LAVA Voronoi Mesher for Wall-Modeled Large-Eddy Simulations

Victor C.B. Sousa*, Abram K. Rodgers †, Keshav Sriram‡, Emre Sozer§, Michael F. Barad¶, Gerrit-Daniel Stich ‖, François Cadieux **, and Jared C. Duensing††
NASA Ames Research Center, Moffett Field, CA, 94035

The unstructured Voronoi mesher currently being developed within the Launch, Ascent, and Vehicle Aerodynamics (LAVA) software framework at NASA Ames Research Center is described in detail. The discussions include, but are not limited to, the strategies used in the seeding and smoothing phases to ensure a high-quality mesh for Wall-Modeled Large-Eddy Simulations (WMLES), the methods behind the cell clipping algorithm responsible for conforming the mesh to a complex geometry, and the approach to create a global mesh from the distinct Voronoi cells. Applications and benefits of the Voronoi meshing approach are also presented. The automated meshing paradigm introduced was able to significantly reduce the time necessary to create a high-quality mesh around a complex geometry when compared against the current LAVA curvilinear overset meshing standard. For example, the task of generating a single mesh around the high-lift common research model requires the full dedication of a mesh generation expert for a period between one and two months. A family of meshes with different refinement levels up to six hundred million cells can be generated by a single engineer in a day or two. This technology has the potential for decreasing the turnaround time for conducting WMLES around complex geometries, as well as facilitating comprehensive mesh refinement studies.

I. Introduction

The rapid growth in computational resources in the High-Performance Computing area has enabled Wall-Modeled Large-Eddy Simulations (WMLES) as a tool for practical applications in industry. Recent work has demonstrated the capability of WMLES to accurately predict quantities of engineering interest in flows involving complex geometries. For example, in the field of aircraft aerodynamics, WMLES has been used to study the flow field around a model of a commercial airplane in high-lift configurations near take-off and landing [1–6] as well as the flow around an airfoil with horn ice accretion on the leading edge [7, 8]. Moreover, WMLES has been successfully used to predict noise levels generated by the flow structures around landing-gear [9] and those caused by a dual-stream nozzle in co-axial jets [10–12]. The increase in the interest for WMLES relies mainly on the fact that, for a reasonable additional cost with respect to traditional methods such as Reynolds-Averaged Navier-Stokes (RANS), WMLES is capable of considerably increasing the prediction accuracy of simulations of unsteady flow phenomena such as the dynamics of a shear layer created by a jet or the non-equilibrium characteristics of a separated turbulent boundary layer [13].

One bottleneck that prevents a more widespread usage of WMLES in flow simulations around complex geometries relevant for engineering applications is the challenge of creating appropriate high-quality, generally large-scale, meshes [14]. There are four established approaches, that could be used separately or in conjunction with one another, for mesh generation over complex geometries for WMLES: Cartesian octree grids coupled with an immersed boundary technique, structured curvilinear overset grids, unstructured meshes based on predetermined cell shapes (e.g. hexahedral, tetrahedral or prismatic), and unstructured meshes based on generic polyhedral shapes generated by a Voronoi tessellation.

Out of the previously mentioned methods, unstructured meshes offer distinct advantages that can supersede the speed limitation drawback that comes from additional numerical requirements, such as having to store a neighbor connectivity graph. These advantages include highly automated and scalable body-conforming mesh generation, and

*Science and Technology Corporation, AIAA Member, victor.decarvalhobrittosousa@nasa.gov
†Computational Aerosciences Branch, AIAA Member, abram.k.rodgers@nasa.gov
‡Computational Aerosciences Branch, AIAA Member, keshav.sriram@nasa.gov
§Computational Aerosciences Branch, AIAA Member, emre.sozer@nasa.gov
¶Computational Aerosciences Branch, AIAA Member, michael.f.barad@nasa.gov
‖Science and Technology Corporation, AIAA Member, gerrit-daniel.stich@nasa.gov
**Computational Aerosciences Branch, AIAA Member, francois.cadieux@nasa.gov
††Computational Aerosciences Branch, AIAA Member, jared.c.duensing@nasa.gov

applicability of mesh smoothing techniques that can mitigate coarse-fine cell volume jumps responsible for numerical impedance and spurious reflections. Additionally, these benefits support the primary goal of the current development effort, which is to eliminate the time-consuming manual process traditionally required for generating body-fitted meshes. In comparison against other unstructured mesh topologies, Voronoi grids offer several advantages for WMLES. Firstly, the cells generated by seeding the domain with regular point distributions have faces halfway between cell centers, enabling the use of robust and accurate, non-dissipative flux discretizations. Secondly, the resulting cells exhibit isotropy for complex geometries and allow for control over the minimum cell size, facilitating the use of efficient explicit time integration schemes.

One of the main advantages, in addition to computation speed, of either Cartesian or structured curvilinear overset meshing frameworks over the unstructured approach is the use of finite-difference methods, which can be extended to high orders of accuracy for a small computational overhead. While studies on numerical methods with high-orders of accuracy applied to generic polyhedra exist [15–18], previous work on WMLES using Voronoi meshes have been limited to second-order accurate schemes. There is a growing recognition within the community of the significance of minimizing dissipation errors compared to dispersion errors in high Reynolds number Large-Eddy Simulations (LES) [19]. Consequently, second-order kinetic energy-preserving finite volume discretizations are widely considered to be highly appropriate for computational aerodynamics in most LES applications and studies of the influence of higher-order methods in arbitrary polyhedral meshes are left for future work.

The Voronoi-based meshing process has only been recently applied in the WMLES framework despite its previous successful use in the field of computational star formation [20, 21]. The first application of Voronoi meshes in the context of turbulent flow simulations with engineering applications can be credited to Cascade Technologies and the Center for Turbulence Research (CTR) at Stanford University in 2018 [10]. Since then, CTR's group has successfully conducted numerical reproductions of several flow setups involving complex geometries which include models of cars, airplanes and aircraft rotors [1, 22–26].

In order to encourage collaboration, the objective of this manuscript is to describe the details behind the algorithms used by the Voronoi-based mesher currently under development within the Launch, Ascent, and Vehicle Aerodynamics (LAVA) group at the NASA Ames Research Center. First, strategies for point cloud generation are discussed in Section II.A. The special treatment applied for near-body layers is discussed in Section II.B. The details of the smoothing algorithm currently employed are discussed in Section II.C. After that, the procedure for computing each individual Voronoi cell from a point cloud is considered in Section II.D. Next, the algorithm used to automatically detect geometric features and retain them in the final global mesh is presented in Section II.E. Then, the algorithm used to clip the generated cells with a triangulation that describes the geometry of interest is explained in Section II.F. The procedure for creating a global mesh from the individual Voronoi cells is introduced in Section II.G. Finally, applications for the LAVA Voronoi mesher as well as benefits are presented in Section III.

II. Voronoi Mesh Generation Methodology

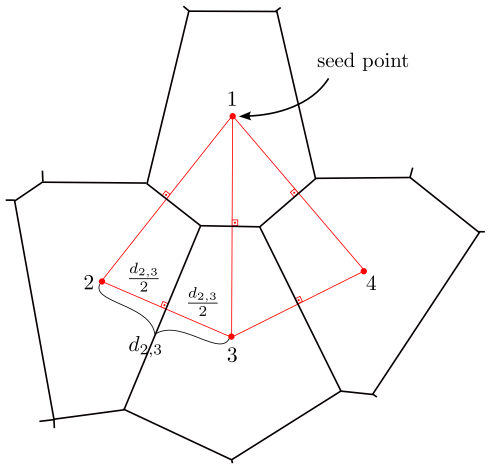

Fig. 1 Example of a 2D cluster of Voronoi cells illustrating desirable properties for numerical computations.

Voronoi meshes possess several intrinsic qualities that make them highly suitable for CFD applications needing high resolution and low dissipation, all the while minimizing the human effort required for mesh generation. A Voronoi diagram is a collection of planar faces dividing the space between points distributed in 3D space. The points are often referred to as seed points or generating locations. The faces enclosing each seed point contain the space that is closer to the seed than any other point. The faces form arbitrary polyhedral shaped cells with three key properties, which are illustrated in Fig. 1:

- Faces are planar, which is not always the case with non-Voronoi arbitrary polyhedral unstructured meshes, such as those resulting from dualization of tetrahedral meshes. The faces are often assumed planar in 2$^{\text{nd}}$ order finite-volume solver implementations.
- Faces are orthogonal to the vector connecting the two generating seeds on either side of the face. This property is distinctly beneficial for the stability of the convective and diffusive flux schemes as described in Sozer *et al.* [27].
- Faces are located midway between the seed points on either side. Therefore, face-averaging can be performed through a straightforward arithmetic average of the cell center values on either side of the face. Since face-averaging is used in gradient calculations and diffusive flux discretization, these operations benefit from truncation error cancellation which leads to a non-dissipative central scheme [28]. Regular seeding patterns are especially benefitted from this property.

All of these properties are major factors for improving the accuracy and robustness of finite volume solvers. This means that if one is careful about using the seed locations and the vectors that connect neighboring seeds, exact cancellation of truncation errors should be recovered in the interpolation and differentiation algorithms even for irregular Voronoi meshes. A more in-depth analysis of the numerical operators on unstructured Voronoi meshes is planned and will be addressed in future publications.

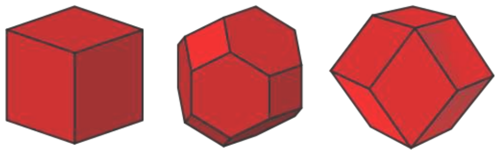

Fig. 2 Cell-type examples: hexahedron (left), truncated octahedron (middle), rhombic dodecahedron (right)

In practice, it is beneficial to use regular patterns in large portions of the generator point cloud. That way, it is possible to create regions of isovolumetric centroidal Voronoi cells that have inherent well-behaved numerical operators dependent on a regular stencil of neighboring cells. Sozer *et al.* [27] has discussed the advantages and disadvantages of using three different regular lattice arrangements: Cartesian seeding, which generated cubic cells; body centered cubic (BCC) seeding, generating truncated octahedron cells; and face centered cubic (FCC) seeding, generating rhombic dodecahedron cells (see Fig. 2). It was found that BCC grid arrangement was a good compromise between the number of face evaluations and final accuracy of the results.

Additionally, it is cost-effective to increase the mesh spacing with distance from the solid boundaries of the geometry. This happens because the length scales associated with the dynamics of the fluid motion also grow with distance from the walls. The use of telescoping regions with seeds of increased spacing can meet both constraints. The caveat is that an abrupt coarse-fine interface will occur at the borders between regions if no extra treatment is performed. The presence of such an interface has been shown to be detrimental to numerical computations of compressible fluid flow because it can create an artificial impedance boundary causing spurious wave reflections and dampening [29]. A Lloyd-smoothing algorithm is employed to mitigate these abrupt changes in cell volumes (see Section II.C). Figure 3 provides an example of the use of telescoping refinement regions when meshing, as well as the effect of Lloyd-smoothing iterations on the final cell distribution.

The following section provides a detailed discussion on the steps involved in creating a clipped Voronoi tessellation around a complex geometry, tailored for WMLES. It is important to note that the entire Voronoi meshing process relies solely on a watertight surface triangulation that defines the geometry of interest, along with a straightforward input file specifying areas requiring refinement.

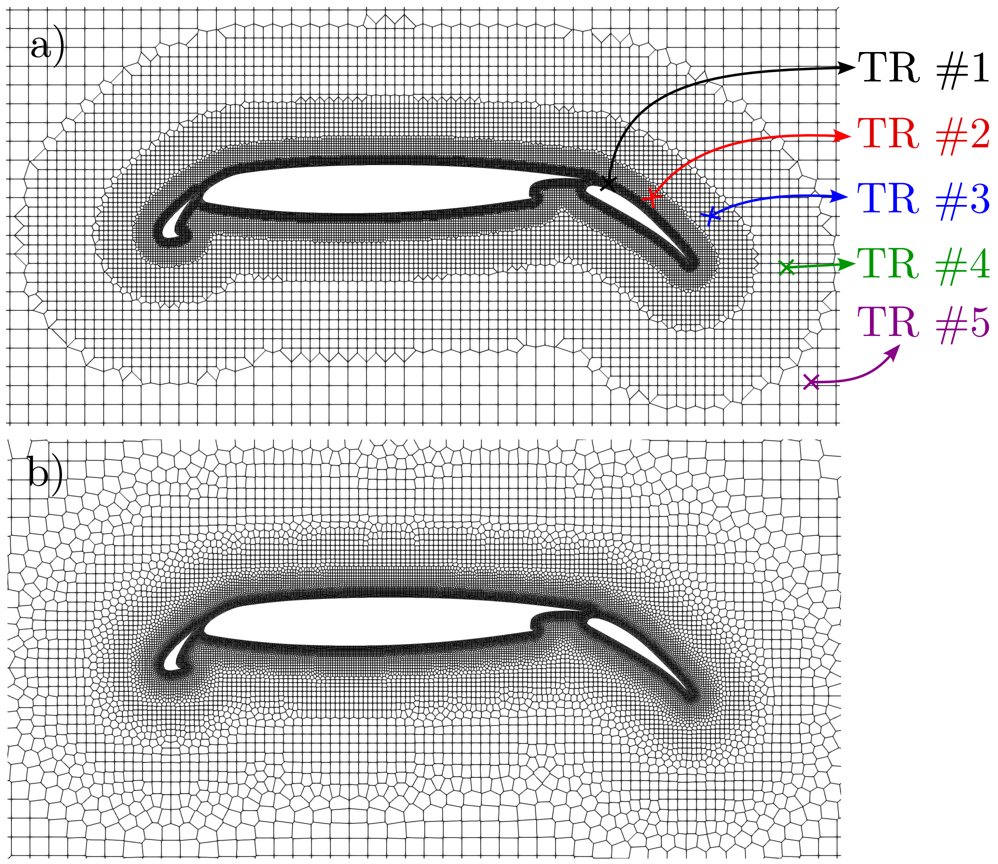

Fig. 3 Sample (a) unsmoothed and (b) smoothed Voronoi meshes around a multi-element airfoil created using the concept of telescoping regions (TRs). In this mesh subset, five TRs are shown.

A. Point Cloud Seeding

For the initialization of the Voronoi diagram, we first construct a cloud of points in a process we refer to as *seeding*. This results in a list of $s_i \in \mathbb{R}^3$ where the seeds are the generating locations of the Voronoi cells. In the seeding algorithm, we cover the seeding region Ω (assumed rectangular) with closed sets $R_i \subset \Omega$, named regions, which are characterized by their unique seed spacing δ_i.

The partition of the seeding domain is used to allow the user to choose the characteristic cell size in different areas of Ω. We define two main types of regions: 1) a near-surface region, defined by a distance from the triangulation that defines the geometry of interest; and 2) a bulk refinement region, defined by analytical shapes such as rectangular prisms, conical frusta, cylinders, or spheres. Additionally, a telescoping functionality is built in within each region such that automatic and gradual coarsening of the mesh with distance from the shape can be achieved at a certain rate chosen by the user. This is especially useful to create body-aligned off-body coarsening.

In some cases, it is beneficial to define the thickness of each subsequent telescoping near-surface region as a function of the number of points and seed spacing desired in each layer. The meshes in Fig. 4, for example, have regions with 8 points each that double in size with each additional layer. The main benefit of using such an arrangement is observed when trying to generate a family of meshes with increasing resolution for a grid sensitivity study. When the discussed strategy is in place, a decrease in the near-surface length scale by half leads to an approximate increase of the total number of cells by 4 (in 3D). This is in contrast with an increase by a factor of 8 that would happen if fixed distance layers were used instead. Using the combination of number of points and seed spacing effectively decreases the thickness of each wall-normal layer when a resolution increase occurs. While this approach is not a consistent global refinement, it is effective for accuracy and numerical cost control when resolution increase is mainly needed near wall surfaces to better represent boundary layers and wall-generated turbulence.

After the regions are defined, they are ordered from finest to coarsest with respect to seed spacing. This ordering is used to determine priority. When regions of similar spacing overlap, the union of such regions should occur seamlessly

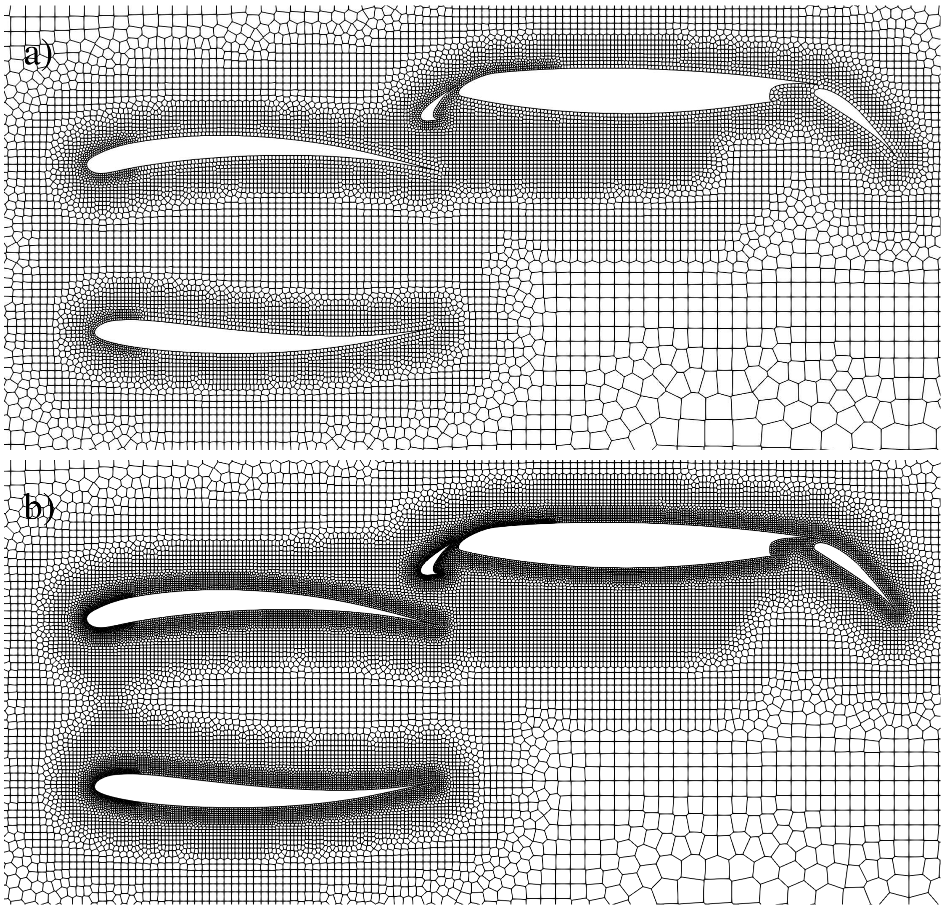

Fig. 4 A 2D mesh at y = 10.5 m around the NASA High-Lift Common Research Model with its smallest cell spacing defined as (a) 32 mm and (b) 16 mm is used to showcase the action of a refinement step on the definition of subsequent telescoping regions.

(see Fig. 5). This behavior is introduced by seeding all regions according to a single origin. In practice, the bounding box of each seeding region is covered by the smallest box which has all its corners on integer multiples of the region's seed spacing with respect to the previously defined origin. This covering is defined here as lattice box. Within this lattice box, test points are created at the primary lattice location and/or at staggered positions depending on the desired final cell shape. For example, some regular cells types are shown in Fig. 2.

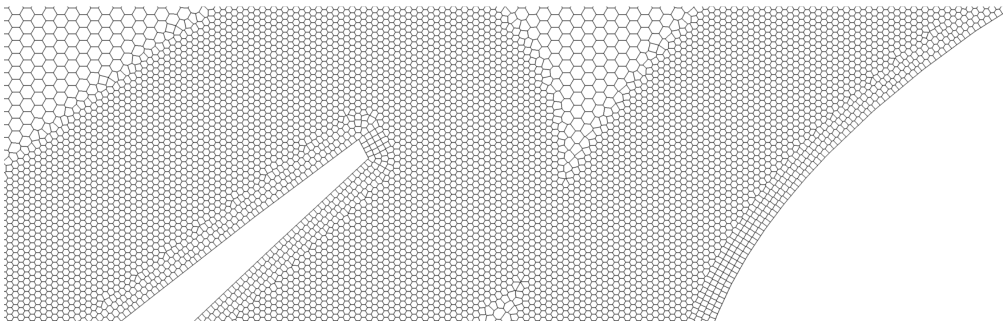

Fig. 5 Computational mesh around slat gap for a multi-element airfoil.

Before being included in the final seed list, a test point is checked to determine whether it lies within the computational domain. This means it must be inside the planes defining Ω and within the region toward which the normals of the triangles that define the geometry of interest point. Additionally, it must be verified whether the point lies within a region of higher priority. If a test point is found to be outside the computational domain or inside a previously seeded region, it is discarded.

This strategy can generate a large number of redundant test points. One case where this behavior is especially apparent is when telescoping regions are concerned. Each subsequent region encompasses another with a higher priority than themselves and therefore, all the test points generated that overlaps the finer-spaced regions are discarded. In this, and many other cases, the testing of each individual test point would be computationally expensive. The scheme devised to overcome this difficulty was to use a Cartesian tiling of the lattice box where the dimensions of each tile encompasses 8 points in each direction. Similar tests to the ones used to determine the validity of an individual seed are performed at the box level with three possible results. First, if a box is found completely outside the current seeding region or completely inside a region of higher priority all points are discarded. Second, in the opposite scenario where the box is completely inside the current seeding region and completely outside all regions of higher priority, all points are included. Finally, if a box intersects any previously seeded region or computational domain boundary, the test is inconclusive and the points within this box are tested individually.

B. Point Cloud Labeling and Near Wall Treatment

After defining the regions with varying refinement levels and constructing an initial seeding, seeds positioned in the vicinity of the interface between these regions, as well as those near the geometry, are identified. The positions of these points will be adjusted to enhance the quality of the final mesh for WMLES. The primary objectives are to achieve smooth transitions between regions with different cell volumes and ensure wall-normal alignment of the near-wall cells. A mesh that meets these criteria will maintain a consistent exchange location between the wall model and the LES solution.

The point cloud labeling step starts by looking for seeds which have at least one neighbor that belongs to a refinement region with a different spacing compared to the original seed's region. The points that satisfy this criteria are marked as the smoothing front. The neighbor information is acquired by looking through the connectivity of the Delaunay graph defined by the point cloud, i.e. the triangulation of points such that no point is inside the circumcircle of any triangle in the triangulation. At the same time, the smallest distance from a seed point to the geometry is inferred and seeds that lie within one lattice spacing are marked as the wall front.

The number of points within the wall- and smoothing-regions are user defined quantities. Each additional layer is found by iterating through the points in the current front and finding neighboring seeds that do not belong to either the current or a previously iterated front. Note that the growing of the smoothing layers occurs in both the directions facing the coarse and the fine refinement regions and the growing of the wall layers only occurs in the direction away from the surface. Finally, any point marked as near-wall is also included in the smoothing front set.

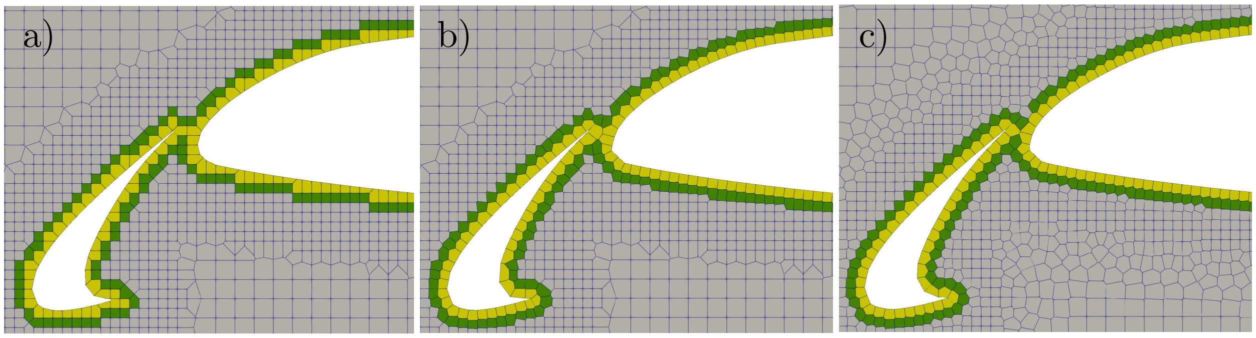

Fig. 6 2D mesh around the front of a multi-element airfoil showing the identification of the near-wall points at three different stages: a) before wall-layer projection step; b) after projection but before smoothing; c) final arrangement after projection and smoothing. The first-wall-layer cells are shown in yellow and the second layer cells are shown in green.

After the labeling step, the near-wall points undergo further treatment. First, the points identified as belonging to the first wall layer are checked to see if any of their neighbors belong to a different wall-layer. If a first-wall-layer point is found to be surrounded only by other first-wall-layer cells, it is marked for removal. The remaining near-wall points are then shifted along the wall normal vector to the desired wall-distance. The new distance to the surface (d_s) is set to the original wall layer number (η) minus half, multiplied by the seed spacing for the current refinement region (δ_i), i.e.

$$d_s = (\eta - 0.5)\,\delta_i. \tag{1}$$

This procedure is defined here as the wall layer projection step and ensures the creation of body-conforming cell layers with the desired wall-normal spacing, as shown in Fig. 6. Following the projection step for all wall layer points, any off-wall points (points that are not marked as near-wall) that fall within the total wall layer distance coverage are also discarded. After the first projection step, the seeds designated as smoothing points undergo Lloyd's smoothing procedure [30], detailed in the following subsection.

C. Point Cloud Smoothing

A Lloyd's smoothing procedure [30] can be used to improve the centroidal property of the limited set of cells marked for smoothing (near coarse/fine transitions and wall-layer zones) as well as providing a smoother gradation of cell volumes across the interface. A centroidal cell is defined as a cell for which the Voronoi generating seed and volumetric centroid of the resulting cell coincide. Given a list of Voronoi cell seeds $s_1, s_2, \cdots, s_n \in \mathbb{R}^3$, Lloyd's smoothing procedure simply computes the centroid of the each cell V_i, $i = 1, \cdots, n$, and updates the generating seed s_i position to coincide with the cell's centroid, c_i. With minor effort, one may show that this is both a contraction and a gradient descent algorithm along the functional:

$$T(s_1, s_2, \cdots, s_n) = \sum_{i=1}^{N} \|c_i - s_i\|^2 \operatorname{vol}(V_i)\,.$$

Note that the volume coefficient is positive. Since it is never the case that all cells have zero volume, the Lloyd step is a descent direction for T. Thus, in an iterative loop, cells approach a uniformly centroidal shape. An illustration of the Lloyd smoothing procedure is provided in Fig. 7. The initial random distribution of points yields Voronoi cells whose centroids are not coincident with the seed locations. After 20 iterations, the cells become almost perfectly centroidal with a much more even spacing.

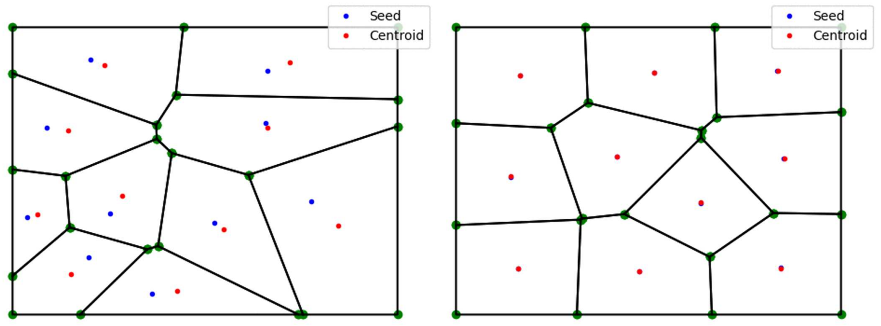

Fig. 7 Illustration of the Lloyd smoothing procedure. (left) initial, (right) after 20 iterations.

Since the LAVA Voronoi mesher intentionally places seeds in regular lattices, a considerable number of seeds may be left in place during the smoothing process depending on the user's specification. The Lloyd's smoothing [30] iterations are performed only on the set of the previously defined smoothing points (see Section II.B). Figure 8 shows the effect of 10 iterations of the smoothing procedure when used to create a body-conforming mesh that gradually coarses away from the wall.

In Section II.A, it was stated that test points that are found to be inside the geometry of interest are discarded. This means that, if no further action is taken, most of the near-wall points are going to generate highly non-centroidal cells and their location will be moved towards, or even inside, the geometry upon a Lloyd's algorithm iteration. To prevent this behavior, additional points are added to the Delaunay graph before each smoothing step and removed afterward. These extra points are defined as the projections of the points from the first near-wall layer onto the geometry's surface. Additionally, to maintain consistent wall-normal spacing, the near-wall cells undergo another wall layer projection step (Section II.B) with each smoothing iteration.

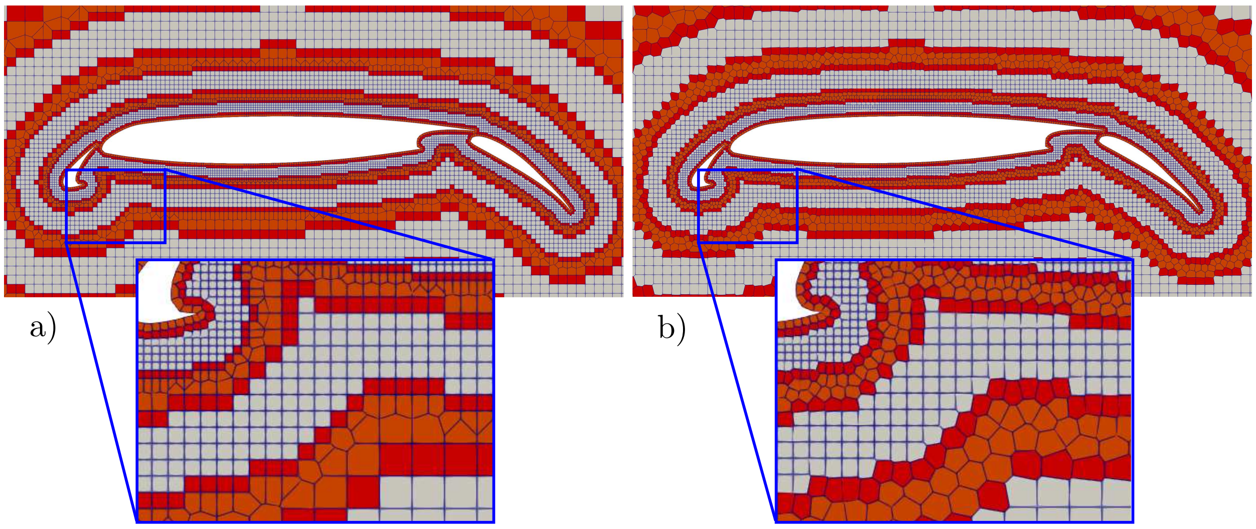

Fig. 8 Example mesh around a multi-element airfoil showcasing the identified smoothing points in red and the frozen points in gray before (a) and after (b) 10 iterations of Lloyd's smoothing algorithm.

D. Single Voronoi Cell Computation

In order to generate a Voronoi diagram given a list of seeds $s_1, s_2, \ldots, s_n \in \mathbb{R}^3$*, we must find a tiling of a domain Ω (assumed rectangular) into sets $V_1, V_2, \ldots, V_n$, that are disjoint except on their boundary, so that for each V_i, all $x \in V_i$ are closer to s_i than to any other seed point given a certain metric. The choice of an Euclidean distance results in convex polyhedral cells tiling Ω since each cell is formed by the intersection of half-spaces. Additionally, the faces of a Voronoi cell contain all the points that are equidistant from the two nearest neighboring seeds.

Since the Voronoi diagram is the dual graph of a Delaunay triangulation, the problem of generating the V_i cells can be recast as the problem of computing its dual graph and translating the results. Put another way, a Delaunay triangulation is the graph of face-neighbor connectivity for a Voronoi diagram. We denote a Delaunay triangulation by $\mathcal{D}(\boldsymbol{s})$ where $\boldsymbol{s}$ is a given seed list $\boldsymbol{s} = [s_1, s_2, \cdots, s_n] \in \mathbb{R}^{3\times n}$ stored in computer memory as a $3 \times n$ column major contiguously allocated matrix. The computation of a single cell via the Delaunay triangulation is denoted $V_i = \boldsymbol{Cell}(\mathcal{D}(\boldsymbol{s}), i)$, which returns a vertex list and face connectivity list sited at cell seed s_i.

This method provides improved numerical robustness compared to the alternative of calculating each cell individually through the intersection of half-spaces among nearby seeds around a specific generator point. If the previous method is used in the case where four seeds are close to sharing a common circumsphere, numerical errors may generate topological inconsistencies and the union of the individual cells may create connectivity errors in the final mesh [31]. Most of such possible failures are avoided by pre-computing the global connectivity graph associated with the Delaunay triangulation. Furthermore, computationally efficient methods for obtaining the Delaunay triangulation are available on open source libraries such as the Computational Geometry Algorithms Library (CGAL) [32] and Geogram [33]. In this work, we rely on the latter. Lastly, even though it seems necessary for this strategy to compute the connectivity amongst all generating seeds, it is possible to generate local graphs that respect the global connectivity [34]. This property allows for a distributed memory implementation of this strategy, which will be a topic of future work.

To generate a particular Voronoi cell, we read the seed-neighbor data structure obtained from the Delaunay graph and define cutting planes in terms of a given seed and its neighbors. In general, a Voronoi diagram may have boundary cells that extend to infinity but, since we are interested in the tessellation of a finite region corresponding to our computational domain, we further restrict the cells by the planes defining the domain Ω, resulting in the cell V_i.

Though the use of a Delaunay triangulation together with Simulation of Simplicity (SoS) algorithms [35] allows for the avoidance of most numerical errors in the generation of Voronoi cells V_i, failures might still occur, especially if regular seeding patterns are used. In the case where the underlying Delaunay triangulation has degeneracies, i.e., if there are any zero-volume tetrahedra in the Delaunay graph, then some of the systems of equations that define a vertex of a Voronoi cell as the intersection of cutting planes may lack a unique solution. A rank revealing QR factorization is

*Though one may write an optimized Voronoi tessellation algorithm for $\mathbb{R}^2$, the current implementation recasts the problem in three dimensions by mirroring the seed cloud across a pair of planes.

used to identify these cases. When a rank degenerate case is found, the set of Delaunay neighbors causing the deficiency is marked. The cell computation is then restarted with the marked vertices removed. Since the Voronoi cells are always well-defined convex polyhedra, the process of removing the collapsed Delaunay triangles will eventually remove all of the degenerate vertices and result in a well defined convex cell in the form of a vertex and face list. Although the process of repeating cell computation may initially sound costly, but it is our practical experience that there are only very few cells which ever enter the class of failure states described above. Accounting for them is simply a matter of algorithmic correctness for the few cells which happen to form degenerate vertices, and so it does not result in a computational bottleneck.

E. Geometry Representation as a Surface Triangulation

Currently, the input of a body in the LAVA Voronoi mesher is made through a closed surface triangulation that defines its external geometry. Initially, regions of this surface triangulation may be manually grouped into different components by the user. These classifications allow not only for the application of different boundary conditions along the geometry but also different seeding specifications. One example of this methodology can be observed in Fig. 9, where the leading edge of an iced airfoil model is identified as a separate component with respect to the rest of the airfoil and a finer seeding region is chosen by the user to resolve the small turbulent structures that may be present in that region.

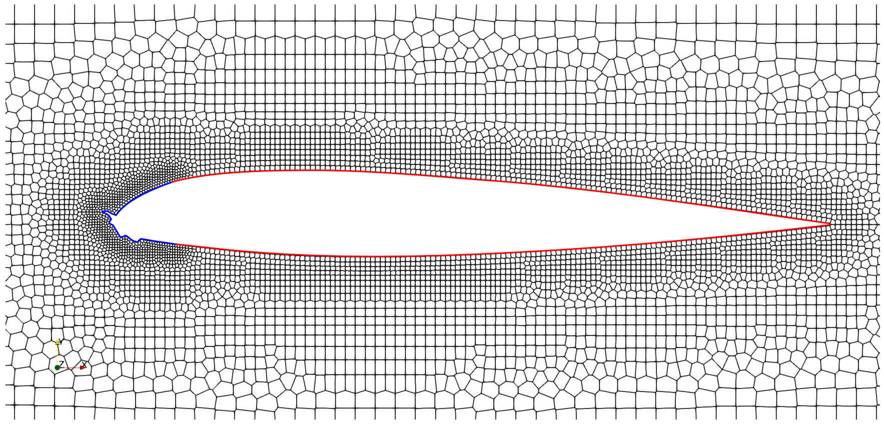

Fig. 9 The triangulation for the of EG1164 iced airfoil [36] is split between two components, the iced leading edge, in blue, and the the remainder in red. The separation in components allows for seed spacing variations in the mesh generation process.

However, in some setups, it is necessary to detect feature lines automatically. In that case, a flood-fill algorithm is used to further segregate the triangles that belong to a given component of the global surface triangulation by grouping those whose normal vectors are within a certain angle of each other. First, a starting triangle is chosen and “colored” with a numerical index we call the triangle’s color. This triangle initializes a list which is called the coloring front. Next, its neighbors are found by identifying the triangles with common edges and the angle between their normal vectors is inspected. For each neighbor, if the angle is smaller than a user specified threshold and the triangle has not yet been colored, then this triangle is colored by the same value as the parent and it is added to a coloring front. After all neighbors are tested, the parent is removed from the coloring front. This process is repeated until no triangles belong to the coloring front. At this stage, a check is performed to see if all the colored triangles share at least an edge and, if that is not the case, neighbors of triangles that only share a vertex are included in the coloring front and the process repeats. Once all the triangles of this preexisting component are already labeled, then the process is complete. On the other hand, if there are still unlabeled triangles, a new coloring front with a new color ID is started. This is repeated until all triangles of all preexisting components are grouped according to a certain angle criterion. Finally, a map is constructed between the newly formed groups and the user-defined components in the original surface triangulation. Even though this combined classification of the triangulation is used in the clipping algorithm (Section II.F), the existence of this map allows for the backwards translation upon the assembly of the global mesh and guarantees that the user will be able to set boundary conditions in the solver as intended based on the original triangulation components.

F. Clipping the Near-body Cells

At this point, the computational domain Ω is populated with a point cloud that was created to be concentrated in the regions of interest and smoothed in transition regions, and the corresponding Voronoi diagram restricted to Ω is generated. One crucial step is missing before WMLES can be performed to study the flow dynamics around complex geometries. The cells that intersect the surface triangulation that defines the geometry of interest need to be clipped and new surface faces need to be generated in a way that ensures cell closure. Each step pertaining this process is explained in the following sections. A simplified flow chart showcasing the logical steps is included in Fig. 10.

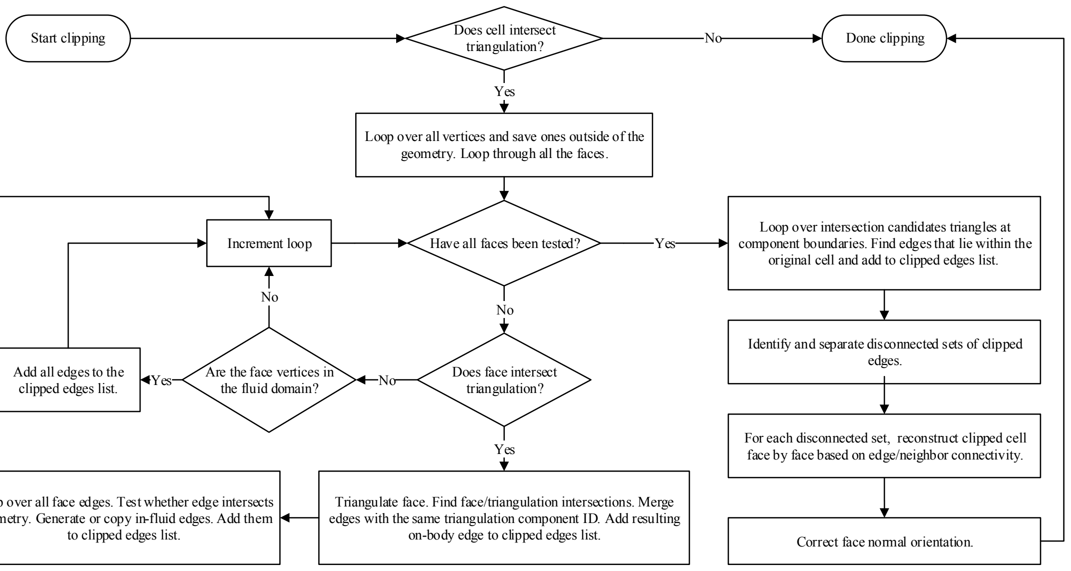
Start clipping
Does cell intersect triangulation?
No
Done clipping
Yes
Loop over all vertices and save ones outside of the geometry. Loop through all the faces.
Increment loop
Have all faces been tested?
Yes
Loop over intersection candidates triangles at component boundaries. Find edges that lie within the original cell and add to clipped edges list.
No
No
Identify and separate disconnected sets of clipped edges.
Add all edges to the clipped edges list.
Yes
Are the face vertices in the fluid domain?
No
Does face intersect triangulation?
For each disconnected set, reconstruct clipped cell face by face based on edge/neighbor connectivity.
Yes
Loop over all face edges. Test whether edge intersects geometry. Generate or copy in-fluid edges. Add them to clipped edges list.
Triangulate face. Find face/triangulation intersections. Merge edges with the same triangulation component ID. Add resulting on-body edge to clipped edges list.
Correct face normal orientation.

Fig. 10 Cell clipping algorithm flow chart.

1. Detecting Candidates for Intersection

Given a cell V_i in the premarked wall layers (see Section II.B) and a surface triangulation (Δ), we start by checking if there are triangles in Δ that possibly intersect the cell. An Axis-Aligned Bounding Box (AABB) tree search is performed and the candidates for intersection are identified. This search results in a subset of triangles, Δ_c, that contains all triangles that intersect with a given cell, as well as others which only have overlapping bounding boxes but do not truly intersect (see Figs. 12 and 13). If no candidates exist, this cell will not be clipped and the clipper returns from execution.

When Δ_c is not empty, a test is performed at all the original cell's vertices to identify if they lie on the inside or outside of the surface triangulation. Vertices in the fluid domain (outside of the triangulation) are appended to a clipped cell list of vertices. Next, a loop is performed over all original cell faces. In this loop, we construct a list of edges that constitutes the cutting of the Voronoi cell by Δ_c.

Each clipped edge object is defined by the indices of its end points in the unique list of clipped cell vertices and by the two faces that it neighbors. The face identification is performed either through the neighboring seed ID, if this is an in-fluid face, or by the triangulation component ID, if this is an on-body face. The information regarding which faces share a certain edge will be used to reconstruct the faces of the final clipped cells and needs to be collected while building the clipped edge list.

For each iteration of the loop over all original cell faces, new edge objects are potentially generated when cutting the original cell by the geometry. Since edges occur at the intersection of two faces, in general, the collection of the edges that belong to the set of clipped cells will happen with repetition. The procedure for recording the identifying information on the faces that share a certain edge starts by checking whether an edge object with the same vertices already exists in the clipped edges list. In the case that this edge is appearing for the first time, it is appended to the clipped edges list and the current iteration's face identifying information is stored, leaving the second index not populated. If, on the other hand, it already exists in the list, a previous iteration over the original cell's faces already created a similar edge object. Now, it has its first index already populated with the ID that identifies the face that previously generated it. In this case, the ID of the second face that this edge neighbors is populated with the information identifying the current iteration's face.

In the event of the edge lying on the body, it will only appear once while iterating through the faces of the original cell. This does not constitute a problem because, when a body edge is created, the identifying information on the two neighboring faces is known. This scenario can happen in one of two ways: either the body edge in question lies in between a fluid face, identified by its neighbor seed ID, and a region of the triangulation, identified by a certain component ID, or it lies between two triangulation regions with different component IDs.

2. Gathering In-fluid and On-body Edges

For each face of the original cell, a second AABB query limited to the cell's intersection candidates is performed. If no possible intersections are found, a vertex of this face is checked to see whether it lies within the surface or in the fluid domain. In the first case, all of the face edges are discarded, in the second, all are appended to the unique edges list. Now, if possible intersections are found, the face is triangulated and a triangle-triangle intersection check is conducted for each pair of face-triangle and surface-triangle. If no actual intersections are found after this step, a similar vertex check as the one previously mentioned is performed and only the in-fluid edges are kept.

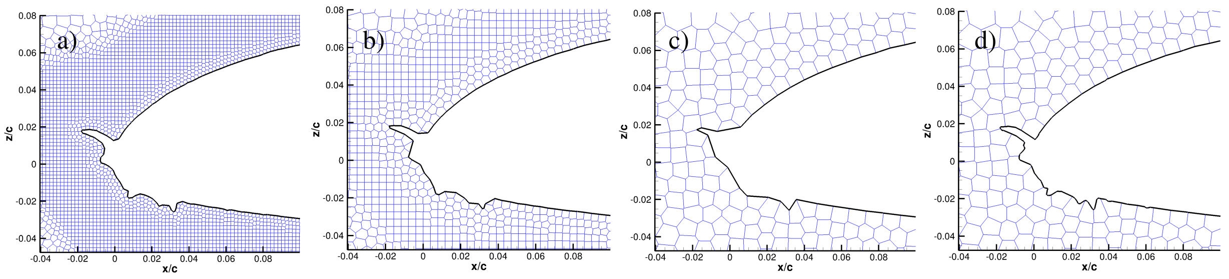

Fig. 11 Example of automatic defeaturing (subplots a, b and c) for different mesh refinement levels near the leading edge of the iced airfoil model EG1164 [36] as well as geometric detail retention on coarse meshes (d) if desired.

If the surface triangulation intersects a face, then each intersecting triangle pair creates an intersection edge. After all intersections are performed, the groups of intersection edges that belong to the same components (see Section II.E) are merged together, creating a single clipped edge per component. Once formed, the merged clipped edge is appended to the unique edges list. Such edge-type is labeled here as on-body edge (see Fig. 12). This step allows for the automatic defeaturing of the triangulation (Δ) since sections that are not resolvable by the incident cells can be skipped. If higher fidelity reproduction of the original geometry is desired, a separation of the triangulation amongst more components can be performed (see Section II.E). An example of both automatic defeaturing and high-fidelity feature representation, even in coarse meshes, is shown in Fig. 11. In the appending process, new vertices may have been formed and those are appended to the unique list of vertices if there is no previous entry at the same spatial location. In the case where a cell is not sufficiently small to resolve a sharp triangulation feature that pierces a cell and belongs to the same component, the merging algorithm results in a closed loop and an edge is not formed. Once this step is complete, both the in-fluid and body edges are appended to the unique edges list.

Next, a loop over the edges of the current face is performed and an edge-triangle intersection test is conducted. Note that in the case where split cells or split faces are formed, an edge can be intersected by the surface triangulation multiple times. If the edge in question is intersected only once, the vertex on the body is paired with the original cell's vertex which lies in the fluid domain. If multiple intersections occur, the intersection points are sorted based on the distance from one end of the original edge. By using the information of whether the end point in question lies within the triangulation or in the fluid domain, one can gather the intersection points in pairs and append the final edges to the unique clipped edges list. These edge types are referred here as intersection edges (see Fig. 12).

Upon conclusion of this set of instructions for all faces of the original cell, the unique edges list contains all the edges that are fully in the fluid region, all those that intersect the body, and all those that lie on the body as result of a face intersecting Δ. The only edges that are not yet included are the ones corresponding to edges between different components in the triangulation. At this point, an iteration is performed on the triangles that lie at the component

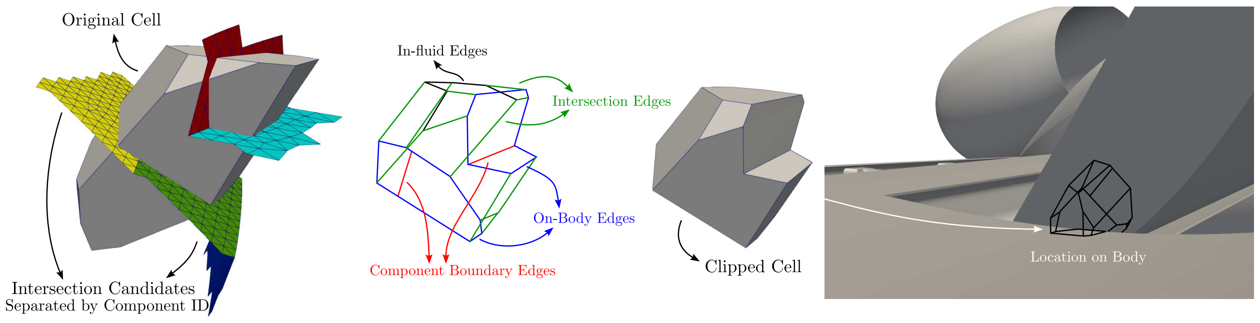

Fig. 12 Schematic representing the different types of clipped edges acquired during the clipping process.

boundaries to identify the boundary edges. Each of the end points of these component boundary edges are then tested to see if they lie completely inside or outside the original cell. If both ends of such an edge lie inside, this edge is appended to the unique clipped edge list, if both ends lie outside, then an intersection test is performed. If this component boundary edge intersects the original cell, then the portion that lies inside is appended to the unique clipped edge list. If no intersection occurs, this edge is discarded. Lastly, if one end is inside the original cell and the other is outside then it is known that the edge intersects a face of the original cell and, again, the portion that is within the original cell's boundaries are appended to the unique clipped edge list. An example of triangulation component boundary edges is shown in Fig. 12.

3. Thin Geometry Handling with Split-cells

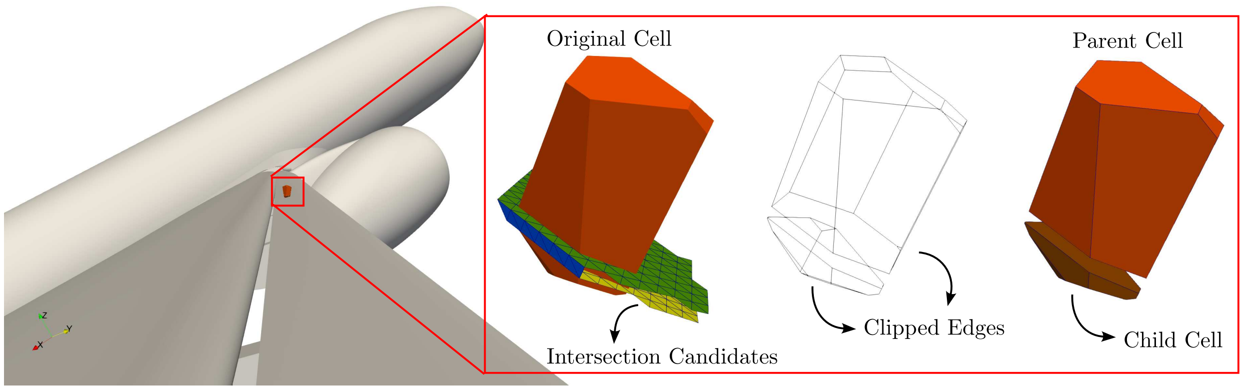

Fig. 13 Detailed representation of the creation of a split cell on the slat of the NASA High-Lift Common Research Model. After the original cell is intersected with the triangulation, two disjoint edge sets are formed. The connectivity of these edges are used to collect them into groups. These collections are used to reconstruct the faces of the parent and child cells.

At this point, all the necessary edges to form the final clipped cells are collected but, since split cells can be formed (see Fig. 13), there is a need to separate the possibly disjoint set of edges into distinct groups. For this, a flood fill algorithm is employed. The first step is to pick a random edge from the unique clipped edges list and to label it with a distinct value. The next step is to identify all the edges that are connected to the end points of this step's labeling front. This is a simple task since the edges are defined by a pair of indices that point into the unique list of vertices. Once all the neighbors are identified, all of these are labeled with the same value and a new labeling front is formed. This process is repeated until the labeling front of edges is empty. If the number of labeled edges correspond to the total number of unique edges, then the clipped cell is not a split cell and the algorithm is concluded. In the case that there are still edges that are unlabeled, then a new front is started and the process repeats until all the distinct groups of connected edges are identified. An example of a disjoint set of clipped edges is shown in Fig. 13.

4. Clipped Faces Formation

For each of the distinctly labeled groups, the edges that belong to the different faces are identified (see Section II.F.2) and faces are formed by cycling through the connectivity between edges, forming a list of vertices. The order of these vertices in the list determine the normal orientation of the formed face, and the last step is to determine correct vertex ordering of the newly formed faces. This process starts by iterating through the fluid faces. Since the original cell is convex, an outward normal orientation of faces that neighbor another fluid cell can always be enforced by testing whether a face normal points in the same direction as a vector that originates on the cell's centroid and points towards the face's centroid. This is not guaranteed for a body face and a special treatment is performed.

After the normal correction is performed on the fluid faces, an iteration on the body faces is conducted. First, note that at least one face neighboring a body face can be assumed to have a correct orientation. That is evident if a body face neighbors a fluid face. In the case that a body face only neighbors other body faces, it is possible to start the normal determination process by the body faces that neighbor a fluid face and move inward. Once there is a neighboring face with a correct normal orientation, one can simply observe the order that the vertices of the shared edge have in the cycle definition of the neighbor face. To enforce an outward normal in the body face, the vertices need to appear in the opposite order when compared to the neighbor in question. Finally, the distinct cells that are formed are then appended to the global mesh by using the procedure described in Section II.G.

G. Global Mesh Assembly

Upon obtaining a cell, we then must integrate this cell into a global mesh for the purposes of simulation on unstructured grids. To this end, we must uniquely identify all faces and vertices which appear with duplication in the list of all cells $V_1, V_2, ..., V_n$. Naturally, each face of V_i is associated with two seeds, (s_i, s_j). Thus we see that for each face of V_i, there is a bi-index (i, j). Upon first inspection, one may think that it is sufficient to use such a bi-index and its symmetry condition to uniquely identify a stored face.

Although this holds true for an unclipped Voronoi diagram, the presence of clipped cells (Section II.F) allows for faces to be split by geometric features. Typically, a split face results from an original cell being divided into a parent cell and multiple child cells. In this case, the split cells are assigned new unique IDs, preventing further issues. However, a split face can also occur without generating split cells. This situation can lead to multiple distinct faces sharing the same bi-index (i, j). If only one of these faces is appended to the global mesh, it can create holes.

To address this, we must account for the possibility of multiple distinct faces with the same (symmetrical) bi-index. Therefore, faces belonging to the seed with the smallest ordering index ($i < j$) are stored. This approach ensures that all unique faces are included in the global mesh without repetition.

When examining the vertices of the Voronoi cells, one may notice that they lie on the intersection of three faces. Additionally, the vertex is equidistant from the three neighboring seeds that identify these faces as well as from the seed that is responsible for generating the cell in question. In other words, they are in correspondence with the tetrahedra present in the Delaunay triangulation, as referenced in Section II.D. The indices of these four seeds can be used to uniquely identify vertices in the global mesh in most cases. The exceptions are related to highly structured locations, such as the corners of a Cartesian grid, where a vertex of a Voronoi cell can be equidistant to more than four generators. In those cases these vertices may be associated with different quad-indices and some duplicity may occur. To handle this, on top of the quad-index labeling, we employ a mesh repair strategy that looks for the nearest neighbors of the vertices in the global mesh within a certain tolerance. If multiple vertices are found in this small neighborhood, the average of these vertices is computed and all the original points are remapped to this representative location. After this step, the face vertices' connectivity cycles need to be rearranged in regards to this new map. During this procedure, very small facets may become edges or be completely removed from the mesh.

Finally, split cells may result from the clipping algorithm (Section II.F). In that case, both the parent cell and the children cells are initially identified with the same generator seed index. To create a global mesh usable for unstructured simulations, it is necessary not only to uniquely identify the resulting children but also to correct the connectivity between neighboring cells. With that in mind, split cells are stored and are not initially appended to the global mesh and split children are given a unique ID. After all the unclipped and unsplit clipped cells are computed and stored, the split cells are iterated upon. All the neighbors of each child cell are investigated and, if the corresponding face has been already appended to the global mesh or if the face belongs to another cell that has been split, then its connectivity information is corrected with the cell identifier of a split child. After this step, all the split cells are stored in the final global mesh.

III. Applications of the LAVA Voronoi Mesher

This section highlights some projects where the LAVA Voronoi mesher is leveraged. Its capability of automated meshing with only simple inputs from the user reduces the meshing turn-around time and allows for a streamlined construction of a family of progressively refined meshes that can be used in a mesh sensitivity study. Additionally, the comparison of the results obtained with the unstructured Voronoi approach with other simulation frameworks within LAVA, such as structured-curvilinear overset or Cartesian-octree immersed-boundary, can increase the confidence in the final results acquired from numerical simulations.

A. Jet Acoustics

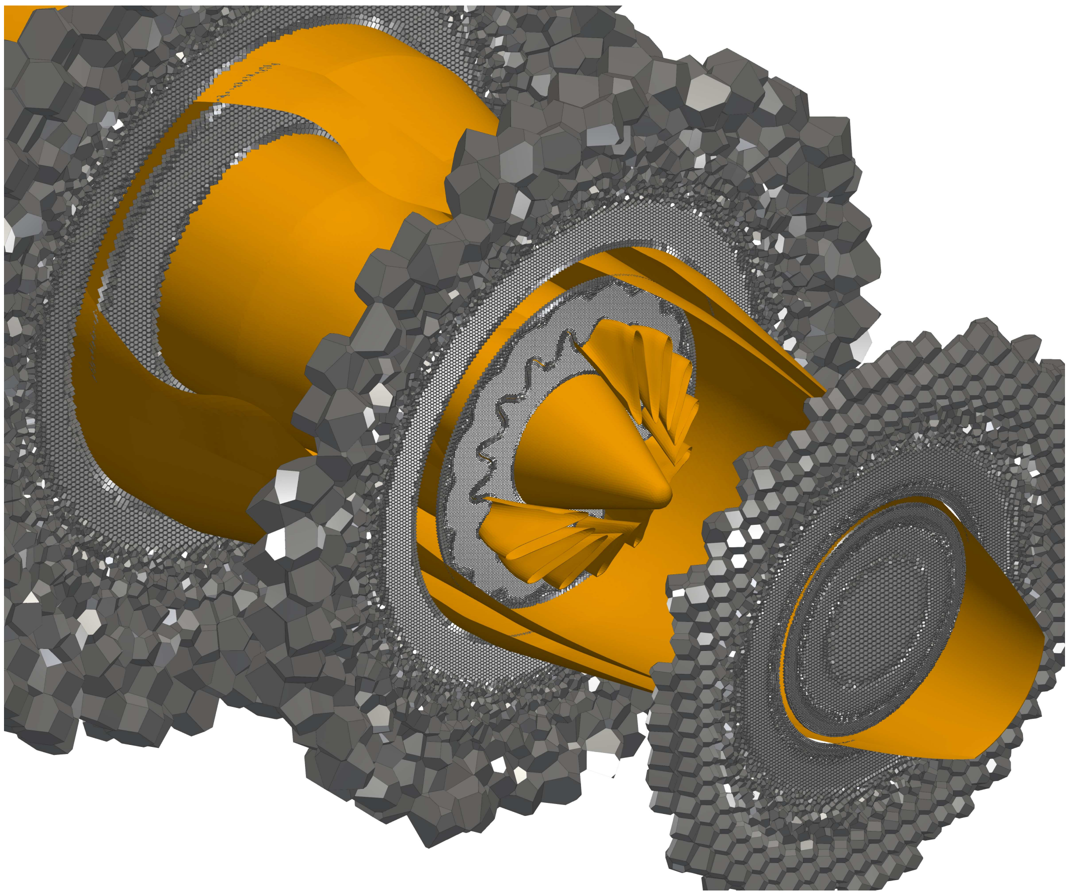

Fig. 14 Voronoi mesh around a potential nozzle geometry for a supersonic commercial airliner comprised of an internal plug and a lobed mixer.

In NASA's ongoing effort to develop a quiet supersonic jet, two main challenges must be addressed before such an aircraft can be certified. The first challenge involves reducing or eliminating the sonic boom ground signature during supersonic cruise. The second challenge is meeting noise constraints during takeoff and landing when the aircraft is traveling at subsonic speeds. For the second challenge, NASA's Commercial Supersonic Technology (CST) project has issued a technical task: to design a quiet propulsion system for a low-boom supersonic aircraft that not only meets the Federal Aviation Administration's airport noise regulations but also does so with a comfortable margin.

Previous work by Stich *et al.* [12] has shown that WMLES can be used to quickly generate the necessary database to characterize the noise levels of simple, axisymmetric jet flows under various conditions. However, it remains to be seen whether WMLES can accurately predict the complex aeroacoustics in more practically relevant geometries for commercial supersonic airliners, such as internally mixed exhaust systems. Understanding of the noise generation dynamics of such systems is still limited. To address this issue, a series of exhaust nozzles was designed [37]. Figure 14 shows a Voronoi mesh around one of these nozzles, featuring an internal plug and a lobed mixer.

The primary obstacle to extending Stich *et al.*'s work to more complex nozzle geometries is the creation of a series of high-quality meshes needed to populate the simulation database. The structured overset curvilinear grid framework used currently requires a labor-intensive and highly manual mesh generation process that could take a skilled engineer several days or even weeks to complete. The LAVA Voronoi mesher technology currently under development is expected to overcome this bottleneck and enable the use of WMLES to inform the design decisions for future quiet supersonic aircraft.

B. High-Lift Aerodynamics

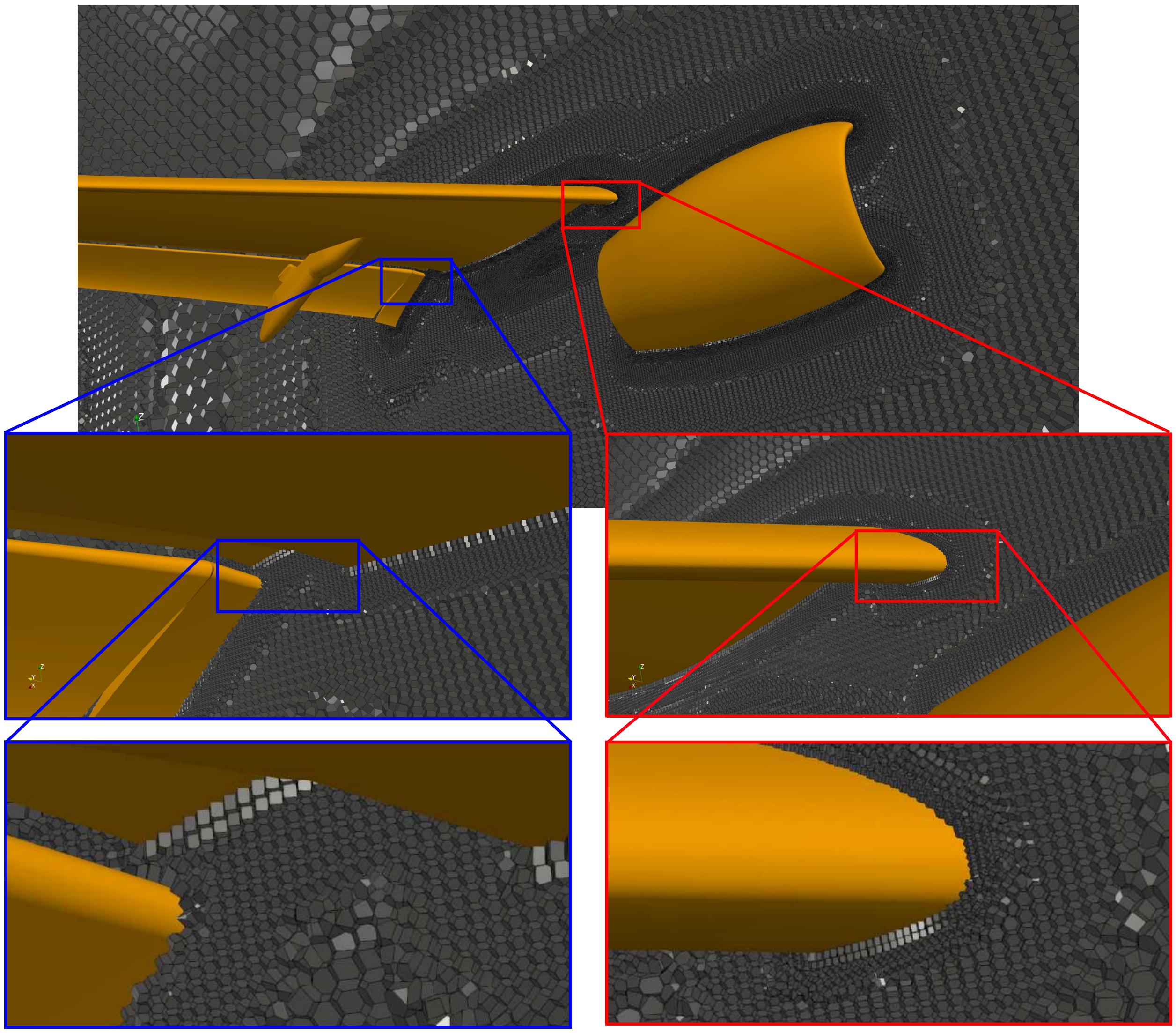

Fig. 15 Voronoi mesh around the NASA High-Lift Common Research Model. In this coarse example case, the smallest distance between cell generating points was set to 16 mm.

Numerical simulations have been widely utilized in the design process of aircraft configurations under cruise conditions. In these scenarios, the flow remains largely attached over the wings, allowing RANS models to produce accurate results. However, these models are less effective at predicting flow dynamics at the edges of the flight envelope. An example of this is when the aircraft's high-lift devices are fully deployed and it may be operating at low speeds and/or high angles of attack. The advancement of computational capabilities for analyzing high-lift configurations would significantly expedite the design, development, and certification of both current and new aircraft across the entire flight envelope. This challenging developmental task has the potential to enhance efficiency while reducing risks and costs [38, 39].

This scenario contributed to the organization of the High Lift Prediction Workshop (HLPW) series [13, 40–42]. The 4th HLPW focused on the NASA High-Lift Common Research Model (CRM-HL) configuration [43]. During this workshop, discrepancies were noted in flow separation and force and moment predictions of RANS-based approaches at high angles of attack near stall when compared to experimental data. In contrast, advancements in scale-resolving methods, such as WMLES, were found to accurately replicate the flow dynamics near stall, showing improved agreement with experimental results [13].

The LAVA group contributed to the HLPW-4 by submitting WMLES results utilizing its structured curvilinear overset framework. This method involves a labor-intensive meshing process with significant manual input from engineers. For a complex geometry like the CRM-HL, multiple engineers had to work simultaneously dedicating part of their time to meshing for several weeks to achieve a satisfactory grid. It was estimated that a single meshing expert would take between one and two months to complete this task considering a full-time dedication to meshing [44]. Conversely, a single engineer using the LAVA Voronoi mesher can create a series of high-quality meshes with varying refinement levels in just a day or two. One example of a Voronoi mesh around the CRM-HL geometry is shown in Fig. 15. The high quality cells achievable with the Voronoi mesher can provide accurate WMLES predictions for complex geometries while significantly reducing the time required to generate them. The Voronoi meshing approach is presently being employed for CRM-HL aerodynamic predictions as part of HLPW-5 and will be documented in a forthcoming publication.

C. Iced Wing

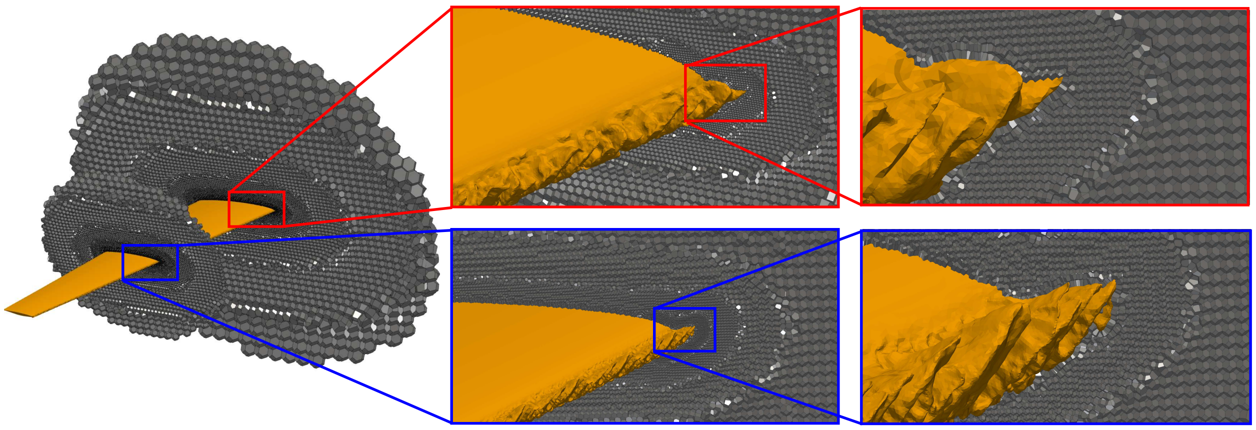

Fig. 16 Voronoi mesh around a scaled-down swept wing model with glaze ice accretion on its leading edge. In this coarse example, the smallest distance between cell generating points was set to 0.6 mm.

Ice accretion on an aircraft poses a significant safety risk. If an aircraft's ice protection system fails, its aerodynamic performance decreases drastically with a reduced lift, increased drag and a lower stall angle. Thus, it is crucial to develop and validate numerical prediction tools that can help improve the current understanding of the fundamental physics involved in iced flight and assist during the next-generation aircraft design process [45].

Simulating the aerodynamics of iced aircraft is especially challenging because ice accumulation tends to create large areas of unsteady separated flow, even at low angles of attack. The presence of such areas pose significant challenges for simulations using the latest RANS-based computational fluid dynamics tools [13, 42]. The LAVA group is currently evaluating the feasibility of performing WMLES on iced swept wing configurations[46] using two different simulation frameworks: a curvilinear overset approach coupled with an immersed boundary model and a Voronoi-based body-fitted grid (see Fig. 16). The immersed boundary technique is used to overcome the difficulties that would arise from trying to create a structured grid around the high-fidelity accreted ice shape. The geometric representations and aerodynamic measurements of the wings (and airfoils) with accreted ice are a result of an experimental campaign at NASA Glenn Research Center's Icing Research Tunnel (IRT) [47, 48]. It is expected that the comparison of the numerical and experimental results based on different frameworks will help the advancement of the simulation capabilities of aircraft under such environmental conditions.

IV. Summary and Future Work

The strategies and procedures behind the automated Voronoi mesher, currently being developed by the LAVA group at the NASA Ames Research Center, were presented in detail. The process of creating different refinement regions and seeding the Voronoi graph generators was explained. This includes methods for automated wall distance-based mesh coarsening, seamless integration of overlapping regions, identification of near-wall and smoothing points, and regularization of the near-body mesh. Additionally, a smoothing algorithm based on Lloyd's iteration method was introduced, allowing for smoother transitions between regions with different refinement levels and ultimately reducing spurious numerical reflections at coarse-fine interfaces. The manuscript also covered the strategy for computing a single Voronoi cell, including clipping by complex geometries. Algorithms for automated defeaturing of the geometry and handling possible split cells were discussed. The approach to creating a global mesh without repeated vertices or faces was also reported. Finally, potential applications and benefits of the LAVA Voronoi mesher were highlighted. The Voronoi mesher can significantly reduce the meshing time for complex geometries and its parametric nature makes it simple to generate a family of meshes with varying refinement levels. For example, for the CRM-HL geometry discussed in the manuscript, a single engineer is able to create a family of meshes up to 600M cells in one or two days using the Voronoi approach. This is a significant speed up when compared to the full dedication of a meshing expert working for weeks to create a single mesh for the same problem when using a highly manual curvilinear overset gridding framework.

All the discussions and meshing examples presented originate from the implementation of a shared memory parallel computing algorithm. While the LAVA Voronoi mesher successfully reduced the manual meshing bottleneck, particularly for complex geometries, the next phase of the project involves implementing a distributed memory approach using a message passing interface. This transition has the potential to further reduce the time required to generate the final mesh and enable the creation of highly refined meshes in a more cost-effective manner.

V. Acknowledgements

This work is funded by the Transformational Tools and Technologies (T^3) project, the Advanced Air Transport Technology (AATT) project, and the Commercial Supersonic Technology (CST) project within NASA's Aeronautics Research Mission Directorate (ARMD). Computing resources were provided by the NASA High-End Computing Capability (HECC) Program's NASA Advanced Supercomputing (NAS) facility at Ames Research Center. We would also like to thank Dr. David Craig-Penner (NASA Ames) and Leonardo Machado (NASA Ames) for their review and helpful feedback.

References

[1] Goc, K. A., Lehmkuhl, O., Park, G. I., Bose, S. T., and Moin, P., "Large eddy simulation of aircraft at affordable cost: a milestone in computational fluid dynamics," *Flow*, Vol. 1, 2021.

[2] Ghate, A. S., Stich, G.-D., Kenway, G. K., Housman, J. A., and Kiris, C. C., "A Wall-Modeled LES Perspective for the High Lift Common Research Model Using LAVA," *AIAA Aviation 2022 Forum*, 2022.

[3] Konig, B., Duda, B. M., and Laskowski, G. M., "Lattice Boltzmann Simulations for the 4th AIAA High-Lift Prediction Workshop using PowerFLOW," *AIAA Aviation 2022 Forum*, 2022, p. 3433.

[4] Ahmad, N. N., Wang, L., Anderson, W. K., Balakumar, P., Iyer, P. S., and Nielsen, E. J., "FUN3D Simulations for the 4th AIAA High-Lift Prediction Workshop," , 2022.

[5] Browne, O. M., Housman, J. A., Kenway, G. K., Ghate, A. S., and Kiris, C. C., "A Hybrid RANS-LES Perspective for the High Lift Common Research Model Using LAVA," *AIAA AVIATION 2022 Forum*, 2022.

[6] Kiris, C. C., Ghate, A. S., Duensing, J. C., Browne, O. M., Housman, J. A., Stich, G.-D., Kenway, G., Dos Santos Fernandes, L. M., and Machado, L. M., "High-Lift Common Research Model: RANS, HRLES, and WMLES perspectives for CLmax prediction using LAVA," *AIAA SciTech 2022 Forum,* AIAA Paper 2022-1554, 2022. https://doi.org/10.2514/6.2022-1554.

[7] Wong, M. L., Ghate, A. S., Stich, G.-D., Kenway, G. K., and Kiris, C. C., "Numerical study on the aerodynamics of an iced airfoil with scale-resolving simulations," *AIAA Scitech 2023 Forum*, 2023.

[8] Bornhoft, B., Jain, S., Goc, K., Bose, S., and Moin, P., "Wall-modeled LES of a NACA23012 airfoil under clean and iced conditions," Tech. rep., SAE Technical Paper, 2023.

[9] Wong, M. L., Kenway, G. K., Ghate, A. S., Stich, G.-D., and Kiris, C. C., "Predictions of LAGOON Nose Landing Gear Flow and Noise using Wall-Modeled Large-Eddy Simulations," *28th AIAA/CEAS Aeroacoustics Conference*, 2022.

[10] Bres, G. A., Bose, S. T., Emory, M., Ham, F. E., Schmidt, O. T., Rigas, G., and Colonius, T., "Large-eddy simulations of co-annular turbulent jet using a Voronoi-based mesh generation framework," *2018 AIAA/CEAS Aeroacoustics Conference*, 2018.

[11] Stich, G.-D., Housman, J. A., Ghate, A. S., and Kiris, C. C., "Jet Noise Prediction with Large-Eddy Simulation for Chevron Nozzle Flows," *AIAA Scitech 2021 Forum*, 2021.

[12] Stich, G.-D., Ghate, A. S., Housman, J. A., and Kiris, C. C., "Wall-Modeled Large-Eddy Simulation of Jet Noise in Flight Conditions," *28th AIAA/CEAS Aeroacoustics 2022 Conference*, 2022.

[13] Rumsey, C. L., Slotnick, J. P., and Woeber, C. D., "Fourth high-lift prediction/third geometry and mesh generation workshops: Overview and summary," *Journal of Aircraft*, Vol. 60, No. 4, 2023, pp. 1160–1177.

[14] Slotnick, J. P., Khodadoust, A., Alonso, J., Darmofal, D., Gropp, W., Lurie, E., and Mavriplis, D. J., "CFD vision 2030 study: a path to revolutionary computational aerosciences," Tech. rep., 2014.

[15] Beirão da Veiga, L., Brezzi, F., Cangiani, A., Manzini, G., Marini, L. D., and Russo, A., "Basic principles of virtual element methods," *Mathematical Models and Methods in Applied Sciences*, Vol. 23, No. 01, 2013, pp. 199–214.

[16] Beirão da Veiga, L., Brezzi, F., Marini, L. D., and Russo, A., "The hitchhiker's guide to the virtual element method," *Mathematical models and methods in applied sciences*, Vol. 24, No. 08, 2014, pp. 1541–1573.

[17] Da Veiga, L. B., Dassi, F., and Russo, A., "High-order virtual element method on polyhedral meshes," *Computers & Mathematics with Applications*, Vol. 74, No. 5, 2017, pp. 1110–1122.

[18] Gaburro, E., Boscheri, W., Chiocchetti, S., Klingenberg, C., Springel, V., and Dumbser, M., "High order direct Arbitrary-Lagrangian-Eulerian schemes on moving Voronoi meshes with topology changes," *Journal of Computational Physics*, Vol. 407, 2020.

[19] Ghate, A., and Lele, S. K., "Finite difference methods for turbulence simulations," *Numerical Methods in Turbulence Simulation*, 2022.

[20] Springel, V., "E pur si muove: Galilean-invariant cosmological hydrodynamical simulations on a moving mesh," *Monthly Notices of the Royal Astronomical Society*, Vol. 401, No. 2, 2010, pp. 791–851.

[21] Springel, V., "Moving-mesh hydrodynamics with the AREPO code," *Proceedings of the International Astronomical Union*, Vol. 6, No. S270, 2010, pp. 203–206.

[22] Jain, N., Bravo, L., Kim, D., Murugan, M., Ghoshal, A., Ham, F., and Flatau, A., "Towards Large Eddy Simulation of Rotating Turbomachinery for Variable Speed Gas Turbine Engine Operation," *Turbo Expo: Power for Land, Sea, and Air*, Vol. 58561, American Society of Mechanical Engineers, 2019.

[23] Ambo, K., Nagaoka, H., Philips, D. A., Ivey, C., Brès, G. A., and Bose, S. T., "Aerodynamic force prediction of the laminar to turbulent flow transition around the front bumper of the vehicle using Dynamic-slip wall model LES," *AIAA Scitech 2020 Forum*, 2020.

[24] Lozano-Duran, A., Bose, S. T., and Moin, P., "Prediction of trailing edge separation on the NASA Juncture Flow using wall-modeled LES," *AIAA SciTech 2020 Forum,* AIAA Paper 2020-1776, 2020. https://doi.org/10.2514/6.2020-1776.

[25] Lehmkuhl, O., Lozano-Durán, A., and Rodriguez, I., "Active flow control for external aerodynamics: from micro air vehicles to a full aircraft in stall," *Journal of Physics: Conference Series*, Vol. 1522, IOP Publishing, 2020.

[26] Brès, G. A., Ivey, C. B., Philips, D. A., Bose, S., Teramura, M., Moriya, T., and Ambo, K., "Large-eddy simulations of multi-bladed VTOL rotors for air vehicle aeroacoustic predictions," *AIAA Aviation 2023 Forum*, 2023.

[27] Sozer, E., Ghate, A. S., Kenway, G. K., Barad, M. F., Sousa, V. C., and Kiris, C. C., "Evaluation of Voronoi Meshes for Large Eddy Simulations of High Lift Aerodynamics," *AIAA Scitech 2023 Forum*, 2023.

[28] Sozer, E., Brehm, C., and Kiris, C. C., "Gradient Calculation Methods on Arbitrary Polyhedral Unstructured Meshes for Cell-Centered CFD Solvers," *52nd Aerospace Sciences Meeting*, 2014. https://doi.org/10.2514/6.2014-1440, URL https://arc.aiaa.org/doi/abs/10.2514/6.2014-1440.

[29] Choi, D.-I., Brown, J. D., Imbiriba, B., Centrella, J., and MacNeice, P., "Interface conditions for wave propagation through mesh refinement boundaries," *Journal of Computational Physics*, Vol. 193, No. 2, 2004, pp. 398–425.

[30] Lloyd, S., "Least squares quantization in PCM," *IEEE Transactions on Information Theory*, Vol. 28, No. 2, 1982, pp. 129–137. https://doi.org/10.1109/TIT.1982.1056489.

[31] Boots, B., Okabe, A., and Sugihara, K., "Spatial tessellations," *Geographical information systems*, Vol. 1, 1999, pp. 503–526.

[32] Fabri, A., and Pion, S., "CGAL: The computational geometry algorithms library," *Proceedings of the 17th ACM SIGSPATIAL international conference on advances in geographic information systems*, 2009, pp. 538–539.

[33] Lévy, B., and Filbois, A., "Geogram: a library for geometric algorithms," 2015.

[34] Peterka, T., Morozov, D., and Phillips, C., "High-performance computation of distributed-memory parallel 3D Voronoi and Delaunay tessellation," *SC'14: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, IEEE, 2014, pp. 997–1007.

[35] Edelsbrunner, H., and Mücke, E. P., "Simulation of simplicity: a technique to cope with degenerate cases in geometric algorithms," *ACM Transactions on Graphics (tog)*, Vol. 9, No. 1, 1990, pp. 66–104.

[36] Broeren, A. P., Bragg, M. B., Addy Jr, H. E., Lee, S., Moens, F., and Guffond, D., "Effect of high-fidelity ice-accretion simulations on full-scale airfoil performance," *Journal of aircraft*, Vol. 47, No. 1, 2010, pp. 240–254.

[37] Bridges, J. E., "Diagnosing Noise Features of Internally Mixed, External Plug Exhaust Systems," *AIAA Aviation 2023 Forum*, 2023.

[38] Slotnick, J. P., and Mavriplis, D., "A grand challenge for the advancement of numerical prediction of high lift aerodynamics," *AIAA Scitech 2021 Forum*, 2021.

[39] Mauery, T., Alonso, J., Cary, A., Lee, V., Malecki, R., Mavriplis, D., Medic, G., Schaefer, J., and Slotnick, J., "A guide for aircraft certification by analysis," *NASA/CR-20210015404*, 2021.

[40] Rumsey, C. L., Slotnick, J., Long, M., Stuever, R., and Wayman, T., "Summary of the first AIAA CFD high-lift prediction workshop," *Journal of Aircraft*, Vol. 48, No. 6, 2011, pp. 2068–2079.

[41] Rumsey, C. L., and Slotnick, J. P., "Overview and summary of the second AIAA high-lift prediction workshop," *Journal of Aircraft*, Vol. 52, No. 4, 2015, pp. 1006–1025.

[42] Rumsey, C. L., Slotnick, J. P., and Sclafani, A. J., "Overview and summary of the third AIAA high lift prediction workshop," *Journal of Aircraft*, Vol. 56, No. 2, 2019, pp. 621–644.

[43] Lacy, D. S., and Clark, A. M., "Definition of Initial Landing and Takeoff Reference Configurations for the High Lift Common Research Model (CRM-HL)," *AIAA Aviation 2020 Forum,* AIAA Paper 2020-2771, 2020. https://doi.org/10.2514/6.2020-2771.

[44] Ghate, A., Stich, G.-D., Kenway, G., Housman, J., and Kiris, C., "A Wall-Modeled LES Perspective for the High Lift Common Research Model Using LAVA," 2022.

[45] Struk, P., Broeren, A., Potapczuk, M., Kreeger, R. E., Blankenship, K., Chen, R.-C., Flegel, A., Porter, C., Ratvasky, T., Van Zante, D., et al., "NASA Analysis of Alternatives Study for Icing Research," *NASA/TM-20220006956*, 2022.

[46] Craig Penner, D. A., Housman, J. A., Stich, G.-D., Sousa, V. C. B., Koch, J. R. L., and Duensing, J. C., "Wall-Modeled Large-Eddy Simulations of a Swept Wing with Leading-Edge Ice," *AIAA Aviation 2024 Forum*, American Institute of Aeronautics and Astronautics, Las Vegas, NV, 2024.

[47] Lee, S., Broeren, A. P., Woodard, B., Lum, C. W., and Smith, T. G., "Comparison of Iced Aerodynamic Measurements on Swept Wing from Two Wind Tunnels," *2018 Atmospheric and Space Environments Conference*, 2018.

[48] Broeren, A. P., Lee, S., Woodard, B. S., and Bragg, M. B., "Effect of geometric fidelity on the aerodynamics of a swept wing with glaze ice accretion," *AIAA Aviation 2020 Forum*, 2020.