SEQUENTIAL METRIC-BASED ADAPTIVE MESH GENERATION

Christos Tsolakis, Fotis Drakopoulos and Nikos P. Chrisochoides

CRTC Lab Computer Science Department
School of Sciences
Old Dominion University
{ctsolakis,fdrakopo,nikos}@cs.odu.edu

ABSTRACT

In this work, CDT3D an in-house developed isotropic mesh generator is extended to include metric based mesh adaptation. Mesh operations perform the same topological transformations but all criteria are evaluated in a transformed space. Details on the required modifications at each meshing step are included as well an evaluation of the new method on metric functions given in analytic form.

Keywords: Anisotropic Mesh Generation, Metric Adaptation, Mesh Adaptation

1 INTRODUCTION

Generating anisotropic meshes translates into generating a mesh in which the desired sizing depends on the direction. In many computational areas including both areas of Finite Elements Analysis and visual graphics, there are cases where the function to be interpolated exhibits high variation in only one direction. For example, meshing the surface of a cylinder with elongated elements aligned with the axis of the cylinder can produce a surface mesh with with high fidelity and low number of elements at the same time. In contrast, creating a surface mesh only with isotropic elements will require a significant higher number of elements in order to achieve the same fidelity. In the context of Computation Fluid Dynamics (CFD) simulations, the traditional approach to simulations is to generate a mesh which will be passed to a solver in order to approximate the solution of the problem. After, a number of iterations the solver will request a new mesh from the mesh generation component while providing information about the error of the solution on the previous mesh. For reasons, similar to the above example it has been found that aligning elements taking into consideration the variation of the solution in each direction can yield a lower interpolation error with fewer elements.

CDT3D (Drakopoulos 2017, Drakopoulos, Tsolakis, and Chrisochoides 2017) is designed to be the speculative component of the telescopic approach to mesh generation presented in (Chrisochoides 2016). However, parallel performance is only one part of the numerous needs of the meshing community, offering functionality that meets the needs of the CFD community is another one. In this paper, the mesh generation algorithm is extended so it can handle the feedback coming from the solver as described above. In the presented work, input to the algorithm is not only the surface mesh as before but also an error estimator at each point in the form of a matrix. The goal is to use this information in order to create a mesh that adapts to the provided information as good as possible. While only the sequential version of the algorithm has been implemented and tested, all the modifications of the code relate to local evaluations, so extending the method in the future, to the parallel version of CDT3D should be straightforward.

1.1 Related Work

One of the first attempts to create anisotropic elements appears in (Peraire, Vahdati, Morgan, and Zienkiewicz 1987), where the authors extract information about the error of the evaluation solution of the PDE and encapsulate it into an error estimator. The estimator is then used to guide their advancing front point creation method by producing a spacing that suites the needs of the problem.

The first use of the 2D Delaunay method to create anisotropic elements appears in (Mavriplis 1990). The author captures anisotropy via a two-dimensional control surface in the three-dimensional space. The control surface is then linearly discretized and a stretch vector for each linear component is evaluated. Local operations like Delaunay cavity evaluation are modified in order to correspond to isotropic evaluations on the high order surface.

Creating anisotropic meshes with the Delaunay method was revisited in (George, Hecht, and Vallet 1991), but this time the error estimator was provided in the mesh generation algorithm in a form of a symmetric positive definite matrix that can induce a metric operator. This representation combines naturally with notions of differential geometry as it is presented below.

Moving forward to present, there is a plethora of mesh adaptation software that uses this representation and properly modified meshing operators in order to produce anisotropic meshes. A non-exhaustive list includes, `Bamg` (Hecht 2018) in two dimensions `mmgs` for surface meshes and `mmg3d` for volume meshes in three dimensions (MMG developers 2018, Dobrzynski and Frey 2009) `Feflo.a` from INRIA (Loseille and Löhner 2009), `refine` from NASA, (Park 2018, Park and Darmofal 2008), `Omega_h` (Ibanez 2018, Ibanez and Shephard 2016), `pragmatic` (Gorman 2018, Gorman, Rokos, Southern, and Kelly 2015) and `EPIC` from Boeing (Michal, Babcock, Kamenetskiy, Krakos, Mani, Glasby, Erwin, and Stefanski 2017, Michal and Krakos 2012).

2 METRIC SPACES IN THE CONTEXT OF MESH GENERATION

The proposed method uses the notion of the Euclidean and Riemannian metric spaces, as models for the transformed space.

Informally, the Euclidean metric space can be thought as a transformed space near the vicinity of a point. In the context of mesh generation, the mesh in the vicinity of the point is transformed through affine transformations in order to create an isotropic mesh in the transformed space. Selecting the transformations appropriately, results in a mesh that exhibits the desired sizing in each direction in the physical space. The notion of Riemannian metric space can be seen as a global metric space which is locally a Euclidean metric space and varies smoothly within the domain.

A short reference of the most relevant to this work definitions follows. For a complete treatment see (Loseille and Alauzet 2011).

Definition 1 (Positive Definite Matrix)**.** *A real symmetric* 3×3 *matrix* M *is called positive-definite if* $\forall u \in \mathbb{R}^3, u \neq 0 \implies u^T M u > 0$.

Remark: For a positive-definite matrix $\mathcal{M}$, the bilinear function $\langle u, v \rangle_{\mathcal{M}} := u^T \mathcal{M} v$ defines an inner product in $\mathbb{R}^3$.

Remark: It can be shown that every inner product on a real vector space can induce a vector space norm by the map $\|u\|_{\mathcal{M}} = \sqrt{\langle u, u \rangle_{\mathcal{M}}}$ and subsequently a metric via the formula $d_{\mathcal{M}}(x,y) = \|x-y\|_{\mathcal{M}}$.

The pair $(\mathbb{R}^3, \mathcal{M}) := (\mathbb{R}^3, d_{\mathcal{M}})$ consists therefore a metric space. For the rest of the paper it will be called a **Euclidean Metric Space** or simply a **Euclidean Space**. In a metric space, notions like length, angle, area and volume which are of particular interest in mesh generation can be defined as usual by substituting the usual metric with the one induced by the positive-definite matrix $\mathcal{M}$:

$$\text{length of a segment} \qquad \ell_{\mathcal{M}}(x,y) = d_{\mathcal{M}}(x,y) \tag{1}$$

$$\text{angle between non-zero vectors} \qquad \cos(\theta_{\mathcal{M}}) = \frac{\langle u, v \rangle_{\mathcal{M}}}{\|u\|_{\mathcal{M}} \|v\|_{\mathcal{M}}}, \quad \theta_{\mathcal{M}} \in [0, \pi] \tag{2}$$

Although these two functions can be used to evaluate most of the required quantities during mesh generation, a general transformation map from physical space $(\mathbb{R}^3, \mathcal{I})$ to the Euclidean space $(\mathbb{R}^3, \mathcal{M})$ would be useful both for speeding up calculations and for allowing more complicated expressions. The first step to derive this map is given by the spectral theorem of linear algebra:

Theorem 1 (Spectral Decomposition)**.** *If $M \in \mathbb{R}^{n \times n}$ is a symmetric matrix then it can be factorized as $M = PDP^T$ where P is a orthogonal matrix whose columns are eigenvectors of M and $D := \operatorname{diag}(d_1, d_2, \ldots, d_n)$ the diagonal matrix of the eigenvalues of M.*

The spectral decomposition enables the definition of the map

$$\begin{aligned} D^{\frac{1}{2}} P^T : \quad & (\mathbb{R}^3, \mathcal{I}) \to (\mathbb{R}^3, \mathcal{M}) \\ & x \mapsto D^{\frac{1}{2}} P^T x, \qquad \text{where } D^{\frac{1}{2}} := \operatorname{diag}\left(\sqrt{d_1}, \sqrt{d_2}, \sqrt{d_3}\right) \end{aligned}$$

that maps a vector from the physical space to the Euclidean space induced by $\mathcal{M}$. Using the above map in conjunction with common geometric formulas, one can transform results from physical space to the Euclidean space, for example using the algebraic properties of the cross product one gets:

$$x \times_{\mathcal{M}} y = \left(D^{\frac{1}{2}} P^T x\right) \times \left(D^{\frac{1}{2}} P^T y\right) = \det\left(D^{\frac{1}{2}} P^T\right) \cdot \left(\left(D^{\frac{1}{2}} P^T\right)^T\right)^{-1} (x \times y) = \sqrt{\det \mathcal{M}} \left(D^{-\frac{1}{2}} P^T\right) (x \times y)$$

since, $PP^T = \mathcal{I}$. Finally, using the cross product the volume of a parallelepiped K in the Euclidean metric space can be evaluated $\mathbb{R}^3$:

$$|K|_{\mathcal{M}} = \langle x, (y \times_{\mathcal{M}} z) \rangle_{\mathcal{M}} = \sqrt{\det \mathcal{M}} \cdot |K|_{\mathcal{I}_3} \tag{3}$$

where $|K|_{\mathcal{I}_3}$ is the volume of the parallelepiped in the physical space.

2.1 Riemannian Metric space

In the case of the Euclidean space the metric tensor is a constant matrix, if one allows the metric tensor to vary smoothly over the computational domain, then the **Riemannian Metric space** or simply **Riemannian space** is defined. Even though in a Riemannian space there is no global definition of the inner product, it is still a useful construct for mesh adaptation purposes since in most applications the metric varies within the domain. First, in order to be able to define a metric tensor for every point of the domain an interpolation scheme on the input background mesh must is needed.

In this work, the Log-Euclidean framework introduced in (Arsigny, Fillard, Pennec, and Ayache 2006) is used as interpolation scheme. One of the main advantages of this scheme over the previous used in the literature is that the interpolation operator is commutative simplifying thus the interpolation procedure during mesh generation. Moreover, in (Michal and Krakos 2012) the Log-Euclidean framework was found to be superior in comparison to the power average and the simultaneous matrix reduction method in various numerical evaluations.

2.2 Metric Interpolation Log-Euclidean Framework

Let x_i , $i = 1 \dots k$ be a set of vertices and $\mathcal{M}_i = (\mathcal{M}(x_i))$ their corresponding metrics. Then, for a point x of the domain with barycentric coordinates a_i :

$$x = \sum_{i=1}^{k} a_i \cdot x_i, \quad \text{with} \quad \sum_{i=1}^{k} a_i = 1$$

the interpolated metric is defined by:

$$\mathcal{M}(x) = \exp\left(\sum_{i=1}^{k} a_i \ln \mathcal{M}_i\right) \tag{4}$$

Note that since $\mathcal{M}_i$ is positive definite it has positive eigenvalues therefore the exponential and logarithm of the metric are well defined and given by

$$\ln(\mathcal{M}) := P \ln(D) P^T \quad \exp(\mathcal{M}) := P \exp(D) P^T$$

Where $\mathcal{M} = PDP^T$ is the spectral decomposition of $\mathcal{M}$.

2.3 Computation of Edge Length in Metric Space

The length of a segment in an adapted mesh plays critical role in the quality of the mesh and it is one of the metrics used in this work to quantify the quality of the generated mesh. For this reason an accurate estimate is needed. In differential geometry the length of a vector $\mathbf{u} = xy$ is computed using the integral

$$\ell_{\mathcal{M}}(xy) = \int_0^1 \|\gamma'(t)\|_{\mathcal{M}} \, \mathrm{d}t = \int_0^1 \sqrt{\mathbf{u}^T \mathcal{M}(x + t\mathbf{u})\mathbf{u}} \, \mathrm{d}t \quad \text{where } \gamma(t) = x + t \cdot \mathbf{u}$$

However, this computational expensive evaluation can be approximated using a variational law presented in (Alauzet 2010) :

Definition 2. *Let $e = p_1 p_2$ be an edge of the mesh and $\mathcal{M}_1 := \mathcal{M}(p_1)$, $\mathcal{M}_2 := \mathcal{M}(p_2)$ the corresponding metric tensor on the endpoints of the edge, Without loss of generality assume that $\ell_{\mathcal{M}_1}(e) > \ell_{\mathcal{M}_2}(e)$ and define $a = \frac{\ell_{M_1}(e)}{\ell_{M_2}(e)}$ then,*

$$\ell_{\mathcal{M}}(e) = \ell_{\mathcal{M}_1}(e) \frac{a-1}{a \ln(a)} \tag{5}$$

2.4 Computation of Element Volume

Finally, many quality measures and validity tests in mesh generation use the value of the volume of a tetrahedron, therefore a formula for the computation of element volume is needed. Formally, in a Riemannian space the volume is evaluated using a continuous version of formula (3). However, in order to avoid the expensive calculations the same approach as in (Alauzet and Marcum 2017) is used, that is, interpolating the metric on the centroid of the element and using the Euclidean volume formula (3).

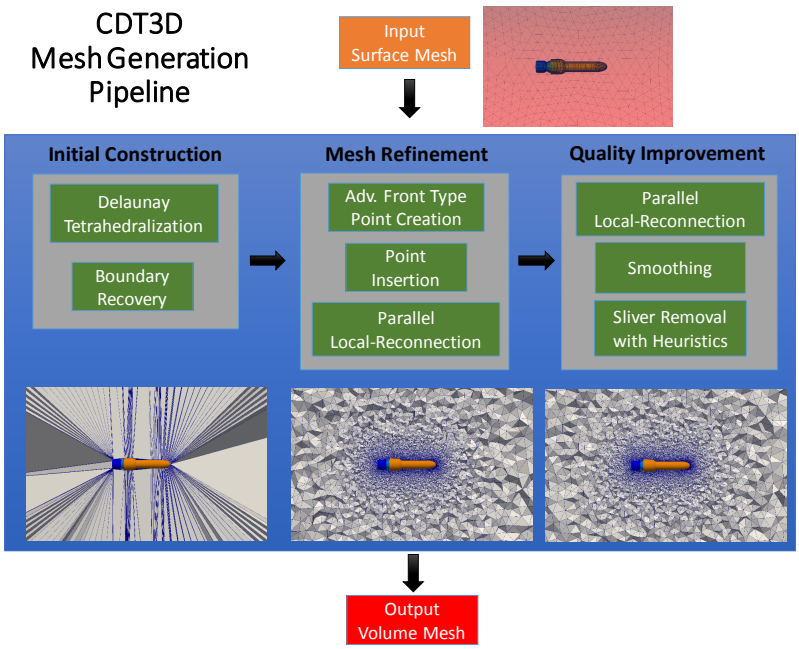

Figure 1: High level mesh generation pipeline of the CDT3D software.

3 CDT3D

3.1 Isotropic Mesh Generation

The presented work is based on the isotropic CDT3D mesh generation software (Drakopoulos 2017, Drakopoulos, Tsolakis, and Chrisochoides 2017). The pipeline of CDT3D can be divided into three parts Initial Mesh Construction, Mesh Refinement and Mesh Optimization (see Figure 1).

The first step creates a Delaunay mesh of the input boundary points and Boundary recovery modifies the Delaunay mesh in order to contain all the initial edges and faces of the input surface triangulation with no or very low number of Steiner points.

Mesh Refinement inserts new points into the mesh until the desired point distribution function is satisfied. New points are created using advancing-front point type placement and direct insertion is used to introduce them into the mesh. A local reconnection step containing edge and face flips is performed in parallel optimizing a combined Delaunay and Min-Max type (maximization of the minimum Laplacian edge weight) criterion. These three steps are performed repetitively until no point is inserted.

During the final stage of the pipeline, the mesh quality is improved repetitively using a combination of vertex smoothing, local reconnection and point insertion with heuristics to remove isolated bad shape elements.

3.2 Anisotropic Mesh Generation

The adaptive version of the code follows the same steps as in figure 1 but performs all the evaluations of distances, angles and volumes using the formulas presented in section 2. Vertex Insertion remains unmodified since it uses direct insertion which modifies the mesh using only mesh connectivity information. Sliver removal has not been implemented in the adaptive version since the heuristics used, depend heavily on the uniformity of the physical space. The rest of the steps are modified as follows:

Vertex Creation During vertex creation, between each pair of active and inactive tetrahedra a candidate point is generated by advancing from the common face in a distance that will create a equilateral element based on an appropriate length scale. Since in the adapted version the quality is specified by the metric tensor, the distance is evaluated in the metric space. More specifically, a candidate point P is created on the centroid of the face between the two tetrahedra. The metric is interpolated on P producing $\mathcal{M}_P$. P is then advanced perpendicular to the face by a unit distance as measured by the Euclidean length (equation (1)) using $\mathcal{M}_P$. As with the isotropic case P will be rejected if it is too close to the boundary, to existing vertices or to other candidate points. For evaluating the distances in this case formula (5) is used, since all the pre-existing vertices and candidate points have already a metric tensor assigned to them.

Local Reconnection Local Reconnection consists of modifying the connectivity of the mesh while optimizing some criteria. In CDT3D, 2-3,3-2 and 4-4 flips are used coupled with the in-sphere or the Min-Max criterion. To introduce anisotropy to these calculations the criteria have been adapted as follows:

The in-sphere test of a point y against a tetrahedron $K = (x_1, x_2, x_3, x_4)$ is replaced with the weighted one presented in (Dobrzynski and Frey 2009) :

$$\begin{cases} \alpha_{\mathcal{M}_y}(y,K) < 1 \\ \sum_{i=1}^{4} \alpha_{\mathcal{M}_{x_i}}(y,K) + \alpha_{\mathcal{M}_Q}(y,K) < 5 \end{cases} \quad \text{where} \quad \begin{array}{r} \alpha_{\mathcal{M}_R}(R,K) = \frac{\|O_K y\|_{\mathcal{M}_R}}{\ell_{\mathcal{M}_R}(r_K)} \\ O_K \text{ circumcenter of } K \\ r_K \text{ circumradius of } K \end{array}$$

In isotropic configurations evaluating the Laplacian Edge Weight for an element K consists in evaluating the following quantity:

$$Q(K) = \max_{i=1\ldots6} \frac{\langle n_{F_{i,1}}, n_{F_{i,2}} \rangle}{6|K|}$$

where $n_{F_{i,1}}, n_{F_{i,2}}$ are the 2 faces attached to the i-th edge.

For the anisotropic case, the formula for $Q(K)$ is adapted using a metric tensor $\mathcal{M}$ evaluated on the centroid of K. Moreover, a formula that uses only inner products is used (Marcum and Alauzet 2013), to avoid the expensive evaluation of cross products:

$$Q_{\mathcal{M}}(K) = \max_{i=1\ldots6} \frac{\langle e_i, e_j \rangle_{\mathcal{M}} \cdot \langle e_i, e_k \rangle_{\mathcal{M}} - \langle e_i, e_i \rangle_{\mathcal{M}} \cdot \langle e_j, e_k \rangle_{\mathcal{M}}}{6|K|_{\mathcal{M}}} \tag{6}$$

Vertex Smoothing One of the most effective parts of mesh quality improvement in CDT3D is the one of mesh smoothing. In this work, the optimal point placement smoothing algorithm was used. The optimal point placement repositions each vertex of the mesh to the average of k optimal positions, where k is the number of boundary faces. In the metric space the optimal position for a face with vertices a, b, c is evaluated using:

$$E_{abc} = C_{abc} + \sqrt{\frac{2}{3}} \cdot \frac{\vec{n}}{\|\vec{n}\|_{\mathcal{M}}} \cdot \frac{1}{3} \left(\|\vec{ab}\|_{\mathcal{M}} + \|\vec{bc}\|_{\mathcal{M}} + \|\vec{ca}\|_{\mathcal{M}} \right)$$

where C_{abc} is the centroid and $\vec{n}$ the normal of the face. Similar to the isotropic case, the final position of the point is found by evaluating the average out of the k optimal points.

4 RESULTS

The adaptive algorithm was evaluated on metric tensors given as analytical functions. During the evaluation of the algorithm it became apparent that if the boundary mesh is also adapted then the final mesh has better

quality. For this reason, the open source MMGS library (Dobrzynski and Frey 2009, Dapogny, Dobrzynski, and Frey 2014, MMG developers 2018) was used to adapt the input surface mesh to the provided metric as pre-processing step.

In this work the quality of the meshes is assessed using the edge lengths of the tetrahedra as evaluated by the metric and the mean ratio (Liu and Joe 1994) in the metric space:

$$\eta(K) = \frac{12(3|K|_{\mathcal{M}})^{\frac{2}{3}}}{\sum\limits_{i=1\ldots6} \ell^2_{\mathcal{M}}(e_i)}$$

where $\mathcal{M}$ is the metric tensor interpolated at the centroid of the tetrahedron. The constants in this formula are selected in a way that the mean ratio of an ideal element is 1.0. For the edge lengths, since the goal is to create a mesh as uniform as posible in the metric space, ideal value of edge length is also 1.0.

The qualitative statistics was generated using the `ref_histogram.sh` script of the open source `refine` software (Park 2018).

4.1 Linear Metric field in unit cube

The fist example is a simple linear function with anisotropic stretching centered on the plane $z = 0$ (Ibanez, Barral, Krakos, Loseille, Michal, and Park 2017). The analytic form is:

$$\mathcal{M}(x,y,z) = \begin{pmatrix} \frac{1}{h_x^2} & 0 & 0 \\ 0 & \frac{1}{h_y^2} & 0 \\ 0 & 0 & \frac{1}{h_z^2} \end{pmatrix}$$

where, $h_x = 0.1$, $h_y = 0.1$, $h_0 = 0.001$, $h_z = h_0 + 2(0.1 - h_0)|z - 0.6|$,

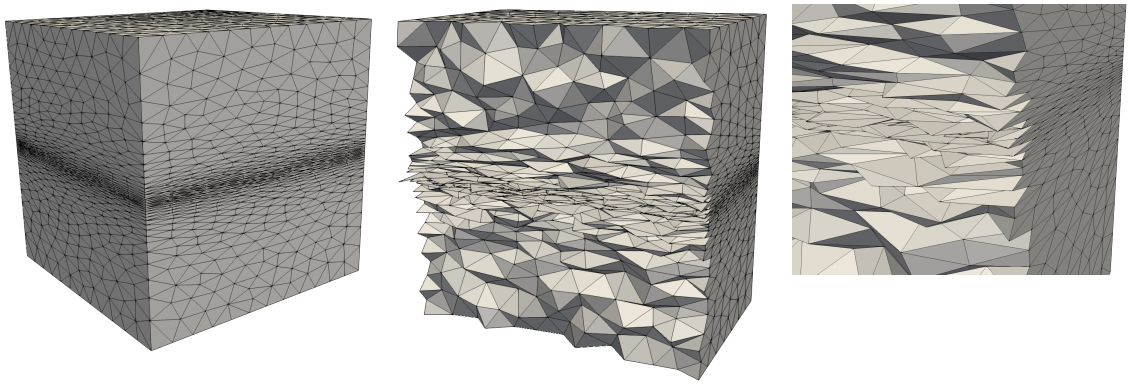

Figure 2: Surface Mesh, cross-cut and detail of the generated volume mesh.

4.2 Quarter Cylinder Metric Function

In the second example curvature is introduced. The domain is again the unit cube, the function enforces a stretching around a cylinder along the z-axis. The analytic formula is taken from (Alauzet and Marcum

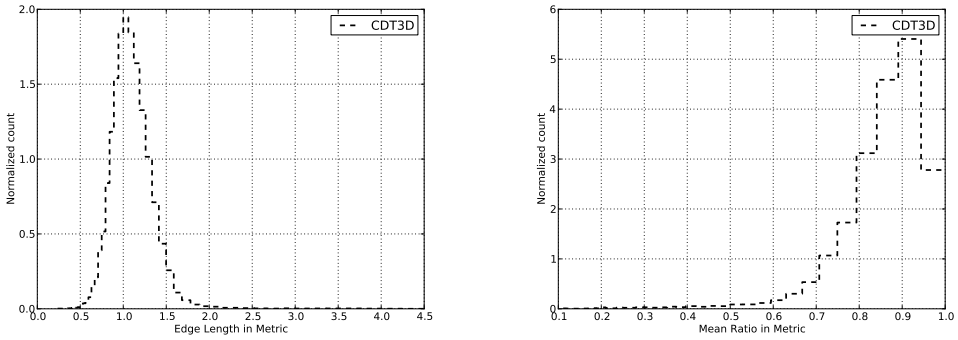

Figure 3: Quality statistics in the metric space for generated volume mesh. 94% of the edges has length between $[0.5, 1.5]$. For the mean ratio 96% of the elements have quality higher that 0.7.

2017):

$$\mathcal{M}(x,y,z) = \begin{pmatrix} \frac{1}{h_x^2}\cos^2\theta + \frac{1}{h_y^2}\sin^2\theta & \left(\frac{1}{h_x^2} - \frac{1}{h_y^2}\right)\cos\theta\sin\theta & 0 \\ \left(\frac{1}{h_x^2} - \frac{1}{h_y^2}\right)\cos\theta\sin\theta & \frac{1}{h_x^2}\sin^2\theta + \frac{1}{h_y^2}\cos^2\theta & 0 \\ 0 & 0 & h_z \end{pmatrix}$$

where, $h_x = \min(0.002 \cdot 5^a, h_{max})$, $h_y = \min(0.05 \cdot 2^a, h_{max})$, $h_z = h_{max}$, $h_{max} = 0.1$, $\theta = \arctan(x,y)$, $a = 10 \cdot |0.75 - \sqrt{x^2+y^2}|$

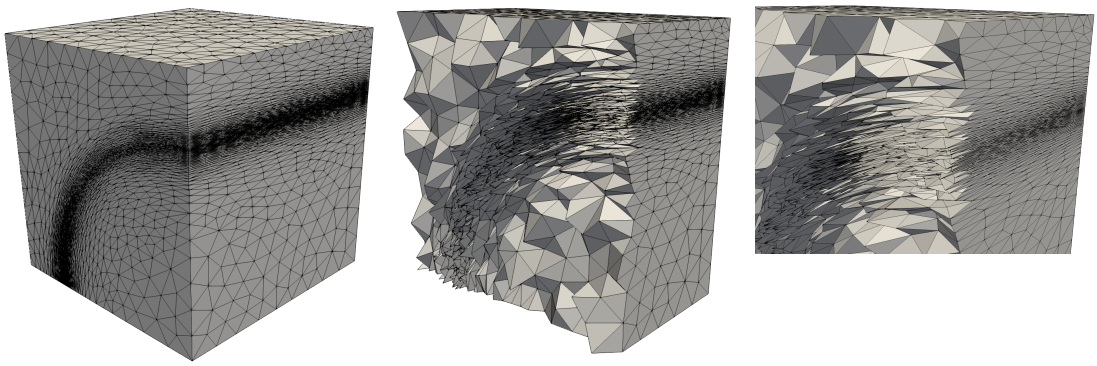

Figure 4: Surface Mesh, cross-cut and detail of the generated volume mesh.

In both examples, the quality distribution is as expected. There is though a small number of low quality elements in the mean ratio quality metric as well as some edges longer as they should which can deteriorate the performance of the subsequent solver. Future versions of the algorithm will target on eliminating those elements.

5 CONCLUSION

In this work, CDT3D, an in-house isotropic mesh generator was modified to support mesh adaptation based on a metric tensor. The modifications on the code were minimal. Topological transformations remain the same but qualitative evaluations are adapted in two ways. Whenever a local evaluation is needed, the Euclidean formulas are used based either on the metric tensor of the point itself or on the metric tensor of an interpolated point. On the other hand, when the length of a segment is needed the Riemannian framework

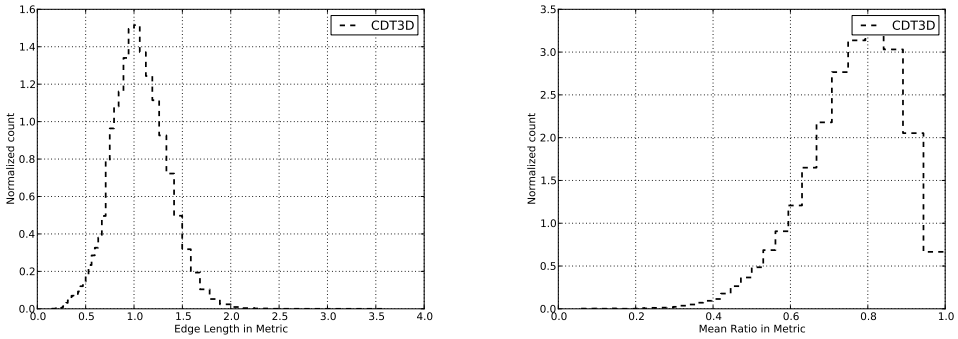

Figure 5: Quality statistics in the metric space for generated volume mesh. 91% of the edges has length between $[0.5, 1.5]$. For the mean ratio 79% of the elements have quality higher that 0.7.

was used in order to take into consideration the metric of both endpoints. Initial results on analytic metrics look promising although there is room for improvement:

To eliminate long edges an additional step could be added targeting only on those or alternatively the edge/face flip criteria could be modified to take into account not only the element quality as a whole but also the lengths of the individual edges. Additionally, it has been shown that metric orthogonal strategies for generating metric adapted meshes offer better quality (Alauzet and Marcum 2017, Loseille 2014). Given the modularity of CDT3D this point creation method could be created with low effort.

In contrast to the provided examples, in most simulation settings the analytic form of the metric is not available. In these cases metric information is given in a form of a coarse background mesh. Although, metric interpolation is already implemented interpolating from a background mesh requires some extra steps that need to be implemented.

REFERENCES

Alauzet, F. 2010. "Size Gradation Control of Anisotropic Meshes". *Finite Elements in Analysis and Design* vol. 46, pp. 181–202.

Alauzet, F., and D. Marcum. 2017. "Metric-Aligned and Metric-Orthogonal Strategies in AFLR". In *23rd AIAA Computational Fluid Dynamics Conference*. Denver,Colorado, American Institute of Aeronautics and Astronautics.

Arsigny, V., P. Fillard, X. Pennec, and N. Ayache. 2006, aug. "Log-Euclidean Metrics for Fast and Simple Calculus on Diffusion Tensors". *Magnetic Resonance in Medicine* vol. 56 (2), pp. 411–421.

Chrisochoides, N. P. 2016. "Telescopic Approach for Extreme-Scale Parallel Mesh Generation for CFD Applications". In *46th AIAA Fluid Dynamics Conference*. American Institute of Aeronautics and Astronautics.

Dapogny, C., C. Dobrzynski, and P. Frey. 2014. "Three-dimensional adaptive domain remeshing, implicit domain meshing, and applications to free and moving boundary problems". *Journal of Computational Physics* vol. 262, pp. 358 – 378.

Dobrzynski, C., and P. Frey. 2009. "Anisotropic Delaunay Mesh Adaptation for Unsteady Simulations". In *Proceedings of the 17th International Meshing Roundtable*, edited by R. V. Garimella, pp. 177–194. Springer Berlin Heidelberg.

Drakopoulos, F. 2017. *Finite Element Modeling Driven by Health Care and Aerospace Applications*. Ph. D. thesis, Computer Science, Old Dominion University.

Drakopoulos, F., C. Tsolakis, and N. P. Chrisochoides. 2017. "Fine-Grained Speculative Topological Transformation Scheme for Local Reconnection Methods". In *AIAA Aviation Forum*, pp. accepted.

George, P. L., F. Hecht, and M. G. Vallet. 1991. "Creation of Internal Points in Voronoi's Type Method. Control Adaptation". *Advances in Engineering Software and Workstations* vol. 13 (5), pp. 303–312.

Gorman, Gerard J. 2018. "pragmatic GitHub site". https://meshadaptation.github.io.

Gorman, G. J., G. Rokos, J. Southern, and P. H. J. Kelly. 2015. "Thread-Parallel Anisotropic Mesh Adaptation". In *New Challenges in Grid Generation and Adaptivity for Scientific Computing*, SEMA SIMAI Springer Series, pp. 113–137. Springer, Cham.

Hecht, Frédéric 2018. "BAMG - Bidimensional Anisotropic Mesh Generator".

Ibanez, Dan 2018. "Omega_h GitHub site". https://github.com/ibaned/omega_h.

Ibanez, D., N. Barral, J. Krakos, A. Loseille, T. Michal, and M. Park. 2017, January. "First Benchmark of the Unstructured Grid Adaptation Working Group". *Procedia Engineering* vol. 203, pp. 154–166.

Ibanez, D., and M. Shephard. 2016. "Mesh Adaptation for Moving Objects on Shared Memory Hardware". *Procedia Engineering*, pp. 5.

Liu, A., and B. Joe. 1994, June. "Relationship between Tetrahedron Shape Measures". *BIT Numerical Mathematics* vol. 34 (2), pp. 268–287.

Loseille, A. 2014, January. "Metric-Orthogonal Anisotropic Mesh Generation". *Procedia Engineering* vol. 82 (Supplement C), pp. 403–415.

Loseille, A., and F. Alauzet. 2011. "Continuous Mesh Framework Part I: Well-Posed Continuous Interpolation Error". *SIAM Journal on Numerical Analysis* vol. 49 (1), pp. 38–60.

Loseille, A., and R. Löhner. 2009. "On 3D Anisotropic Local Remeshing for Surface, Volume and Boundary Layers". In *Proceedings of the 18th International Meshing Roundtable*, pp. 611–630. Springer, Berlin, Heidelberg.

Marcum, D. L., and F. Alauzet. 2013. "Unstructured Mesh Generation Using Advancing Layers and Metric-Based Transition for Viscous Flowfields". In *21st AIAA Computational Fluid Dynamics Conference*. American Institute of Aeronautics and Astronautics.

Mavriplis, D. J. 1990, October. "Adaptive Mesh Generation for Viscous Flows Using Triangulation". *Journal of Computational Physics* vol. 90 (2), pp. 271–291.

Michal, T., and J. Krakos. 2012. "Anisotropic Mesh Adaptation Through Edge Primitive Operations". In *50th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition*. American Institute of Aeronautics and Astronautics.

Michal, T. R., D. Babcock, D. S. Kamenetskiy, J. Krakos, M. Mani, R. S. Glasby, T. Erwin, and D. Stefanski. 2017. "Comparison of Fixed and Adaptive Unstructured Grid Results for Drag Prediction Workshop 6". In *55th AIAA Aerospace Sciences Meeting*. American Institute of Aeronautics and Astronautics.

MMG developers 2018. "Mmg Platorm". https://www.mmgtools.org.

Park, Mike 2018. "refine GitHub site". https://github.com/nasa/refine.

Park, M., and D. Darmofal. 2008. "Parallel Anisotropic Tetrahedral Adaptation". In *46th AIAA Aerospace Sciences Meeting and Exhibit*. American Institute of Aeronautics and Astronautics.

Peraire, J., M. Vahdati, K. Morgan, and O. C. Zienkiewicz. 1987, October. "Adaptive Remeshing for Compressible Flow Computations". *Journal of Computational Physics* vol. 72 (2), pp. 449–466.