_p\ V_y\ +\ \cos\theta_p\ V_x)}{B} - \frac{A\ (\sin\theta_p\ E_y\ +\ \cos\theta_p\ E_x)}{B^2} \tag{4.16}$$

$$\frac{\partial t_p}{\partial x_{q-2}} = \frac{-R_y\ (A+B)}{B^2} \tag{4.17}$$

$$\frac{\partial t_p}{\partial x_q} = \frac{R_y\ A}{B^2} \tag{4.18}$$

Similarly, the following expressions are derived for derivatives with respect to y-direction,

$$\frac{\partial t_p}{\partial y_p} = \frac{-R_x}{B} \tag{4.19}$$

$$\frac{\partial t_p}{\partial y_{p-1}} = \frac{(\cos\theta_p\ V_y\ +\ \sin\theta_p\ V_x)}{B} - \frac{A\ (-\cos\theta_p\ E_y\ +\ \sin\theta_p\ E_x)}{B^2} \tag{4.20}$$

$$\frac{\partial t_p}{\partial y_{p+1}} = \frac{(\cos\theta_p\ V_y\ -\ \sin\theta_p\ V_x)}{B} - \frac{A\ (\cos\theta_p\ E_y\ -\ \sin\theta_p\ E_x)}{B^2} \tag{4.21}$$

$$\frac{\partial t_p}{\partial y_{q-2}} = \frac{R_x\ (A+B)}{B^2} \tag{4.22}$$

$$\frac{\partial t_p}{\partial y_q} = \frac{-R_x \ A}{B^2} \tag{4.23}$$

To accommodate so many values in terms of nodal dependencies, the following data structure is implemented in MORIS. For each node on the surface mesh, we require thickness or shape diameter values, the nodal dependency vector, and for each dependent node a data structure that stores sensitivities for the x, y, and z. In Figure 4.2, the data structure adopts a standard map of C++ that stores shape diameter values corresponding to a nodal index. For storing nodal dependencies and sensitivities, it uses a vector within a vector-type data structure where the outer vector corresponds to a node of surface mesh, and the inner vector stores nodal dependencies or sensitivity information. There's a scope for improvement here by using a more compact data structure as the current module requires multiple maps while computing the final implicit part.

Figure 4.2: MORIS Data Structure for Sensitivity Information

i^{th} Node on Surface Mesh

Shape Diameter Value						std::map<index, real>
Nodal Dependency (Node Index – MTK Mesh)	1	2	3	4	5	Vector< Vector< index > >
$\partial t/\partial x$						Vector< Vector< index > >
$\partial t/\partial y$						

Once, we have all sensitivity information at all relevant nodes, depending on the index associated with x_Γ it looks for that index in the above data structure and picks the value associated with it.

Chapter 5

Results

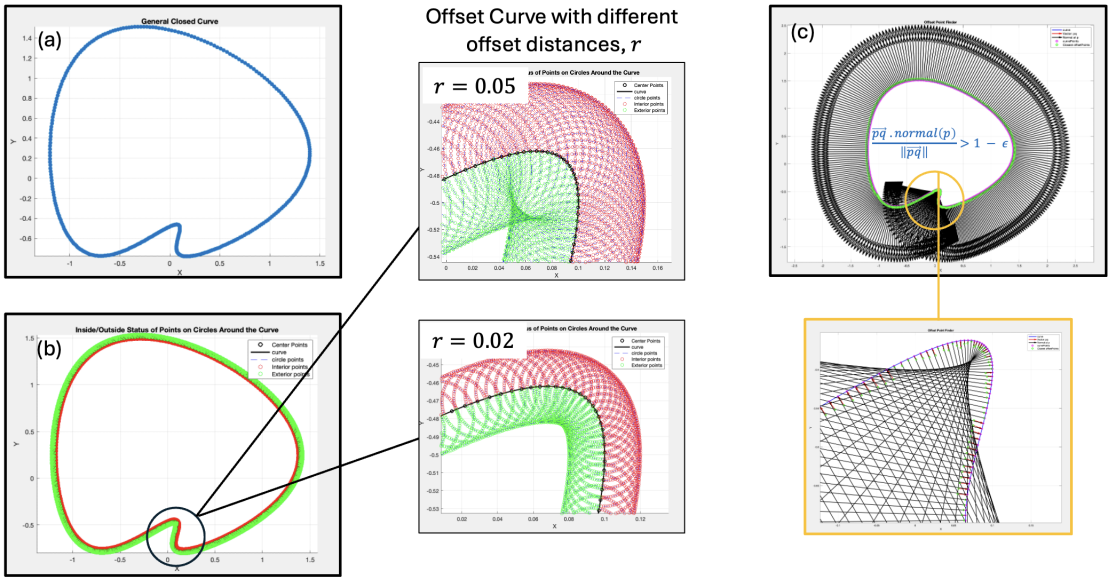
This section shows results corresponding to shape diameter computation both in MATLAB and MORIS. It starts with MATLAB based implementation for the offset curve method mentioned in Chapter 2 and highlights a key limitation associated with it for constructing offset curves accurately. Then a comparison between inward con-based shape diameter with the offset curve method sets up further exploration of results with cone-based computation. For certain interesting 2D and 3D problems, the shape diameter computation using MORIS implementation is discussed after that. The statistical analysis along with a noisy circle example highlights the advantages and limitations of this approach.

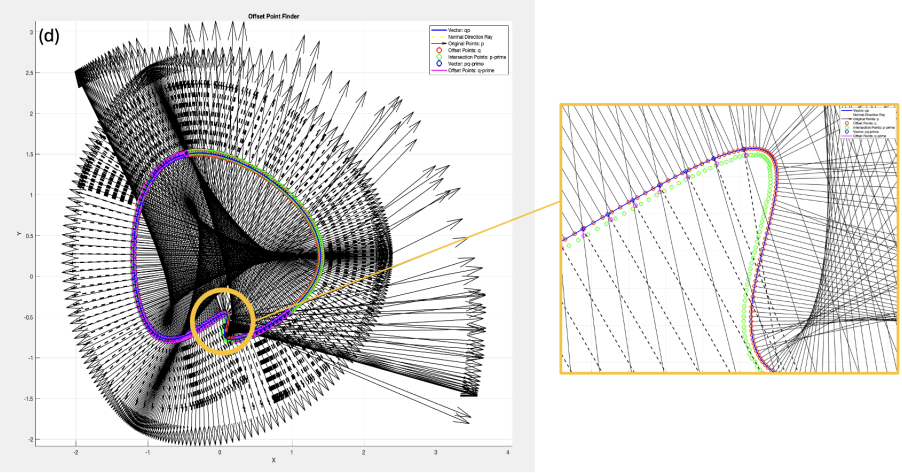
5.1 MATLAB Implementation for Offset Curve Method

This method, as mentioned in Chapter 2, requires creating an offset curve around a given closed surface mesh. After that, the closest point corresponding to each point on the surface mesh is found using a directional segment as normal for that point. Here, the offset distance is a problem-dependent parameter. In our understanding, this parameter depends on local curvature at a given point and hence, the accuracy of this method comes down to accurately constructing an offset curve. After constructing the offset curve and finding the closest points on the offset curve, a ray is cast from the closest point and allowed to intersect on a diametrically opposite side of the surface mesh on the offset curve. The shape diameter or thickness, therefore, is the difference between the intersected distance and twice the offset distance. Figure 5.1 shows a step-by-step process of

MATLAB implementation.

Figure 5.1: MATLAB Implementation of Offset Curve Method [4] (a) Input Surface Mesh, (b) Offset Curve Construction with different Offset Distances, (c) Closest Point Extraction, and (d) Intersection Point Extraction

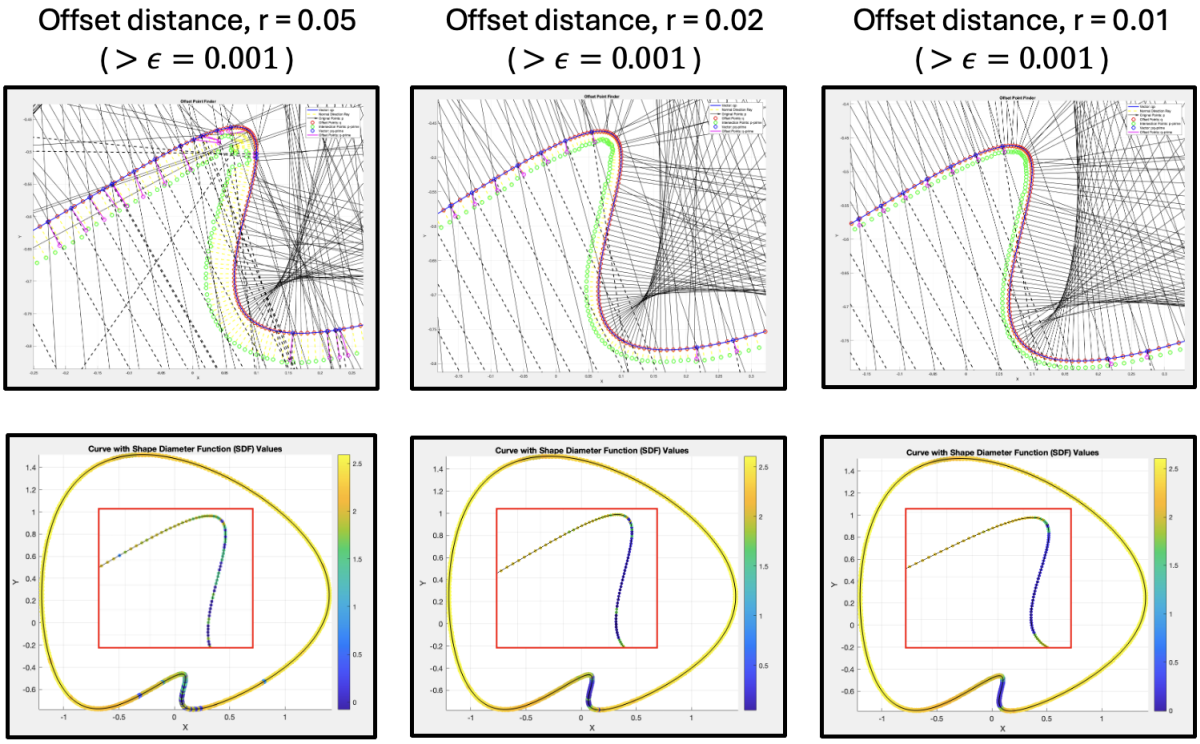




See Figure 5.2 for the shape diameter values for the curve under consideration with different offset distances. As mentioned earlier, due to dependence on local curvature, if the offset distance exceeds this value, the resulting shape diameter will not be accurately computed due to an error

in the computation of the closest point.

Figure 5.2: Offset Curve Method Shape Diameter Values



Due to the lack of connectivity information and corresponding normal computation, it is understood that this method is better suited for point clouds. However, for a closed surface mesh, the process of constructing an offset curve first and then casting a ray in a normal direction lacks an intuitive understanding. Besides, the original idea of computing shape diameter is to encode neighborhood volumetric information onto a scalar field whereas this approach only encodes information corresponding to a diametrically opposite point.

5.2 MATLAB Based Comparison of Offset Curve Method and Inward Cone Formulation

This section shows the result of shape diameter values and a primitive computation timing comparison in order to assess the accuracy and efficiency of each method.

5.2.1 Shape Diameter of Circle

As mentioned earlier, the shape diameter of a circle is equal to its diameter. This test is necessary to check the accuracy of both methods. In this test, the circle of diameter equal to 10 units is used. For the offset method, the offset distance of 1 unit is taken. For an inward cone method, a cone angle of 30 degrees with 60 rays is used as parameters. As shown in Figure 5.3, both methods appear to work satisfactorily while the cone-based formulation computes close to analytical values. Figure 5.4, a very primitive computation time comparison as the number of points on the surface mesh increases. Surprisingly, for this example, the cone-based method performs better compared to the offset curve.

Figure 5.3: (a) Inward Cone Formulation, and (b) Offset Curve Method

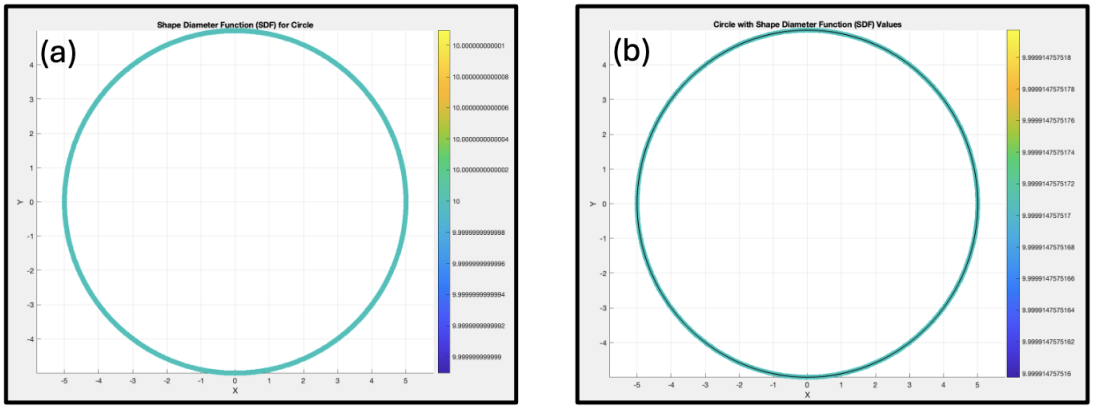
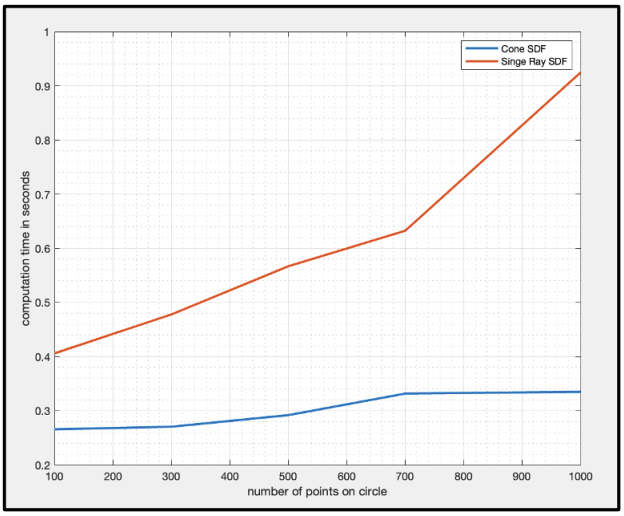


Figure 5.4: Computation Time Comparison



5.2.2 Shape Diameter of a Generic Curve

This particular test is done to identify the problems associated with both methods. For example, the inward cone method becomes sensitive to extremely high cone angles as suggested by Shapira et al. [18] and Madaras et al. [14]. In the case of an offset curve method, although it improves significantly on timing tests, it shows sensitivity to offset distances at high curvature regions. For this test, the offset distance of 0.01 unit is taken whereas the cone angle of 60 degrees with 90 rays is taken for the other case. It is hard to judge the accuracy of two methods for any generic curve as there is no analytical solution, however, a quantitative comparison between two methods can provide a trend in favor or against a method. Figure 5.5 shows a first comparison between these two methods for a generic curve. Although it's hard to tell the difference directly, it confirms that both methods yield similar results at most points on surface meshes. Timing comparison favors the offset curve method. However, the next set of comparisons with different cone angles provides a trend that shows the sensitivity of cone angle to provide inaccurate results with extremely high cone angles. By increasing the number of rays with an increase in cone angle some improvement has been observed but at high angles (> 75 degrees), this method is not useful.

Figure 5.5: (a) Inward Cone Formulation, (b) Offset Curve Method, and (c) Computation Time Comparison

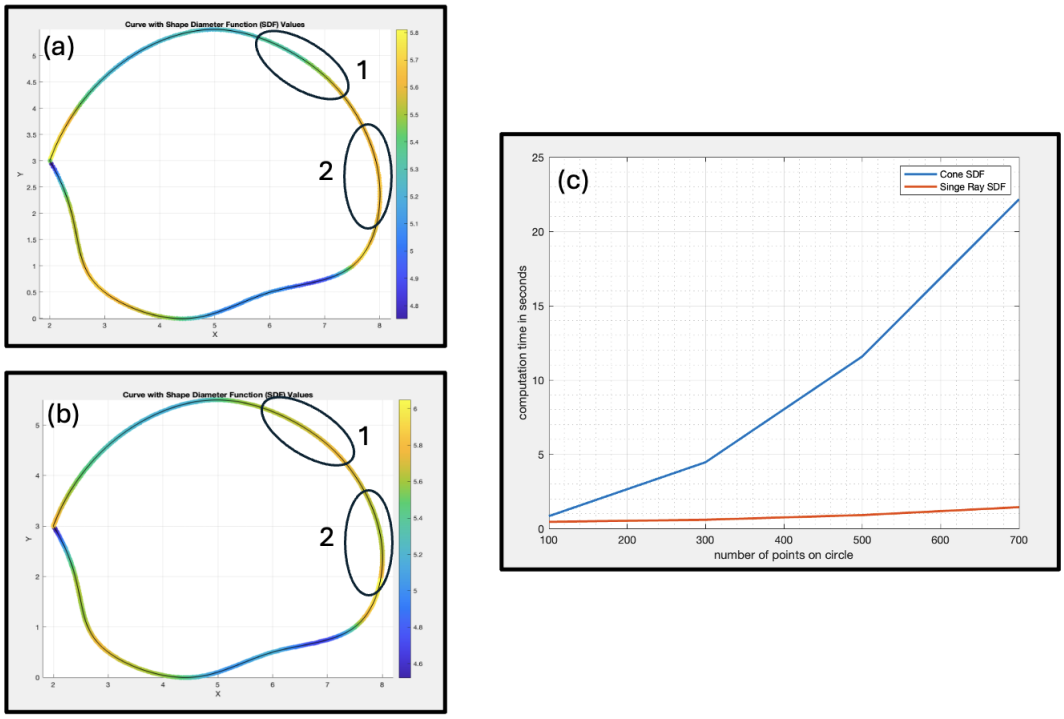
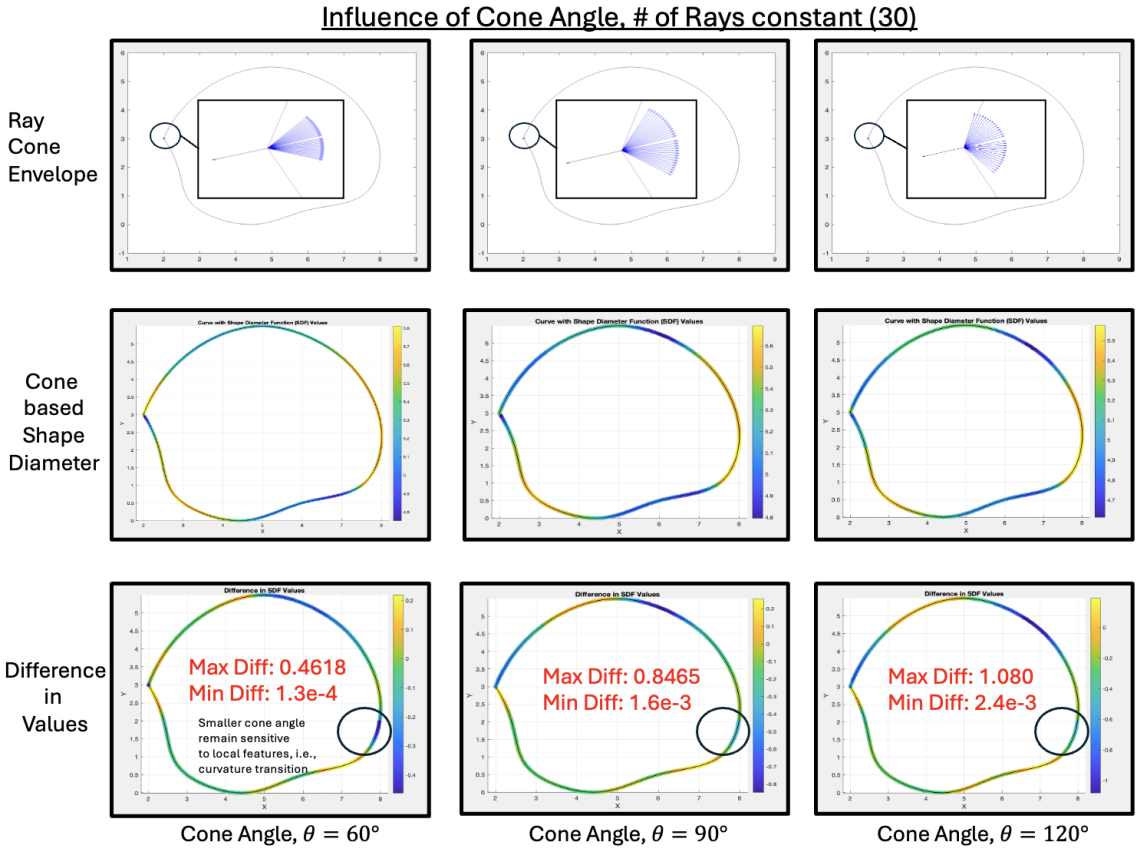


Figure 5.6 shows the comparison of cone-based formulation as the cone angle increases. The last row in Figure 5.6 plots the difference in the value of shape diameter between the two methods. This example indicates sensitivity with respect to input parameters, i.e., cone angle. As indicated, the maximum and minimum difference between these two methods increases with an increase in cone angle. This is an indication that as the cone angle increases, the cone method attracts more rays falling within the standard deviation criteria. This encodes more noise into the final shape diameter expression.

A similar trend is observed for another curve as shown in Figure 5.7.

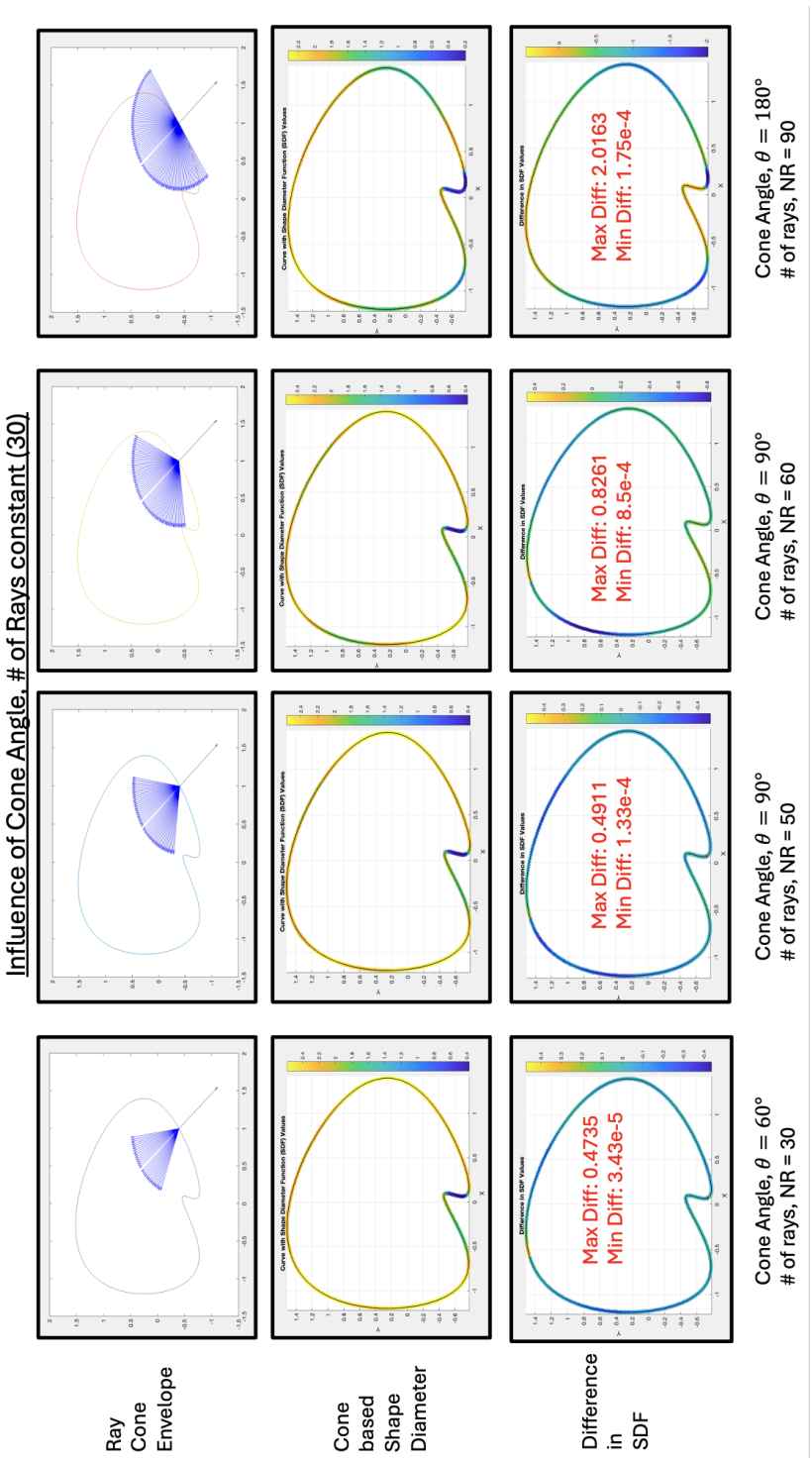
Figure 5.6: Influence of Cone Angle, First Row: Cone Ray Envelope, Second Row: Shape Diameter Value, Third Row: Difference in Shape Diameter Compared with Offset Method



5.3 MORIS Results for Shape Diameter Using Inward Cone Formulation

As discussed in Chapter 3, MORIS implementation of cone formulation can work for both inward-outward cone formulation simultaneously. Meaning for the optimization problem where the user-defined geometry, i.e. circle, is embedded into the computational domain, it is necessary to compute the shape diameter based on a hollow surface mesh. The initial few examples for both 2D and 3D implementation validate the implementation.

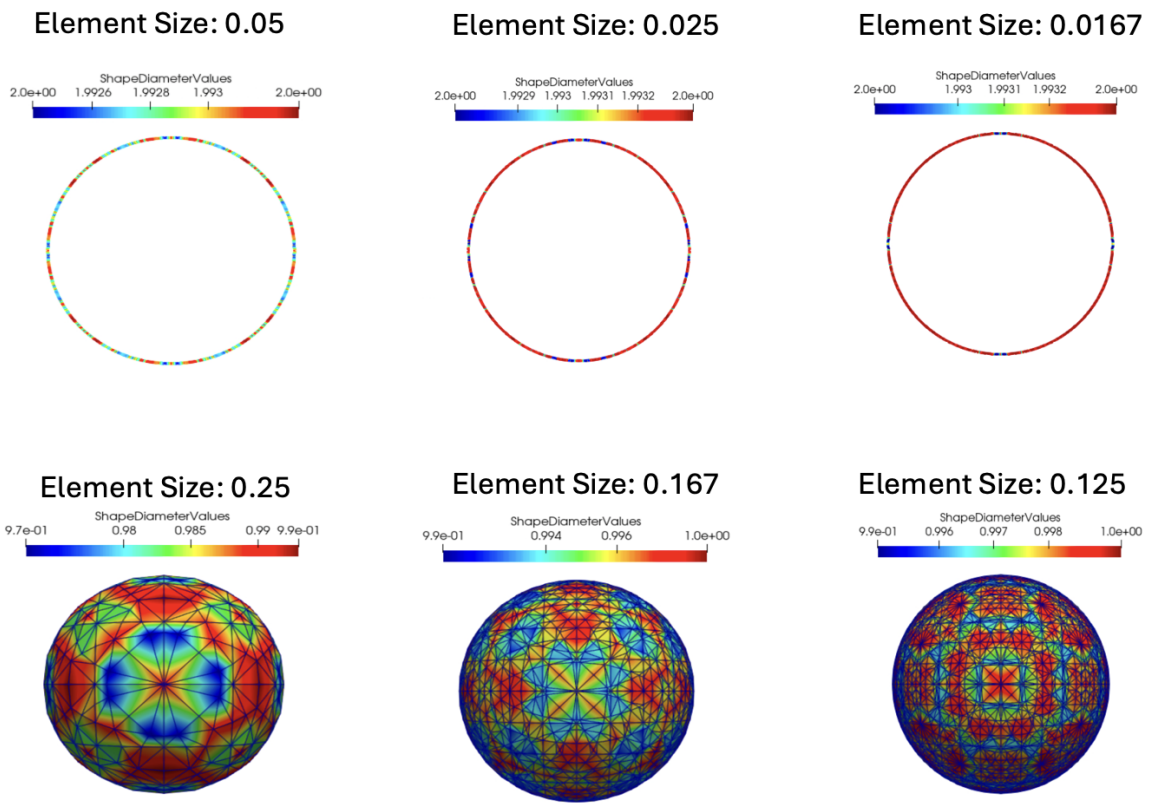
Figure 5.7: Influence of Cone Angle, First Row: Cone Ray Envelope, Second Row: Shape Diameter Value, Third Row: Difference in Shape Diameter Compared with Offset Method



5.3.1 Shape Diameter of a Circle and a Sphere

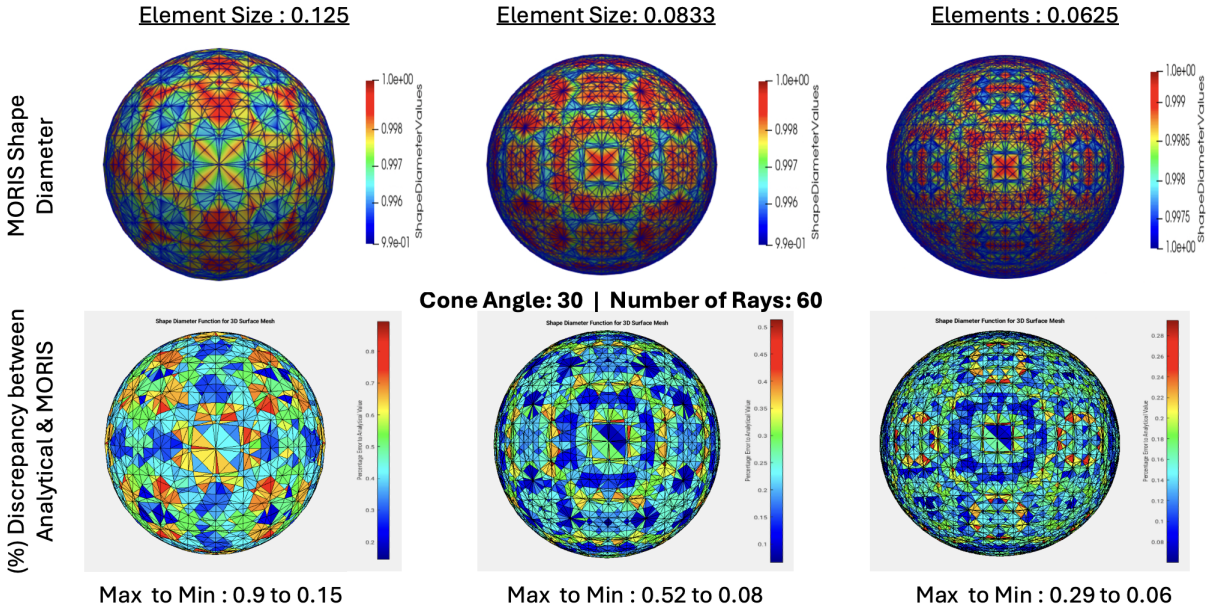
For this study, a cone angle of 30 with 30 rays is used for the circle and a similar cone angle with 100 rays is used for the sphere. As expected with mesh refinement, the values of shape diameter converge to the analytical solution. The circle has a radius of 1 unit whereas the sphere has a radius of 0.5 unit. For the sphere case, a visually symmetric pattern is the evidence of robustness of this method in handling 3D surface meshes.

Figure 5.8: MORIS Shape Diameter Results, First Row: Circle, Second Row: Sphere



However, the discrepancy against the analytical solution as shown in Figure 5.9, particularly for the sphere case, suggests that this method still suffers from accuracy if the mesh is not sufficiently fine. However, with mesh refinement, the error appears to go down and converges to the analytical solution.

Figure 5.9: Discrepancy Plots: Sphere

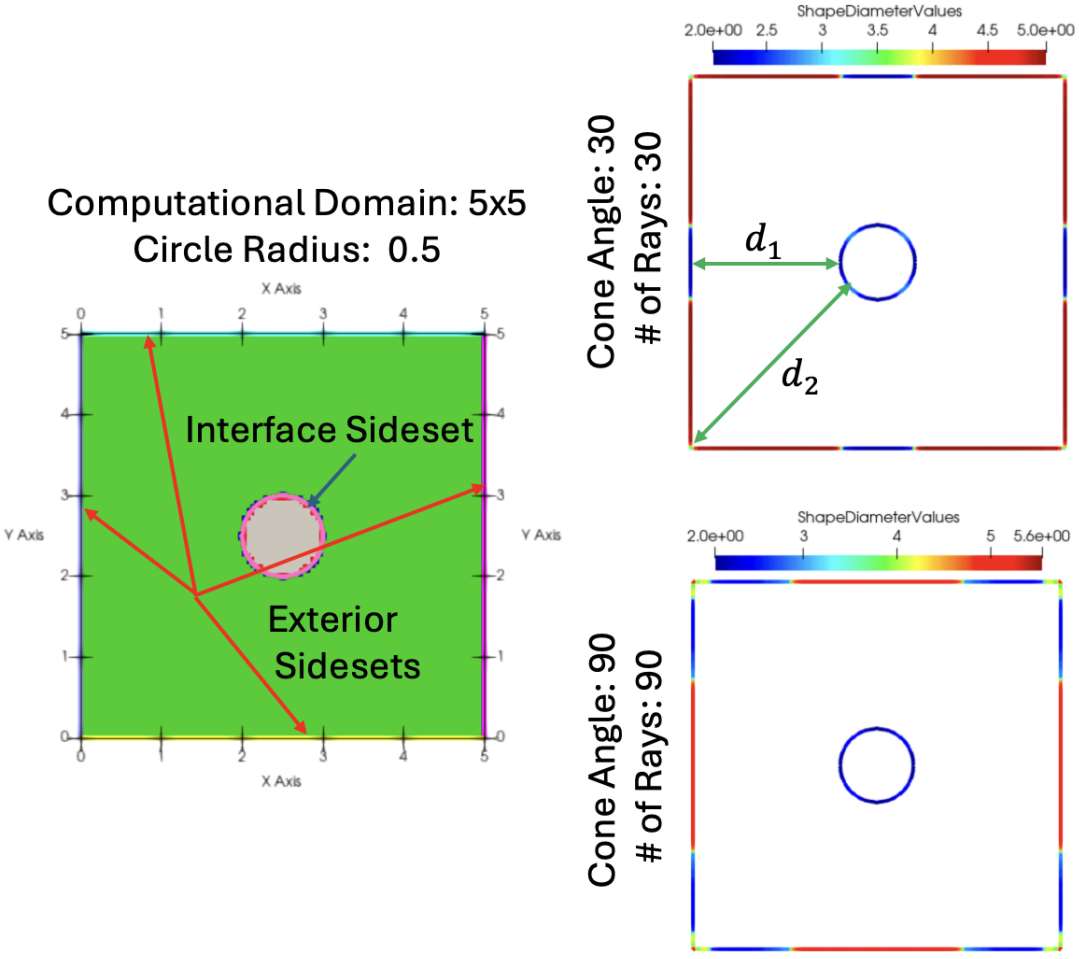


5.3.2 Shape Diameter of a Circle and a Sphere Inside a Computational Domain

The previous example being a closed surface mesh, only needed an inward-looking cone aligned to a vertex normal. In this section, a hollow surface mesh in the sense of having a multi-material domain separated by the interface is studied. The exterior sidesets, the boundaries of the square computational domain as shown in Figure 5.10, need to construct inward looking cone, whereas, the interface sideset, that is the interface between a square and a circle domain in Figure 5.10, needs to create an outward-looking cone. The ability to handle this hollow surface mesh makes this method suitable for optimization problems. As seen in Figure 5.10, with an increase in cone angle, although the shape diameter values at the interface do not explode, at the exterior sidesets, particularly in the middle of each sideset and at the corner, due to increase in number of valid rays through standard deviation criteria, the shape diameter values explode significantly. At the corners and at the middle of each exterior sidesets, however, for small cone angles, the values match the analytical solution. For example, in Figure 5.10, the thickness measure d_1 *and* d_2 are just geometric distances. These can be computed by subtracting the radius of a circle from the

distance to the center of the computational domain.

Figure 5.10: Shape Diameter for Hollow Surface Mesh: Circle



For the sphere case embedded into the computational domain, a case for increasing ray density is found. By increasing ray density within the same cone, a spurious numerical artifact at the corner of a cube is removed. In this case, a sphere of radius 0.5 unit is centered inside the cubic computational domain of size 5 x 5 x 5. The following two figures show the shape diameter values on exterior sidesets and the interface sideset, that is the surface of a sphere. Figure 5.11 and 5.12 shows the improvement due to denser ray distribution.

Figure 5.11: Shape Diameter for Hollow Surface Mesh: Sphere, Coarse Mesh

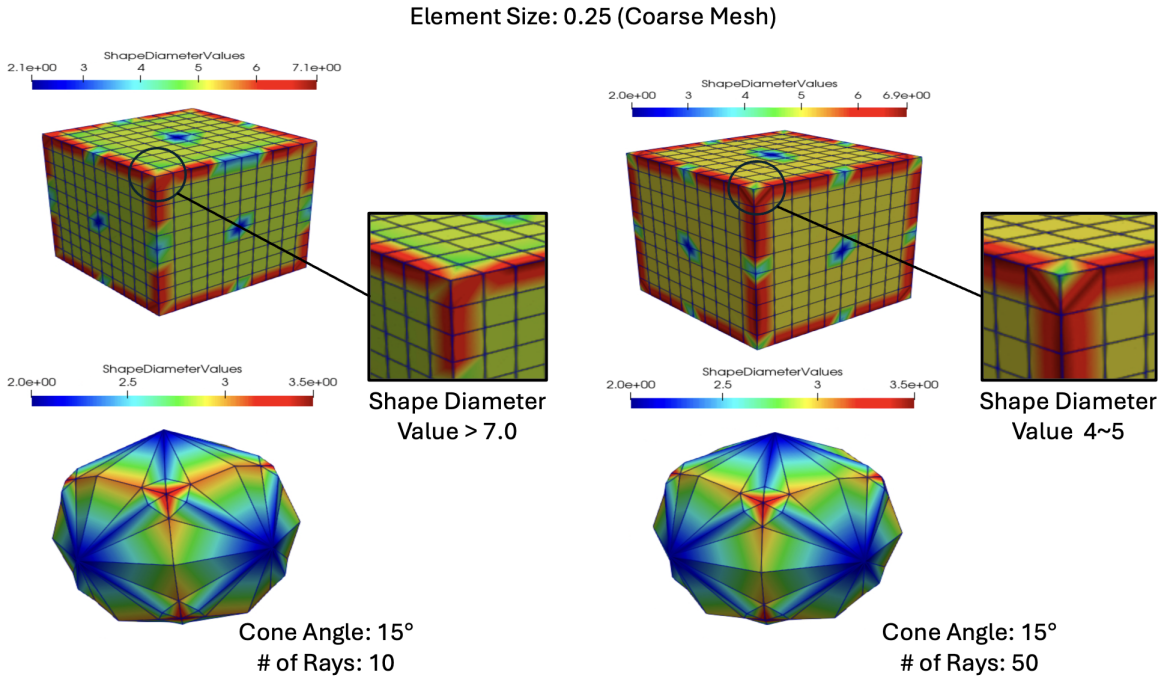
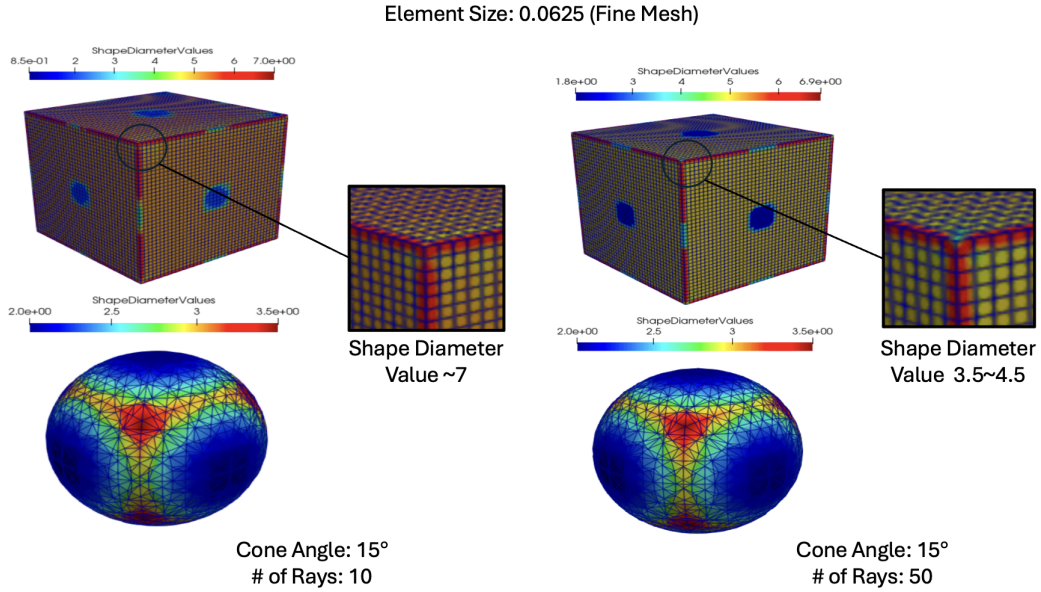


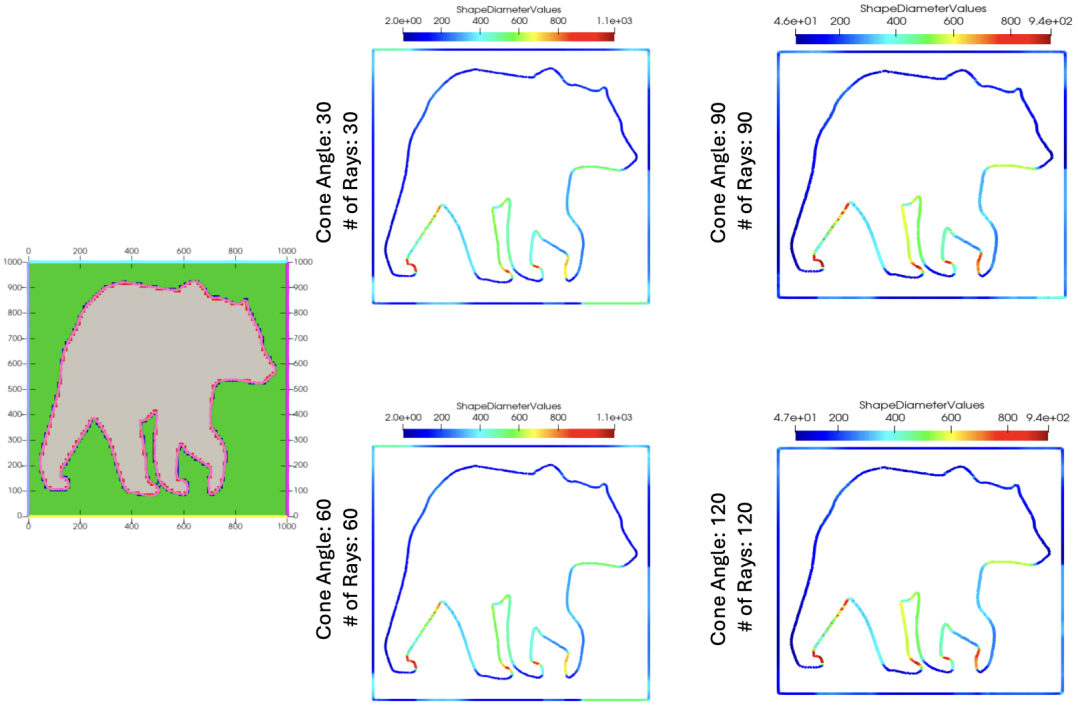
Figure 5.12: Shape Diameter for Hollow Surface Mesh: Sphere, Fine Mesh



5.3.3 Shape Diameter of a 2D California Bear

This example investigates the robustness and accuracy of this implementation for a geometry with rich details in the sense of moderate to high curvature changes. However, here, the cone-ray density is kept constant for all the cases to understand if equivalent changes in a number of rays beyond a threshold value of cone angle are helpful or not. While, the interface sideset, that is the boundary of California bear, provides consistent results, the noise is encoded in the values of shape diameter for exterior sidesets with an increase in cone angle. This 2D geometry of the California bear is embedded in the computational domain as a signed distance image. While previous examples have shown a common trend against increasing the cone angle, this example, in particular, shows consistent results even with higher angles at least for the interface nodes. It could be due to the fact that at moderate to higher curvature regions, bigger cone angles flatten out the local curvature noise yielding better results.

Figure 5.13: Shape Diameter of 2D California Bear



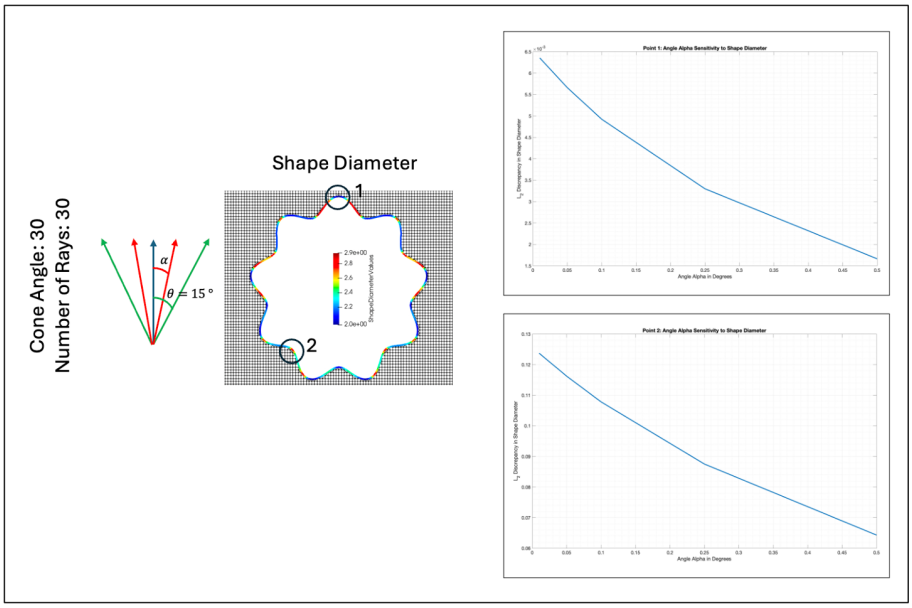
5.4 Perturbation Study

Various example in this section reinforces the effectiveness of cone-based shape diameter computation over point-to-point mapping. In this context, point-to-point mapping refers to using a single ray that is aligned in the direction of normal at the vertex of the surface mesh. This approach is similar to the offset curve method described earlier, however, it does not need to compute any offset curve. This study demonstrates the need to use multiple rays over a single-ray approach.

5.4.1 Small Angle Sensitivity

As discussed in Chapter 3, the angle of the closest ray to the normal within a cone has an influence on the shape diameter values, particularly for smaller cone angles i.e. less than 30 degrees. At higher cone angles, since it attracts a higher degree of noise from the neighborhood, the effect of a small angle, α, is not that significant. For a wiggly closed surface mesh embedded within the computational domain as shown in Figure 5.14, a small angle α sensitivity study is performed.

Figure 5.14: Discrepancy Plots of Shape Diameter with Different Values of α



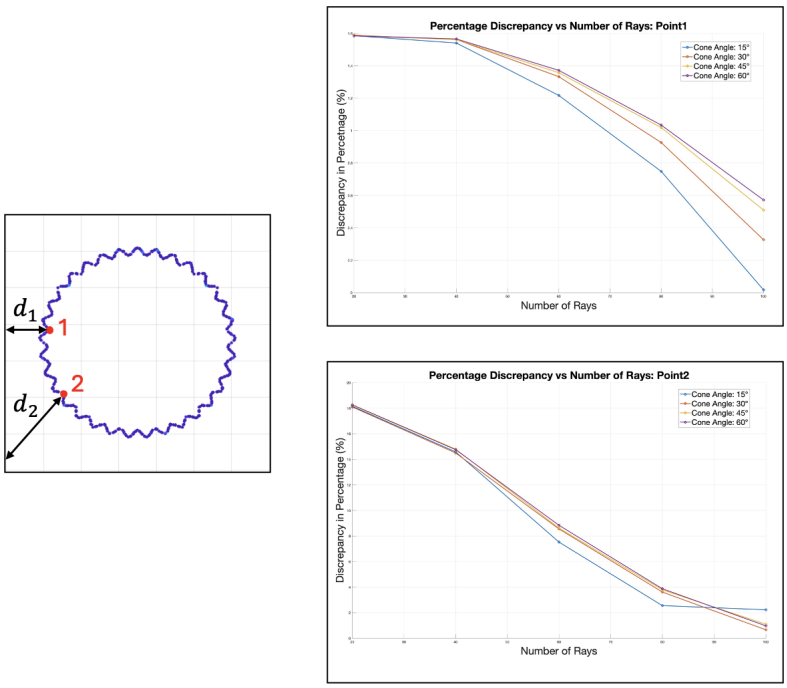
As expected, with a decrease in the value of α the error more or less increases linearly for different points on the wiggly curve. This is because, as this angle α decreases corresponding weight associated with the ray would explode. This reinforces the need to limit the smallest unit angle of a ray within the cone.

5.4.2 Shape Diameter for a Circle with Artificial Noise

In this study, an artificial harmonic noise is introduced in the circle to study the influence of noise on the shape diameter values when compared with different cone-ray distributions. The geometry is defined as,

$$d = r \ + A \ \sin f\theta \tag{5.1}$$

Figure 5.15: Discrepancy Plots of Shape Diameter with Artificial Noise



Where A is the amplitude and f is the frequency of oscillation. For this particular study amplitude is randomly set to 0.02 and frequency to 33 in order to a particular pattern of oscillation

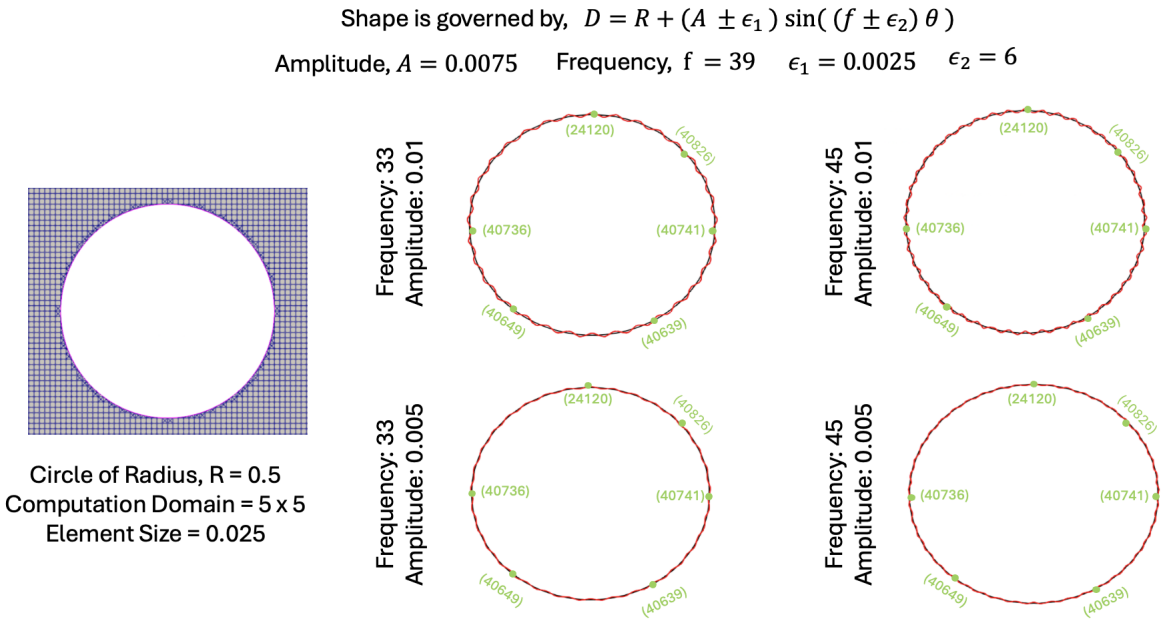
such that the values can be compared with ground truth, that is a circle. Point-1 and Point-2 are selected such that the analytical values are comparable with point-to-point mapping through a single ray.

The plots in Figure 5.15 show that by increasing the number of rays and using a smaller angle, the error in shape diameter values decreases. Although, the influence of cone angle beyond 15 degrees is more or less similar, slightly unexpected results are obtained at higher ray density even for small cone angles.

5.4.3 Influence of Error in Normal with Artificial Noise

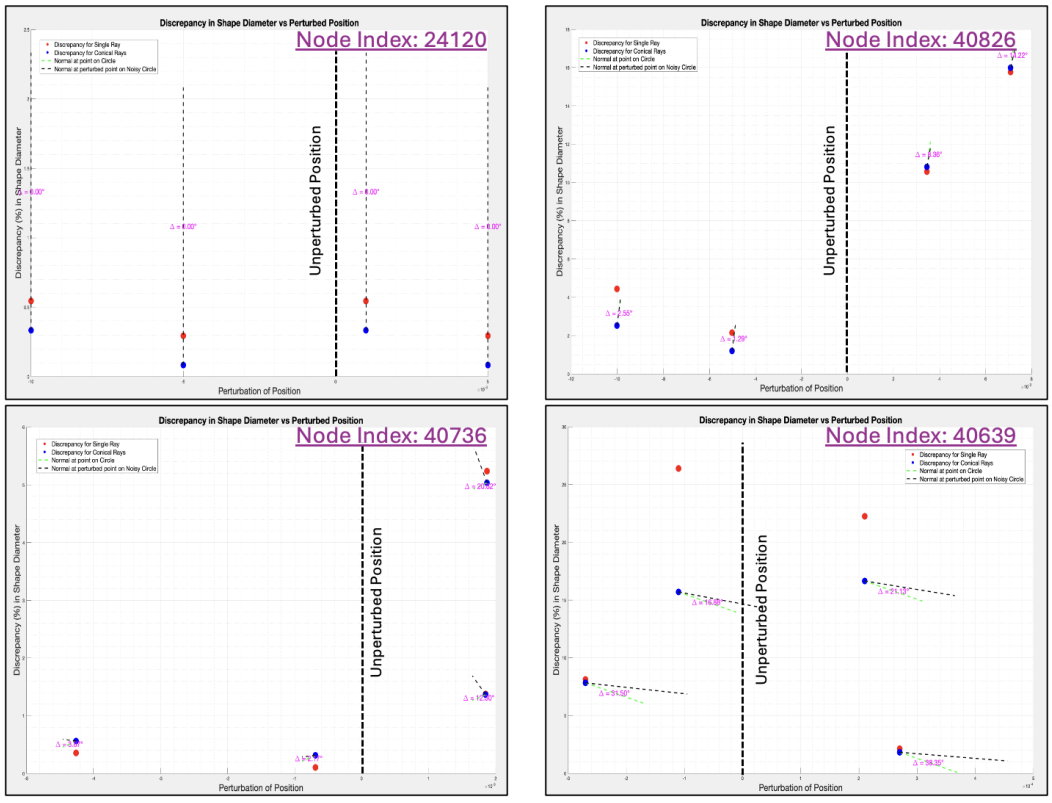
In a similar study as described in section 5.4.2, the geometry with artificial noise which depends on mesh discretization creates an error in normal. From this study, at various points on the perturbed circle, it is concluded that the error in shape diameter is directly related to the error in normal computation. For this particular case, the results of point-to-point mapping are also plotted to see if the cone formulation is effective or not.

Figure 5.16: Geometry Description and Concerned Nodes



As shown in Figure 5.16, the frequency and amplitude variation provide four different perturbed configurations of artificial noise to study. A comparative plots for different nodes shown as green dots, are produced. The plots shown in Figure 5.17 compare the discrepancy in shape diameter associated with the cone approach and point-to-point mapping. It also shows the error in normal as Δ. For most nodes, it is evident that cone formulation has a lower magnitude of discrepancy compared to point-to-point mapping. It is also evident that in most cases, the discrepancy in shape diameter is related to the error in normal. In this study, the cone angle of 30 degrees with 150 rays is used. The values of the shape diameter are compared to the Euclidean distance measure.

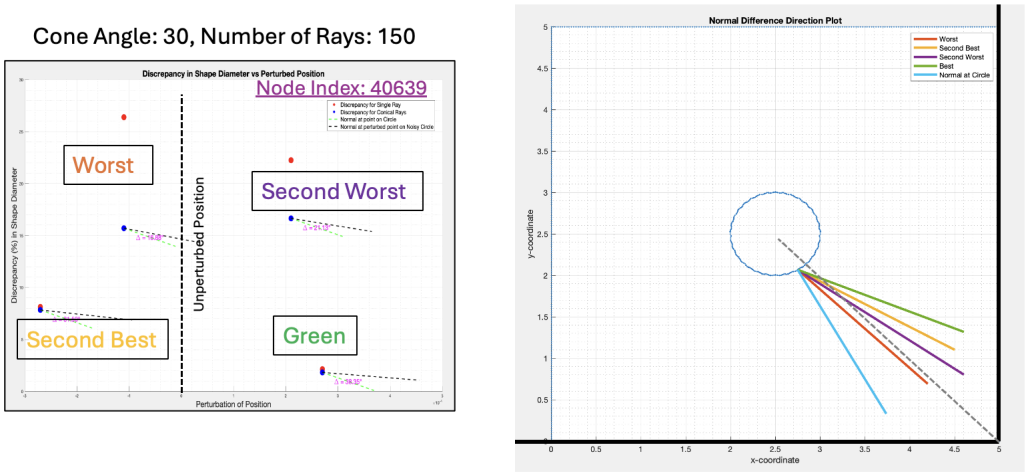
Figure 5.17: Error in Normal Relation with Discrepancy in Shape Diameters



Notes: Blue dots in Figure 5.17 indicate the discrepancy in shape diameter for the cone-based approach as the position of the node is perturbed. Red dots refer to point-to-point mapping discrep-

ancies. Δ refers to the error in normal for a perturbed node. The black bold dotted line indicates the original unperturbed position of each node. The black dotted line indicates the numerical normal, whereas the green dotted line indicates the analytical normal. The only counter-intuitive case is of node index 40639, where for a relatively higher error in normal along with high perturbation, the discrepancy in shape diameter remains significantly low. Upon further investigation, a limiting case for this approach is found. To understand this consider Figure 5.18, where the analytical normal is plotted along with other normals (corresponding to different perturbed positions) for the same node. It is found that the numerical normal with an error of about 38 degrees is somewhat symmetric to the analytical normal with respect to a diagonal drawn from the corner node. It implies that even with a huge error in normal, it will still attract a similar cone envelope as if the numerical normal is exactly aligned with the analytical normal.

Figure 5.18: Counter-Intuitive Result due to Symmetric Normal



5.4.4 Parallel Planes with Artificial Noise

This example is a particular case where the large cone angle formulation performs much better compared to point-to-point mapping and small cone formulation due to the extremely noisy neighborhood.

The harmonic noise of the following form in the Y-direction is introduced,

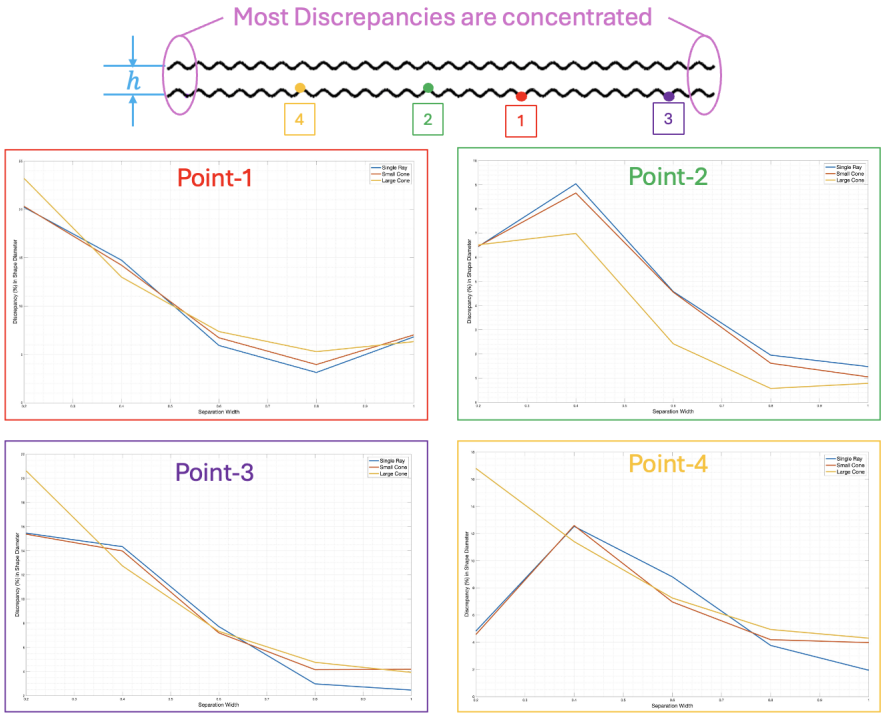
$$\delta_y = A\sin(fx + \phi) \tag{5.2}$$

where A corresponds to some constant amplitude, f is the frequency and ϕ is the phase associated with each line.

Case: In-phase Parallel Noisy Lines

In this study, a histogram plot of shape diameter value distribution is computed to understand the effect of a larger cone envelope. As seen in Figure 5.19, a larger cone removes the outlier of shape diameter values consistently with different separation widths.

Figure 5.19: In-phase Noisy Parallel Line Shape Diameter Convergence Plots



Discrepancy plots indicate a smoother convergence for most cases for a larger cone. This is attributed to the fact that as the separation width increases beyond the amplitude of noise, the influence of noise on the error becomes less significant.

Figure 5.20: In-phase Noisy Parallel Line Histogram : Point-To-Point Map

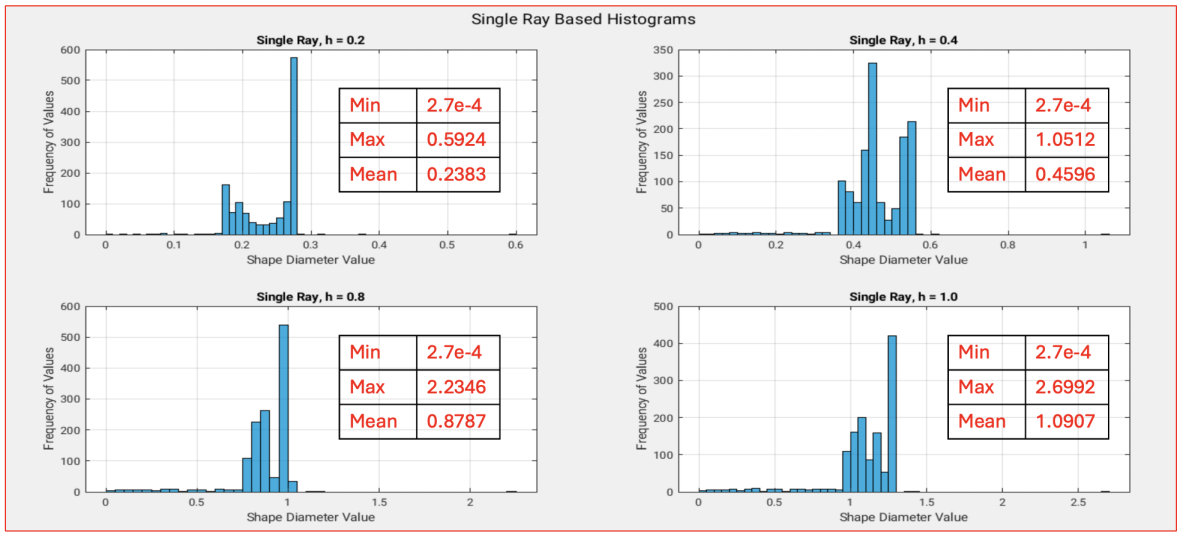
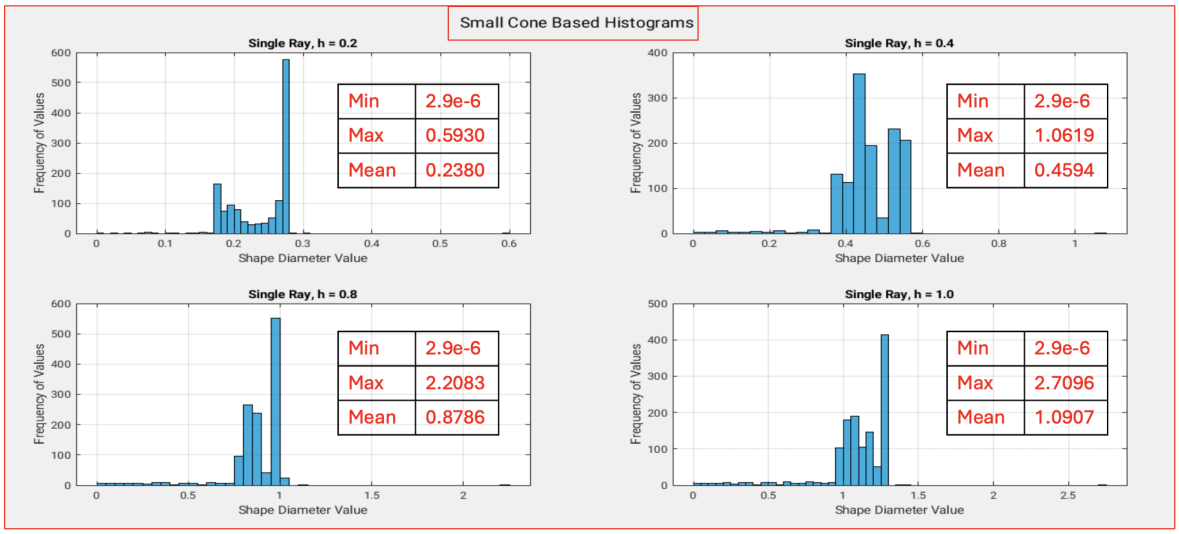
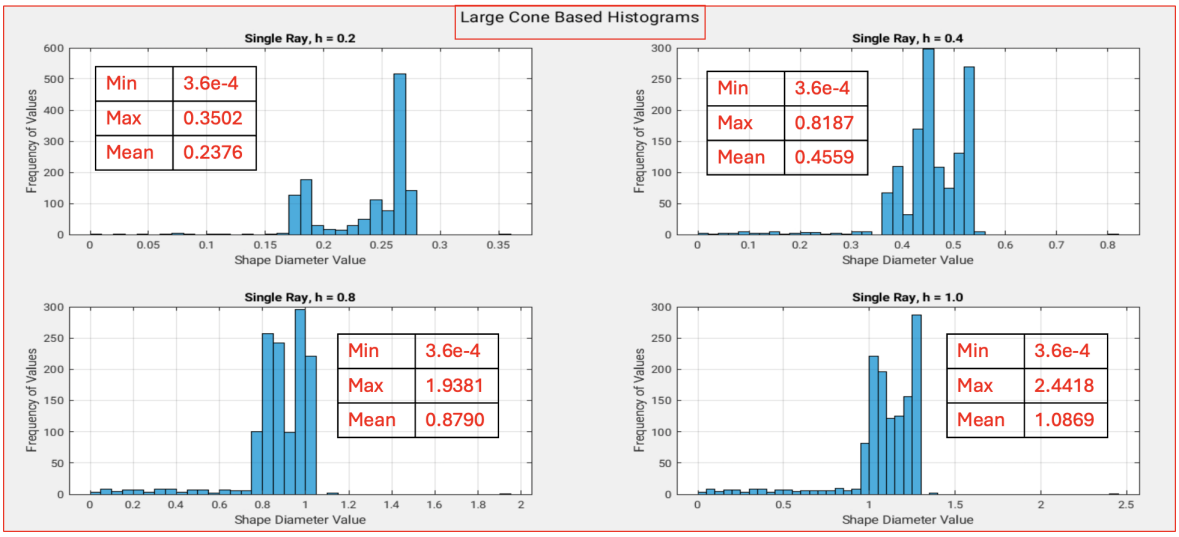


Figure 5.21: In-phase Noisy Parallel Line Histogram : Small Cone



Figures 5.20 to 5.22 show the histogram plots associated with (a) Single Ray-based point-to-point mapping, (b) Small Cone, and (c) Larger Cone.

Figure 5.22: In-phase Noisy Parallel Line Histogram : Large Cone



5.5 Integrated Quantity for Shape Diameter

The integrated quantity of shape diameter represents a cumulative measure of shape diameter integrated over the surface in this case over the sidesets. As this measure will be used for optimization problems, it is necessary to verify any discrepancies associated with it. For a circle with inward cone formulation, the integrated quantity can be computed analytically as follows,

$$IQI = \int_{\theta=0}^{\theta=2\pi} D_{SDF} R d\theta = 2\pi R D_{SDF} \tag{5.3}$$

For a circle of unit radius, $D_{SDF} = 2$, therefore,

$$IQI = 4\pi = 12.5664 \tag{5.4}$$

The following table shows, the different IQI values for various mesh discretizations,

Similarly, for a sphere, the following analytical value is derived,

$$IQI = \int_{\theta=0}^{\theta=2\pi} \int_{\phi=0}^{\phi=\pi} D_{SDF}\, R^2 \sin(\theta)\, d\theta\, d\phi = 4\pi R^2 D_{SDF}$$

Table 5.1: Integrated Value of Shape Diameter at Various Mesh Discretization for Circle

Element Size	Integrated Value	Percentage Discrepancy
0.0500	12.5211	0.360
0.0250	12.5235	0.341
0.0167	12.5241	0.336

For a circle of 0.5 radius, $D_{SDF} = 1$, therefore,

$$IQI = 3.1416 \tag{5.6}$$

Table 5.2: Integrated Value of Shape Diameter at Various Mesh Discretization for Sphere

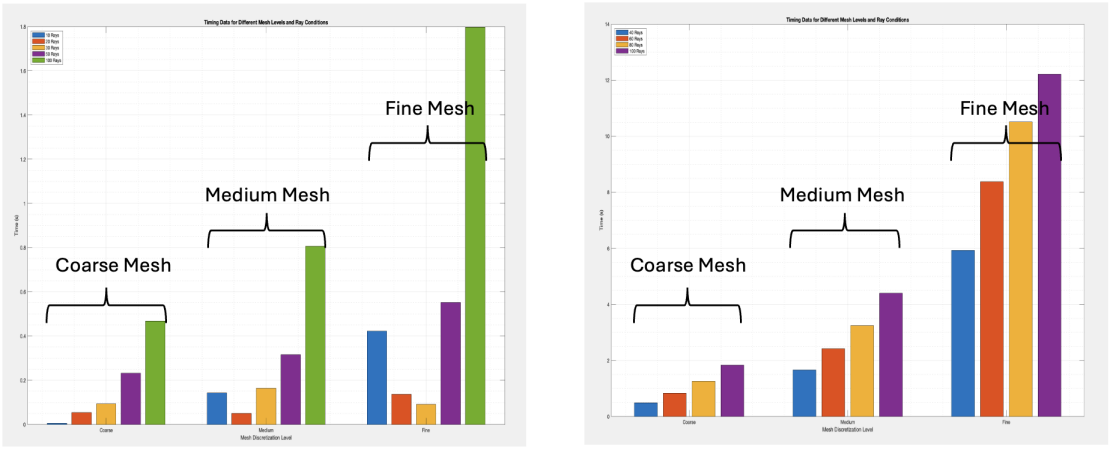
Element Size	Integrated Value	Percentage Discrepancy
0.250	3.02554	3.694
0.125	3.10882	1.040
0.083	3.12629	0.487

5.6 Computation Time Study

A computational cost study for MORIS is performed to understand the time taken to perform shape-diameter computation for both circle (2D) and sphere (3D) cases. Due to the addition of batching functionality that calls ArborX library [11] and performs ray-triangle intersection simultaneously for all rays on all points the computational time drops by some margin. However, a more rigorous computational cost study with varying degrees of examples is required to comment any further.

Figure 5.21 shows that with an increase in the number of rays per point, the computational cost increases linearly for both the circle and the sphere with one exceptional case for a circle. Overall, the addition of time to MORIS run time is quite insignificant except for the sphere case with denser ray distribution. For a full optimization run in 3D problems, this requires efficient data storage and transfer.

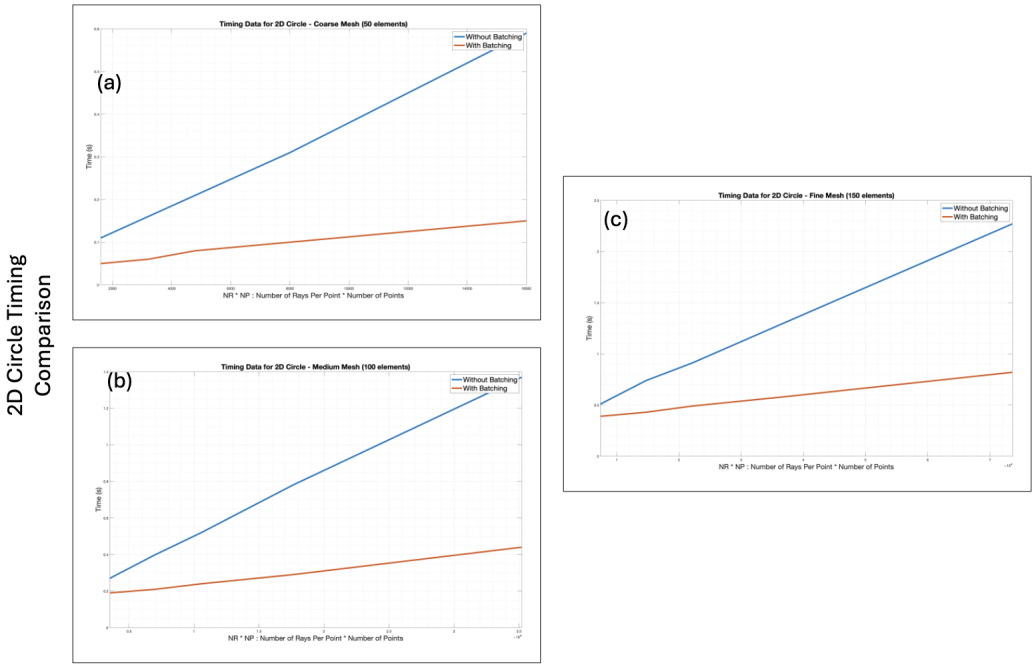
Figure 5.23: Computation Time Comparison in MORIS



2D Circle Timing Comparison

3D Sphere Timing Comparison

Figure 5.24: Computation Time Comparison in MORIS with and without Batching (a) Coarse Mesh, (b) Medium Mesh, and (c) Fine Mesh



Chapter 6

Conclusion

This thesis intends to demonstrate various mathematical formulations that can be applied to construct a constraint or penalty term within a constrained optimization problem. During this thesis, various approaches to enforce a minimum feature size. Prior to the formulation of this thesis topic, work on shape regularization using perimeter area with curvature information embedded in the formulation was explored. Although this thesis does not explicitly talk about optimization results obtained from that study, it certainly highlights the limitation associated with it for geometric features with extreme curvatures.

One way to use prior thickness constraint formulation is by stabilizing the curvature of the interfaces. A Tikhonov-formulation-based approach was studied and implemented by adding higher-order derivative terms into the L_2 map. Although this attempt yielded improved curvature results for moderate to small curvature geometric entities, for extreme curvature cases, the curvature noise remained extremely high. Future attempts exploring stabilization methods for this extreme case may be well-suited for optimization problems.

For the scope of this thesis, a unique way to measure local thickness as a shape diameter is explored. Due to a lack of an analytical relationship between the shape diameter and physical thickness, there's no direct way found to enforce an equality constraint on feature size. However, an implicit way to enforce the feature size constraint through the shape diameter is discussed. While this implicit formulation can work as long as the shape diameter value remains equal to or lower than the corresponding thickness value, it requires rigorous numerical experiments to

fine-tune factors for the other inequality case. For shape regularization using shape diameter, at least two methods are discussed. The first formulation uses the skeleton-based approach, where the skeleton structure is constructed using shape diameter, and then using a min-max radius circle fitting approach at each node of the skeleton, or by constructing a minimum thickness penalty term using shape diameter. The shape diameter computation through mainly the offset curve method and cone ray casting approach is discussed in detail. Based on limited numerical experiments, while the offset curve method is faster, the use cases are limited in the sense of not being applicable to non-watertight meshes and robustness of normal computation. The major challenge for this method lies in the construction of an offset curve. On the other hand, the original cone ray casting formulation has a wider base of examples to conduct numerical experiments on. This implementation is simple and robust for both 2D and 3D examples. One key disadvantage is that this method requires fine-tuning of parameters such as cone angle and number of rays, depending on the problem. For a smaller noise compared to the size of the geometry, the small cone provides expected results, whereas, when the magnitude of noise in the geometry is comparable to the size of the geometry, a large cone provides comparable results as in the case of parallel lines. Histogram plots for one of the examples highlight the influence of a large cone in removing outlier values of shape diameter. For a noisy geometric feature, we have studied the influence of error in normal at different mesh discretizations. A detailed investigation of this problem suggested that although with mesh discretization, there's some improvement in normal estimation, there's not much impact on shape diameter values. Various 2D and 3D examples have demonstrated the applicability of this method for a variety of problems.

Although the original scope of this thesis is to demonstrate the applicability of shape diameter for a shape regularization problem, a numerical error pertaining to the implementation of sensitivity analysis has halted that progress. However, Chapter 4 discusses in detail the sensitivity analysis regarding the cone-based shape diameter approach. The complicated part of sensitivity analysis is to store the data structure with accurate nodal maps for later use.

In summary, this thesis is an attempt to showcase a potential use of the shape diameter

measure for a shape regularization problem in Topology Optimization. While there's no concrete evidence yet for this approach to provide a better result compared to other methods, with implementation developed in MORIS during the course of this thesis, it is hoped that future attempts will successfully demonstrate the use case of it. Due to the computational cost involved with the ray casting approach, a robust implementation through the offset curve method will be suitable for an optimization problem.

Bibliography

[1] Oscar Au, Chiew-Lan Tai, Hung-Kuo Chu, Daniel Cohen-Or, and Tong-Yee Lee. Skeleton extraction by mesh contraction. ACM Trans. Graph., 27, 08 2008.

[2] Jorge-Luis Barrera, Markus Geiss, and Kurt Maute. Minimum feature size control in level set topology optimization via density fields. 03 2021.

[3] Shikui Chen, Michael Yu Wang, and Ai Qun Liu. Shape feature control in structural topology optimization. Computer-Aided Design, 40(9):951–962, 2008.

[4] Shuangmin Chen, Taijun Liu, Zhenyu Shu, Shi-Qing Xin, Ying He, and Changhe Tu. Fast and robust shape diameter function. PLOS ONE, 13:e0190666, 01 2018.

[5] Keenan Crane, Clarisse Weischedel, and Max Wardetzky. The heat method for distance computation. Commun. ACM, 60(11):90–99, October 2017.

[6] Markus Josef Geiss. Level-set-XFEM-density topology optimization of active structures: methods and applications. PhD thesis, University of Colorado at Boulder, 2019.

[7] Xu Guo, Weisheng Zhang, and Wenliang Zhong. Explicit feature control in structural topology optimization via level set method. Computer Methods in Applied Mechanics and Engineering, 272:354–378, 2014.

[8] Xu Guo, Weisheng Zhang, and Wenliang Zhong. Explicit feature control in structural topology optimization via level set method. Computer Methods in Applied Mechanics and Engineering, 272:354–378, 2014.

[9] Martin Huska and Serena Morigi. A meshless strategy for shape diameter analysis. The Visual Computer, 33, 03 2017.

[10] Sebastian Kreissl, Georg Pingen, and Kurt Maute. Topology optimization for unsteady flow. International Journal for Numerical Methods in Engineering, 87:1229 – 1253, 09 2011.

[11] D. Lebrun-Grandié, Andrey Prokopenko, Bruno Turcksin, and S. Slattery. Arborx: A performance portable geometric search library. ACM Transactions on Mathematical Software, 47:1–15, 12 2020.

[12] Jikai Liu, Lei Li, and Yongsheng Ma. Uniform thickness control without pre-specifying the length scale target under the level set topology optimization framework. Advances in Engineering Software, 115:204–216, 2018.

[13] Zhen Luo, Michael Wang, S. Wang, and Peng Wei. A level set-based parameterization method for structural shape and topology optimization. International Journal for Numerical Methods in Engineering, 76:1 – 26, 10 2008.

[14] Martin Madaras, Rastislav Kamenický, Adam Riečický, Roman Ďurikovič, Andrea Baldacci, Paolo Cignoni, and Roberto Scopigno. Gpu-based approaches for shape diameter function computation and its applications focused on skeleton extraction. Computers Graphics, 59, 07 2016.

[15] Lise Noël and Pierre Duysinx. Shape optimization of microstructural designs subject to local stress constraints within an xfem-level set framework. Structural and Multidisciplinary Optimization, 55, 06 2017.

[16] Stanley Osher and James A Sethian. Fronts propagating with curvature-dependent speed: Algorithms based on hamilton-jacobi formulations. Journal of Computational Physics, 79(1):12–49, 1988.

[17] Xavier Rolland-Nevière, Gwenaël Doërr, and Pierre Alliez. Robust diameter-based thickness estimation of 3d objects. Graphical Models, 75(6):279–296, 2013.

[18] Ariel Shamir and Daniel Cohen-Or. Consistent mesh partitioning and skeletonisation using the shape diameter function. The Visual Computer, 24:249–259, 04 2008.

[19] Julien Tierny, Jean-Philippe Vandeborre, and Mohamed Daoudi. 3d mesh skeleton extraction using topological and geometrical analyses. pages 85–94, 10 2006.

[20] Carlos Villanueva and Kurt Maute. Density and level set-xfem schemes for topology optimization of 3-d structures. Computational Mechanics, 54, 01 2014.

[21] Yiqiang Wang, Lei Zhang, and Michael Yu Wang. Length scale control for structural optimization by level sets. Computer Methods in Applied Mechanics and Engineering, 305:891–909, 2016.

[22] Tongxing Zuo, Haitao Han, Qianglong Wang, Qiangwei Zhao, and Zhenyu Liu. An explicit topology and thickness control approach in simp-based topology optimization. Computers Structures, 307:107631, 2025.

[23] Dennis Zvegincev. A tikhonov approach to level set curvature computation. 03 2022.