Three-Dimensional Surface Evolution and Mesh Deformation for Aircraft Icing Applications

Xialoing Tong,* David Thompson,† Qiuhan Arnoldus,‡ Eric Collins,§ and Edward Luke¶
Mississippi State University, Mississippi State, Mississippi 39762

DOI: 10.2514/1.C033949

This paper presents a mesh generation strategy that facilitates the numerical simulation of ice accretion on realistic aircraft configurations by automating the deformation of surface and volume meshes in response to the evolving ice shape. The discrete surface evolution algorithm is based on a face-offsetting strategy that uses an eigenvalue decomposition to determine 1) the nodal offset direction and 2) a null space in which the quality of the surface mesh is improved via point redistribution. A fast algebraic technique is then used to propagate the computed surface deformations into the surrounding volume mesh. Due to inherent limitations in the icing model employed here, there is no intent to present a tool to predict three-dimensional ice accretions but, instead, to demonstrate a meshing strategy for surface evolution and mesh deformation that is appropriate for aircraft icing applications. In this context, sample results are presented for a complex glaze-ice accretion on a rectangular-planform wing with a constant GLC-305 airfoil section and rime-ice accretion on a swept, tapered wing (also with a GLC-305 cross section). This meshing strategy is demonstrated to be robust and largely automatic, except for severe deformations.

Nomenclature

A_i	=	swept area
A_l	=	left facet area
A_r	=	right facet area
a_1, a_3	=	area of two parallel faces
a_2	=	area of a midway plane between two parallel faces a_1 and a_3
C_i	=	contraction factor for a given node
$\boldsymbol{c}_i$	=	vector from node to the offset centroid of the ith face incident on the node
h	=	face-offset height
$h_{\max}$	=	face-offset height at which volume accretion is maximum
h_{step}	=	face-offset height based on current substep
l	=	nodal displacement
M_i	=	rotation matrix in volume mesh deformation for the ith node
$\boldsymbol{n}$	=	face normal
$\boldsymbol{n}_e$	=	edge surface normal
$\boldsymbol{n}_i$	=	surface nodal normal
$\boldsymbol{n}_s$	=	normal vector of the swept edge area
$\boldsymbol{p}$	=	nodal position
$\boldsymbol{p}'$	=	nodal position after null-space smoothing
r	=	volume accretion rate
$\boldsymbol{r}$	=	coordinate vector for the original mesh in volume mesh deformation
$\boldsymbol{s}_i$	=	displacement field on the deforming surface for node i
s_t	=	safety factor to limit the magnitude of the tangential displacement
$s_{\Delta t}$	=	safety factor to adjust the maximum allowable time step
$\boldsymbol{t}$	=	tangential displacement of a node
V	=	accreted volume between offset faces
V_f	=	integrated accretion volume for facets
V_{flux}	=	volume flux exchanged between facets
$V_{\max}$	=	maximum accreted volume
w_i	=	weight function for the ith face at a given node
α_{Jiao}	=	stable time-step fraction using Jiao's method
α_h	=	height smoothing threshold
$\alpha_{\Delta t}$	=	stable time-step fraction for current substep
β	=	height smoothing factor
Δt	=	time between each time step
$\Delta t_{\max}$	=	maximum allowable time step
Δt_s	=	substep time
$\Delta t^i_{V_{\max}}$	=	maximum allowable time step for the ith face
ϕ_i	=	inner angle of ith face at a given node

I. Introduction

IN-FLIGHT icing remains a safety issue for both the commercial and general aviation communities. Atmospheric conditions between 9000 and 20,000 ft can produce supercooled water droplets that, although cooled below the freezing temperature by ambient conditions, exist in a liquid state [1]. The impact of these droplets on aerodynamic surfaces or engine inlets can provide the mechanism needed to initiate solidification. The resulting ice accretion may cause a loss of control of the vehicle or engine failure and, in severe instances, loss of the vehicle. Depending on various parameters such as air temperature, droplet size, liquid water content, aircraft flight speed, and the duration of the icing event, several different icing scenarios may occur [2], each of which produces ice shapes with different geometrical characteristics. It should be noted that these delineations are not distinct and mixed-type accretions are often observed. Small-scale roughness ice forms in the initial stages of the ice accretion process and typically is highly three-dimensional. If the temperature is relatively cold (i.e., a rime-icing condition), the droplets freeze on impact with the surface. This is primarily a deposition problem, and the resulting ice accretion generally follows the shape of the underlying airfoil surface; although, at long icing times, horns may appear. Bragg et al. [2] termed this shape streamwise ice. If the temperature is somewhat warmer (i.e., a glaze-icing condition), the droplet may run back on the surface before freezing, which produces the classic horn ice accretion. Spanwise-ridge ice accretion occurs when runback traverses the surface to a

Presented as Paper 2014–2201 at the AIAA 6th Atmospheric and Space Environments Conference, Atlanta, GA, 16–20 June 2014; received 22 March 2016; revision received 26 July 2016; accepted for publication 27 July 2016; published online 20 October 2016.

*Assistant Research Professor, Center for Advanced Vehicular Systems, P.O. Box 5405. Member AIAA.

†Professor, Department of Aerospace Engineering, P.O. Box A. Associate Fellow AIAA.

‡Research Associate III, Center for Advanced Vehicular Systems, P.O. Box 5405. Member AIAA.

§Assistant Research Professor, Center for Advanced Vehicular Systems, P.O. Box 5405. Member AIAA.

¶Professor, Department of Computer Science and Engineering, P.O. Box 9637. Member AIAA.

region downstream of the ice protection system and freezes. These irregular ice shapes are typically observed in supercooled large droplet icing conditions, such as those encountered in freezing rain or freezing drizzle. The effects of three-dimensionality further complicate the process as exceedingly complex scallop ice shapes are observed on swept wings [3].

There are several technical challenges associated with mesh generation for simulating ice accretion on a three-dimensional configuration. First, ice accretion is an evolutionary process; therefore, the discrete representation of the geometry must evolve in response to the growth of the ice. In addition to a redefinition of the discrete surface representation, each accretion step requires a new volume mesh for the flow simulation. However, a full mesh regeneration may be time consuming for complex configurations. A more attractive alternative is to deform the mesh in response to the surface evolution. The second challenge is that accreted ice shapes can become quite complex. Although the current state of the art in ice accretion prediction does not produce shapes with exceedingly complex geometries, such as scallops [3], the predicted ice shapes can nevertheless present significant challenges for meshing software. However, as the fidelity of ice accretion prediction increases, the complexity of the numerically generated ice shapes will increase.

The strategy described in the following sections is designed to address the following specific needs relevant to mesh generation for realistic configurations based on grid-based ice accretion simulations:

1) The first need is automation. Simulating the evolving ice shape necessarily requires generating a new mesh for each ice shape. For this approach to be practical in a production environment, it is necessary that the mesh generation process be as automated as possible. The underlying strategy is that the user should be removed from the loop to the extent that is possible. Therefore, it is critical that the discrete surface mesh retains its quality as the ice shape evolves.

2) The second need is efficiency. Although a new mesh must be generated for each ice shape, the process must be efficient. The simplest approach, completely regenerating the mesh after each accretion step, is potentially a time-consuming task for complex aircraft configurations and not appropriate for ice accretion simulations. Global regeneration of meshes should be avoided whenever possible.

3) The third need is robustness. Any mesh generation tool that is to be employed in an automated analysis environment must be robust. In the context of mesh generation for ice accretion simulations, robustness implies that a valid mesh of reasonable quality must be generated for ice shapes of varying complexity. The challenge here is to ensure that the surface mesh retains sufficient quality as the ice surface evolves.

This paper describes the algorithms employed in a meshing strategy that is appropriate for problems with evolving surfaces. The approach is demonstrated for two representative aircraft icing problems. These sample results are obtained by coupling LEWICE3D [4] with the flow solver Loci/CHEM [5] to generate an ice accretion rate map [6,7]. This rate map is used as input to the discrete surface evolution tool iceSurf, which is described in Sec. IV. A. The tool iceSurf is designed to produce a new surface mesh given the current surface mesh, a face-centroid accretion rate map, and the icing time. It should be noted that the surface propagation tool is solver neutral, i.e., other ice accretion/flow solver software could be used to provide the ice accretion rate map. The surface displacements are then propagated into the volume mesh using the mesh deformation tool gridMover [8]. Due to inherent limitations in the icing model currently employed by LEWICE3D, we do not suggest that the resulting coupled software package is a tool that can be used to predict three-dimensional ice accretions. Instead, our intent is to demonstrate a meshing strategy for surface evolution and mesh deformation that is appropriate for aircraft icing applications.

We first provide brief descriptions of existing ice accretion methods as well as general surface evolution algorithms. We next provide background on the surface evolution algorithm developed by Jiao [9,10], upon which our surface evolution algorithm is based. We then describe our surface evolution algorithm in some detail. We next describe the algebraic method employed to propagate the surface displacements into the volume mesh. These techniques are demonstrated by generating ice accretion results for 1) a complex, 22.5 min ice shape produced by glaze-icing conditions for a rectangular-planform wing with a GLC-305 cross section [11]; and 2) ice shapes produced by rime-icing conditions for a swept and tapered wing, also with a GLC-305 cross section [3]. Finally, the paper concludes with a description of potential future enhancements to the meshing process.

II. Literature Review

Aircraft icing codes perform multiphysics simulations that typically combine models for fluid flow, multiphase droplet trajectories, heat transfer, phase change (solidification), and surface evolution. Often, ice accretion is simulated using a multistep strategy in which the effects of the evolving ice shape on the fluid flow and multiphase trajectories are considered. In this paradigm, a time integration of ice accumulation rates is required to track the evolution of the iced surface. The flow simulation is then updated using this new surface definition, and the process is repeated. For grid-based flow solutions, mesh deformation or regeneration strategies are needed to provide the fluid model with an appropriate volume mesh. There are a number of ice accretion prediction codes [4,12–18] that follow this basic modeling architecture. One notable exception is the morphogenetic approach of Szilder and Lozowski [19,20]. Their Lagrangian approach is based on a discrete formulation of the Messinger model [21] that produces a stochastic freezing probability for a fluid parcel as it moved along the surface after impact. Instead of representing ice shapes in the form of an evolving surface, this model represents the ice volume as a collection of discrete frozen parcels that form the shape in aggregate. Because of its unique formulation, the morphogenetic model can predict ice shapes with surface roughness as well as variable ice density, which can exhibit significant deviation due to the shadowing effect and the formation of voids.

LEWICE [17] is a well-established ice accretion code. The code is based on the two-dimensional Messinger icing model [21], which employs a quasi-steady energy balance to compute a freezing fraction along a strip. LEWICE can perform multiple icing steps by updating the flowfield using its native panel method flow solver as the ice shape evolves. The capability to use the Navier–Stokes code WIND [22] to compute the flow solution, including automated grid generation, was added to LEWICE by Thompson and Soni [23]. In this case, the two-dimensional nature of the problem considerably simplified the grid generation process. Points on the airfoil surface were redistributed on a non-uniform rational basis spline (NURBS) representation of the new iced airfoil shape obtained from LEWICE using a curvature-weighted equidistribution scheme. A new grid was then generated in the two-dimensional computational domain, which required only a matter of seconds. LEWICE was extended to enable three-dimensional icing simulations by combining a collection of two-dimensional strips in the LEWICE3D [4] code. However, LEWICE3D currently only performs a single icing step and has not addressed the problem of automated multistep ice accretion prediction.

Three-dimensional multistep icing computations have been demonstrated by Pendenza et al. [18] using FENSAP-ICE3D. The ICE3D component was based on a 2.5-dimensional shallow-water icing model developed by Bourgault et al. [24] and produced an icing rate map on the surface. Their description focused on the volume mesh deformation technique but provided only a high-level discussion of the surface evolution aspect. The mesh movement strategy was based on an arbitrary Lagrangian–Eulerian formulation [25] that employed a frame-based formulation in the region near the deforming surface in order to maintain orthogonality and a standard elasticity-based mesh movement strategy in the outer region. Problems that occurred in the surface evolution step were corrected using a regularization procedure that included an ad hoc smoothing technique. In extreme cases, the volume mesh was regenerated rather than deformed from a previous step.

Generally, surface evolution problems can be solved with either interface-tracking or interface-capturing methods. Interface-tracking

methods directly track the evolution of the surface using boundary fitted meshes, whereas interface-capturing methods model the surface as an implicit function and model the interface evolution through advection of the implicit function. A detailed discussion of the numerical challenges associated with interface evolution problems was given by Sethian. [26]. Generally, the most robust algorithms are in the interface-capturing class and include volume-of-fluid [27], level set [28], and marker-and-cell [29] methods. An implicit formulation of the interface-capturing problem usually requires flow solvers that can support immersed boundaries [30] using methods such as the ghost fluid method [31]. Unfortunately, these immersed boundary methods are generally limited to problems at low Reynolds numbers due to difficulties in efficiently resolving thin boundary-layer effects at immersed boundaries. Icing models that track dendritic ice growth under low-Reynolds-number conditions have been implemented using interface-capturing methods [32], for example.

Given the challenges of simulating aircraft icing and the desire to couple to existing well-validated high-Reynolds-number flow solvers that use boundary fitted meshes, interface-tracking methods are more directly applicable to our problem. A number of researchers have investigated interface-tracking methods [33–35] for capturing complex interface dynamics, including the growth of solidification boundaries. Interface-tracking methods have generally been plagued with numerical difficulties associated with instabilities manifesting as growing oscillations or problems with self-intersection. This has resulted in the development of hybrid methods that combine both approaches [36,37]. Generally, these methods have complex implementations and can suffer from diffusion of singularities that degrade the accuracy of interface-capturing techniques. A robust interface-tracking method was developed by Jiao [10] that used a face-offsetting method to achieve a surface evolution approach that was accurate and stable. This method provides a foundation for applying the entropy-satisfying Huygens principle that is used in level set methods, which results in numerically stable surface evolution. In addition to numerical stability, an advantage of Jiao's algorithm is that it does not require the solution of equations on a background mesh, as would be required by an interface-capturing method.

Once a robust algorithm for describing the evolution of the iced surface has been identified, the flow solver requires a good-quality volume mesh that conforms to the updated surface mesh. One approach to achieving such a conformal mesh is mesh regeneration. Unfortunately, regeneration of viscous meshes on full aircraft configurations would be exceedingly expensive in a multistep framework. An alternative, and a more commonly employed approach, is to deform the previous volume mesh to conform to the updated surface. Physical analogy methods include elastic- [38–40], spring- [41,42], or diffusion-inspired [43] approaches; they generally require a fairly expensive, nonlinear solver and can lack robustness for large deformations. Alternative methods include interpolation methods such as radial basis function [44,45] and inverse distance weighted interpolation methods [8,46]. These methods are generally very robust, efficient, and are easily applied to arbitrary mesh topologies. In particular, the inverse distance weighted approach of Luke et al. [8] used a fast-multipole approximation algorithms to achieve robust and efficient large-scale mesh deformations.

III. Background: Surface Evolution

One of the challenges associated with evolving a faceted, discrete surface representation is that the normal at a node is not unique. This is caused by the discontinuous nature of the discrete representation of the surface. One possible solution is to define a displacement direction at each node, based on the normals in the adjacent faces, and displace the surface a prescribed distance in this direction at each node. However, there are numerous challenges associated with this approach, not the least of which is conservation of volume. Alternatively, the surface evolution could be modeled by generating a plane that is parallel to a given face by extruding a specified distance (the product of the accretion rate and the time step) from the face

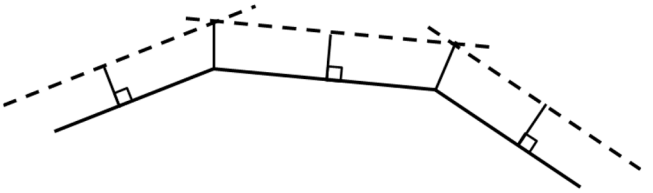


Fig. 1 Face offsetting producing unambiguous nodal positions in two dimensions.

centroid in the direction of the face normal, as shown for a two-dimensional surface in Fig. 1. As seen in the figure, there is no ambiguity in the location of the nodes in the new surface in two dimensions; however, this is not the case in three dimensions, in which any two of the nonparallel offset planes intersect in a line while any three nonparallel planes intersect at a point. In general, the intersection of four or more planes is not defined in three dimensions. Except in special cases, the number of faces that shares a given node in a typical triangular surface mesh is usually more than three; consequently, the position of the node is overspecified. This results in an ambiguity in how the nodal positions are defined in the new surface.

The approach we have chosen has shown much promise for evolving a surface mesh while conserving volume [9,10]. Since this is the basis of our surface evolution algorithm, we now describe Jiao's algorithm in some detail. Jiao employed a singular value decomposition to solve a least-square problem and then applied an eigenvalue/eigenvector analysis at each node to resolve its normal motion, which generated the surface geometry, and its tangential motion, which could be used to maintain surface mesh quality.

The first step is to propagate or "offset" each evolving face in the direction of its normal. Since the face velocity is given, a simple first-order Euler scheme is chosen to integrate along the face normal. This provides the offset distance for each face.

The second step is to reconstruct the vertices. After computing the new face positions, a new position for each node on the surface must be determined. For simplicity, assume that the node under consideration is located at the origin. Each plane passing through a point $\boldsymbol{p}$ with unit normal $\boldsymbol{n}$ can be expressed by a linear equation $\boldsymbol{n}^T\boldsymbol{x} = \delta$, where $\delta = \boldsymbol{n}^T\boldsymbol{p}$. If there are m faces passing through a node, an $m \times 3$ linear system will be formed:

$$\boldsymbol{N}\boldsymbol{x} = \boldsymbol{a} \tag{1}$$

Here, each row of the system corresponds to one of the m faces that are incident on the node and elements of $\boldsymbol{a}$ are the offset distances for each incident face. The linear system given by Eq. (1) can be underdetermined or overdetermined, depending on the value of m. To address this difficulty, a least-square solution is computed. A point is chosen that minimizes the weighted sum of squared distances to the face planes, which is a solution of the following 3×3 linear system:

$$A\boldsymbol{x} = \boldsymbol{b} \tag{2}$$

where $A = \boldsymbol{N}^T\boldsymbol{W}\boldsymbol{N}$, $\boldsymbol{b} = \boldsymbol{N}^T\boldsymbol{W}\boldsymbol{a}$, and $\boldsymbol{W}$ is an $m \times m$ diagonal matrix with W_{ii} equal to the weight associated with the ith face, which is based on the area of the face incident on node $\boldsymbol{p}$.

Since the matrix A in Eq. (2) is symmetric and positive semidefinite, it has an eigenvalue decomposition $A = \boldsymbol{V}\Lambda\boldsymbol{V}^T$, where Λ is the diagonal matrix consisting of the eigenvalues of A, which are real and nonnegative, and the corresponding eigenvectors are the columns of $\boldsymbol{V}$. Since $A = \boldsymbol{N}^T\boldsymbol{W}\boldsymbol{N}$, the following singular value decomposition can be derived:

$$\sqrt{\boldsymbol{W}}\boldsymbol{N} = U\sqrt{\Lambda}\boldsymbol{V}^T \tag{3}$$

where U is a $m \times 3$ matrix.

The vector space spanned by the eigenvectors corresponding to the larger eigenvalues of A is called the primary space, and the complementary space is the null space. An eigenvalue analysis is

performed to identify the primary space. All of the eigenvectors corresponding to eigenvalues smaller than a threshold will be filtered to avoid instability due to division by a very small number. The nodal displacement is restricted to the primary space.

The solution to Eq. (2) represents an advective motion in which the resulting surface is the intersection of the propagated face planes. For wave front motion, such as that produced by burning, erosion, and deposition, the displacement in the primary space satisfies an entropy condition. Unfortunately, the exact displacement for wave front motion can be difficult to compute. A simple solution is to assume the direction of the displacement will not change and adjust the displacement to satisfy the required offset.

After the displacement was computed, Jiao improved mesh quality by performing a null-space smoothing by computing a tangential motion $\boldsymbol{t}$ at each vertex $\boldsymbol{v}$ [9,10], where $\boldsymbol{t}$ was a weighted average of the neighborhood of $\boldsymbol{v}$ projected onto the null space. This smoothing scheme has been shown to preserve sharp features and to introduce only very small-volume errors. Jiao suggested repeating this step to incorporate a global smoothing into the algorithm that preserved the accreted volume.

IV. Approach

A. Surface Mesh Evolution

In this section, we document, in detail, the algorithm used to evolve the discrete representation of the wetted surface (clean and iced) that describes the geometry. In our strategy, the discrete geometry that describes the surface also serves as the surface mesh in the computational simulation. Our basic algorithm employs elements of Jiao's face-offsetting algorithm [9,10], except that, instead of integrating a normal velocity at the surface, we integrate volume accretion rates that are produced by the lofted icing model [6,7]. As a result, the effective surface normal velocity may vary throughout the ice accumulation integration step. The iceSurf tool uses the offset direction in the primary space defined by Jiao's method as the initial nodal displacement direction, and then it employs global and local smoothing algorithms to maintain mesh quality.

Inputs to iceSurf include a surface mesh file, an accretion rate file, and an accretion time. Given this information, iceSurf will propagate the surface mesh while maintaining volume conservation and mesh quality. If contracting faces exist, the propagation may need multiple substeps. Figure 2 illustrates the overall procedure used to propagate the surface for one icing step. The algorithm begins on the upper left corner with an integration of the icing volume rate and completes on the upper right corner with a final volume correction step.

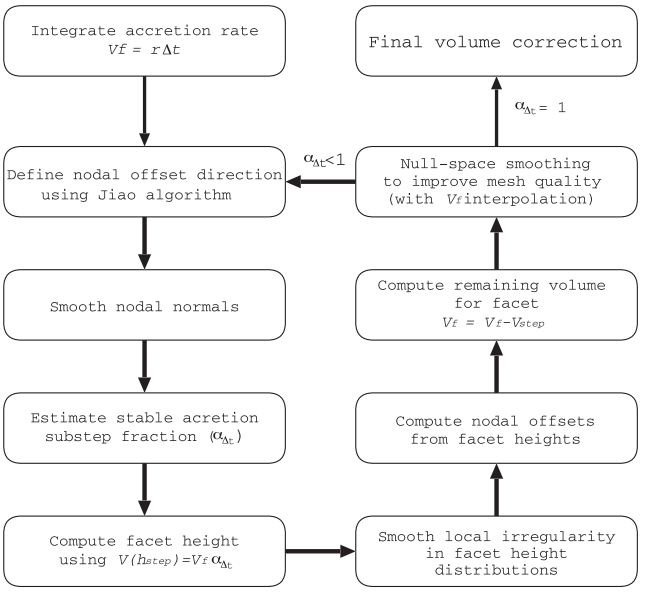


Fig. 2 Flowchart for iceSurf surface propagation substeps.

1. Generate Accretion Rate

In general, our tool is designed to couple with a three-dimensional icing model that calculates local icing volume accretion rates. Due to the inherent geometrical nature of the lofted ice shape reconstruction method [6,7], the most natural output datum for the current quasi-three-dimensional model is the face-offset height, which is the distance that the face center moves to intersect with the projected ice surface. We use this icing offset height for each face to compute an effective volume rate. Note that a true three-dimensional icing model would likely generate the icing rate directly. In the case of the lofted data, the icing rate is computed as the volume of the prismatoid obtained by extruding parallel to the face following the surface nodal normals (see Fig. 3). Then, the volume rate for the facet is simply this volume divided by the current icing time step. The surface nodal normals are computed using Jiao's algorithm [9,10]. In general, once we have the volume accretion rate, then the accreted volume is found by integrating the volume rate over the icing time step. For the present method, a simple explicit Euler integration is used such that the volume assigned to each facet at the beginning of the surface evolution is $V_f = r_f \Delta t$. Due to the highly nonlinear behavior of surface normals during the surface evolution, a single icing time step is usually divided into a number of substeps in order to ensure stability.

2. Define Nodal Offset Direction

The first step in the iterative surface evolution method is the generation of an initial nodal offset direction. In Jiao's original algorithm [9,10], the normal speed was directly used to compute the offset height. This step was followed by the solution to a least-squares problem to find the intersection of the advancing face planes. However, for the icing rate integration employed here, the estimated offset height is a strongly nonlinear function of facet volume targets as well as nodal normals. To resolve this difficulty, we employ Jiao's algorithm using a unit height on all faces. Furthermore, the weights in the least-squares problem are based on the included angle of each face rather than its area to reduce sensitivity to regions where triangle sizes are highly variable. We then select the primary space from the eigenanalysis as the normal direction used for face offsetting. Since these normal directions can be used to define the volume associated with a face-offset height, we can now perform the volume integration as a separate step.

3. Reduce Surface Noise by Local Normal Smoothing

In icing simulations, the local volume of accreted ice is sensitive to the surface normals, which also affect the direction of face propagation. Noise in the surface can be amplified during surface propagation. Left unchecked, this noise can cause the dihedral angle between two faces to become large, and therefore limit the maximum allowable time step. To reduce the surface noise, we apply a local smoothing by adjusting the nodal offset direction in problematic regions so that it more closely aligns with the directions of its neighbors. This method can improve the surface smoothness in some situations.

The main goal of normal smoothing is to push points out of concave regions where normals may locally converge. The normal smoothing is achieved through a series of weighted averages that are designed to give weight to the normals generated by problematic features. The first step of this normal smoothing is to average nodal

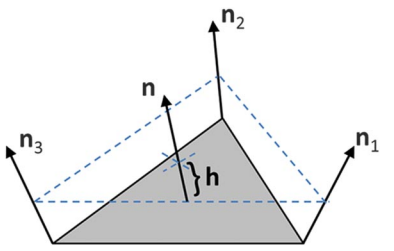


Fig. 3 Accreted volume based on height field.

normals to the face. Here, if some of the nodal normals deviate strongly from the face normal, this indicates that those nodes are part of a high curvature feature that needs to be propagated to surrounding nodes. Thus, we give weight to node normals that deviate significantly from the local face normal. Thus, the normals averaged to the face f, $\boldsymbol{fn}_f$, are computed over all its defining nodes using

$$\boldsymbol{fn}_f = \sum_n w_n^f \boldsymbol{nn}_n / \sum_n w_n^f \tag{4}$$

where $w_n^f = \max(s(1 - \boldsymbol{fn}_f \cdot \boldsymbol{nn}_n), 0.15)$, and s is a user-selectable factor with a default value of 10.0. This weighting scheme favors the node with a larger angle between the nodal normal and the face normal. The nodes with angles less than a specified threshold will have a constant weight of 0.15. Once the face-averaged normals are computed, the nodal averages are formed from the face-averaged normals. The nodal average normal for node n, $\boldsymbol{nn}_n$, is computed over all the faces incident on the node using

$$\boldsymbol{nn}_n = \sum_f w_f \boldsymbol{fn}_f / \sum_f w_f \tag{5}$$

with w_f defined by the inverse of face area. This weighting favors faces with smaller areas, which has the effect of preventing small triangles from collapsing further for accretion in concave regions. The process is then repeated until suitable smoothness objectives are reached.

4. *Compute Maximum Stable Time-Step Fraction*

When integrating the accreted ice over time, it is possible for numerical instabilities to develop in the surface evolution. The most obvious case is caused when the projection of normals intersect. In this case, a time step that is too large will cause the surface to fold on itself. Therefore, the icing step may need to be integrated over a number of substeps during the surface evolution.

Given the nodal directions for a given face, it is possible to detect unstable surface evolution behavior by observing how the volume changes with increasing offset height h. If the node normals of a facet are parallel or diverging, then the volume will increase monotonically with increasing offset height. However, in more complex cases, the volume may reach an extremal value and then actually begin to decrease with increasing height. For these facets, we limit the substep such that this maximum volume is not exceeded. To identify faces that would exhibit this behavior in the current time step, recognize that the volume formed by extruding a triangular face using a parallel offset plane by h forms a prismatoid that has a volume given by a cubic function of height h:

$$V(h) = ah + bh^2 + ch^3 \tag{6}$$

where a, b, and c are constants determined by $\boldsymbol{p}_i$ and $\boldsymbol{n}_i$:

$$\begin{aligned} a &= 0.5\|\boldsymbol{p}_{2,1} \times \boldsymbol{p}_{3,1}\|4, \\ b &= 0.25(\boldsymbol{p}_{2,1} \times \boldsymbol{u}_{3,1} + \boldsymbol{u}_{2,1} \times \boldsymbol{p}_{3,1}) \cdot \boldsymbol{n}, \\ c &= 0.25(\boldsymbol{u}_{2,1} \times \boldsymbol{u}_{3,1}) \cdot \boldsymbol{n} \end{aligned} \tag{7}$$

with $\boldsymbol{p}_{i,j} = \boldsymbol{p}_i - \boldsymbol{p}_j$ (and similarly for $\boldsymbol{u}_{i,j}$), $\boldsymbol{u}_i = \boldsymbol{n}_i / \boldsymbol{n} \cdot \boldsymbol{n}_i$, and $\boldsymbol{n}$ is the face normal.

The first maximum value that the volume attains can be found by solving for the roots of the quadratic equation that results from differentiating Eq. (6). If the roots are imaginary or negative, the function is monotonic with increasing h and no time-step limit is implied. However, when the roots are positive real values, the smallest positive root gives the height at which the maximum volume is attained, which is denoted as $V_{\max}$. From this, we can compute a maximum fraction of the icing time step that is required to ensure reasonable behavior of volume accumulation. In addition to this step size limit, Jiao also suggested a stability limit [10] α_{Jiao} that is based on how the normal directions change as the surface evolves. Both the maximum volume test described here and Jiao's method are combined to determine the stable time-step fraction such that the maximum allowable time-step fraction for the ith face is defined as

$$\alpha_{\Delta t}^i = \begin{cases} \min\left(s_{\Delta t} \frac{V_{\max}^i}{V_f}, \alpha_{\text{Jiao}}, 1\right) & \text{if } V_{\max}^i \text{ exists} \\ \alpha_{\text{Jiao}} & \text{if } V_{\max}^i \text{ does not exist} \end{cases} \tag{8}$$

where $s_{\Delta t}$ $(0 < s_{\Delta t} < 1)$ is a safety factor, with a default value of $s_{\Delta t} = 0.25$; and V_f is the current accretion volume for the ith face. The fraction used in the surface evolution is the global minimum value of $\alpha_{\Delta t}$ for all faces.

5. *Define Height Field*

After the time-step fraction is computed, the volume accreted for the current substep is simply $\alpha_{\Delta t} V_f$. To advance the surface, it is necessary to determine the height field that will correspond to this volume; from this height field, the nodal displacements can be determined. We again employ the prismatoid volume defined in Eq. (6). Here, we have to find the height that produces the volume of $\alpha_{\Delta_t} V_f$, which is easily obtained using a bracketed Newton method where the solution is known to lie between zero and $h_{\max}$. The solution of $V(h_{\text{step}}) = \alpha_{\Delta t} V_f$ provides the initial height field that is used to advance the surface.

6. *Reduce Surface Noise by Height Smoothing*

The purpose of the optional height smoothing step is to filter out the high-frequency noise in the height field by reducing the height difference between adjacent faces. In general, the heights for two triangular faces that share an edge will not be equal. A volume-conserving height smoothing is then employed to reduce the difference by redistributing the volume.

Assume two triangles, T_1 and T_2, that share an edge, with heights h_1 and h_2, respectively, and $h_1 > h_2$. Define a volume increment

$$\Delta V = \min(h_1 - h_2, \alpha_h H) A_1 \tag{9}$$

where A_1 is the area of T_1 at the height $(h_1 + h_2)/2$, H is the maximum value of the height field, and α_h is the user-defined height smoothing threshold $(0 < \alpha_h < 1)$ with a default value of $\alpha_h = 0.2$.

The height smoothing threshold α_h is used to protect the "real" high-frequency features that exist in the surface. It is observed that, during icing, in the area where volume accretion rates are large, the height differences at edges are also large. These high-frequency features are considered real and will be protected from unnecessary smoothing. On the other hand, noise in the height field usually appears in the areas where volume accretion rates are small. In these regions, the surface may sometimes wrinkle to form a groove or similar unwanted feature. Therefore, the noise is identified by comparing the height difference with a threshold based on a fraction of the largest height value.

The volumes of T_1 and T_2 are then adjusted by a fraction of the volume increment:

$$V_1 = V_1 - \beta \Delta V \quad V_2 = V_2 + \beta \Delta V \tag{10}$$

where β is user-defined height smoothing factor: $0 < \beta < 1/2$. The default setting for this parameter is $\beta = 0.1$.

Once the new volumes are created for all of the faces in the mesh, the process of determining a new height field is repeated. Typically, 0–20 height field smoothing iterations are employed, depending on the application and the expected complexity of the simulated ice shape.

7. *Update Surface Nodal Positions*

The next step is to determine the nodal positions using the height field. As discussed by Jiao [10], the nodal position should be the intersection point of the offset planes for advective motion. In contrast, for wave front motion, the node should reside on a smooth nonlinear patch. According to Huygens principle [10], each point on

the wave front acts as a point source that emits spherical wavelets. The wave front at a later time is then defined as the envelope that encloses all of these wavelets. To simplify the computation, it is assumed that the direction of the wave front nodal displacement is the same as the direction of the advective displacement, and the nodal displacement is adjusted so that the node will be on the nonlinear patch.

The magnitude of the nodal displacement is a weighted sum of the contributions of all the faces that are incident on the node:

$$l = \frac{\sum_{i=1}^{m} w_i l_i}{\sum_{i=1}^{m} w_i} \tag{11}$$

Referring to Fig. 3, assume that h_i represents the offset distance in the direction of the face normal of the ith face. Then, the nodal offset defined for the ith face is given by $l_i = h_i/C_i$, where C_i is the contraction factor while the associated weight is given as $w_i = \phi_i C_i^2$, where ϕ_i is the inner angle of the node formed by the face. The contraction factor is defined as

$$C_i = \begin{cases} |\boldsymbol{n} \cdot \boldsymbol{n}_i| & \text{if contracting} \\ 1 & \text{if diverging} \end{cases} \tag{12}$$

After the wave front adjustment, the offset position of a node at position $\boldsymbol{p}$ with offset direction $\boldsymbol{n}_p$ is given by

$$\boldsymbol{p}'' = \boldsymbol{p} + l\boldsymbol{n}_p \tag{13}$$

8. Redistribute Remaining Volume

After the nodal offsets are computed, we need to deduct the volume accreted from the facet accretion volume V_f. However, we note that this volume will only be approximately equal to $\alpha_{\Delta t} V_f$ because the new nodal positions will not necessarily produce a face that is parallel to its original position. The resulting volume swept out by the surface evolution is actually a polyhedron. So, to update V_f, we compute the volume of this polyhedron by subdividing it into 14 tetrahedrons obtained by subdividing the quadfaces using midpoint insertion. The true volume swept out by the facet during this step V_{step} is then deducted from the volume assigned to the face:

$$V_f^{n+1} = V_f^n - V_{\text{step}}^n \tag{14}$$

This simple update comes with a potential pitfall: at the edge of an ice accumulation, the value of V_f may become negative. If this volume deficit is not corrected, the iced surface could actually regress into the wing at the ice edge. To correct for this problem, we redistribute the negative volumes to the neighboring face with the largest value for V_f. If V_f remains negative after this redistribution, it is reset to zero, causing a small loss of volume conservation. This occurs only at the edge of the icing area and is typically a trivial amount compared to the total accreted volume.

9. Redistribution of Nodes on the Evolved Surface Mesh (Null-Space Smoothing)

The evolution of the surface will tend to pack nodes into concave regions where the surface normals converge, whereas spreading occurs in convex regions where the surface normals diverge. If the nodes are not redistributed, it can become impossible to proceed with a productive, stable time step. To improve the surface mesh quality, the nodes are redistributed on the surface using a smoothing in the null space provided by the eigenanalysis. This procedure follows the basic null-space smoothing approach described by Jiao [9,10]. The technique is robust and is able to redistribute points while maintaining the integrity of the underlying geometry. The null space is defined by the tangent plane (for smooth regions), tangent line (for surface creases), or the empty set (for corners). These spaces are formed from selected eigenvalues found in the eigenanalysis described earlier.

The eigenvalue analysis will separate the primary space from the null space. The base vectors of the null space at a node $\boldsymbol{p}$ are the subset of its eigenvectors:

$$\{\boldsymbol{e}_i\}, \begin{cases} i \in \{2,3\} & \text{if } k = 1 \\ i \in \{3\} & \text{if } k = 2 \\ i \in \varnothing & \text{if } k = 3 \end{cases} \tag{15}$$

where k is the feature rank, i.e., the rank of the primary space. Also, k is one, two, or three at a smooth, ridge, or corner vertex, respectively.

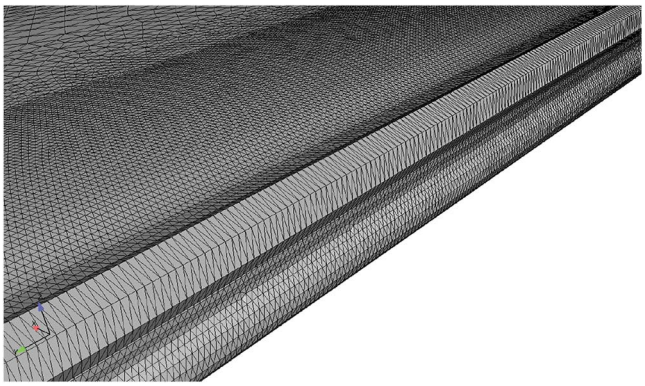
The smoothing at a given node $\boldsymbol{p}$ is implemented by first computing the displacement as the weighted average of the neighborhood of $\boldsymbol{p}$ and projecting the displacement into the null space of $\boldsymbol{p}$. If the central node $\boldsymbol{p}$ has m neighbors, the position of the central node is then adjusted to lie at the weighted average of the centroids of the incident triangles using

$$\boldsymbol{dv} = \frac{\sum_{i=1}^{m} w_i \boldsymbol{c}_i}{\sum_{i=1}^{m} w_i} \tag{16}$$

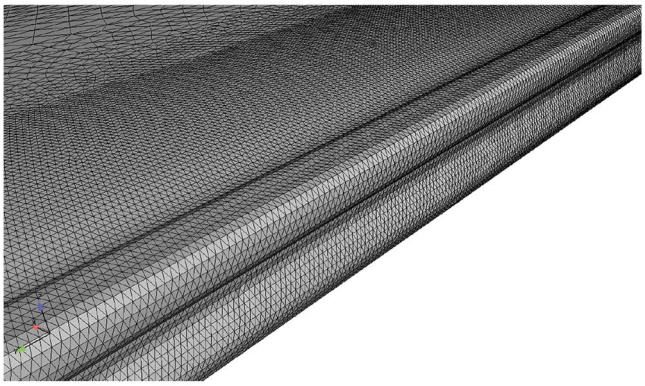
where $\boldsymbol{c}_i$ is the vector from $\boldsymbol{p}$ to the offset centroid of the ith face incident on $\boldsymbol{p}$. In the current version of the algorithm, the local weighting is based on the current area of the incident triangles divided by the area in the original mesh. Defining the weighting in this way will preserve the resolution of the original mesh, as illustrated in Fig. 4.

The final displacement is the projection of $\boldsymbol{dp}$ to the null space is

$$\boldsymbol{t} = \begin{cases} s_t \sum_{i=k+1}^{3} (\boldsymbol{dv} \cdot \boldsymbol{e}_i)\boldsymbol{e}_i & \text{if } k = 1 \quad \text{or} \quad k = 2 \\ \boldsymbol{0} & \text{if } k = 3 \end{cases} \tag{17}$$



a) Before null-space smoothing



b) After null-space smoothing

Fig. 4 Demonstration of null-space smoothing.

where s_t ($0 < s_t < 1$) is a safety factor that limits the magnitude of the displacement. The default value of s_t is taken to be 0.2. The position of $\boldsymbol{p}$ after smoothing is given by

$$\boldsymbol{p}' = \boldsymbol{p} + T \tag{18}$$

By limiting the magnitude of the displacement, $\boldsymbol{p}'$ will remain on the surface so that the volume and the surface shape can be preserved. This process is repeated for all nodes in the mesh.

Typically, many null-space smoothing iterations are used to improve the quality of surface mesh. We now illustrate the effectiveness of the null-space smoothing in Fig. 4, which shows the "before" and "after" images for a surface mesh that describes a glaze-ice shape. The elongated, high-aspect-ratio surface elements present on the horn in the before image occur because ice is growing in a convex region of the surface in which the surface normals naturally diverge. After 500 null-space smoothing steps, the quality of the surface elements is restored through the motion of the mesh points along the surface. In this case, points are moved from the clean surfaces of the wing to the horn to maintain mesh quality.

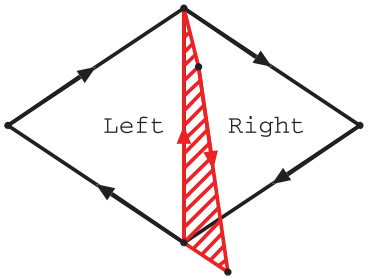


Fig. 5 Area swept by an edge (in red) used in computing volume fluxes.

10. Null-Space Smoothing Accretion Volume Interpolation

When null-space smoothing is performed between substeps, errors in the location of the volume accretion result due to the relocation of surface facets as they move along the surface mesh. To correct for this potential source of error, the accretion volume V_f is interpolated between each null-space smoothing step. The interpolation is computed using a straightforward upwinding approach, whereby the area swept along the surface by each edge is used to compute volume

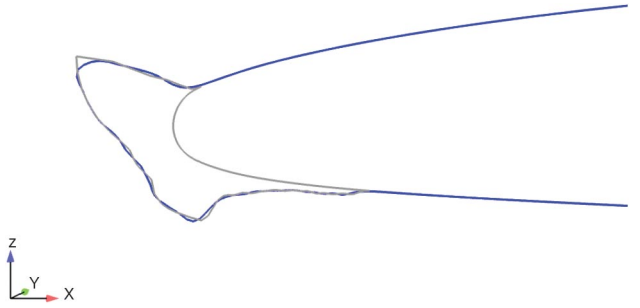
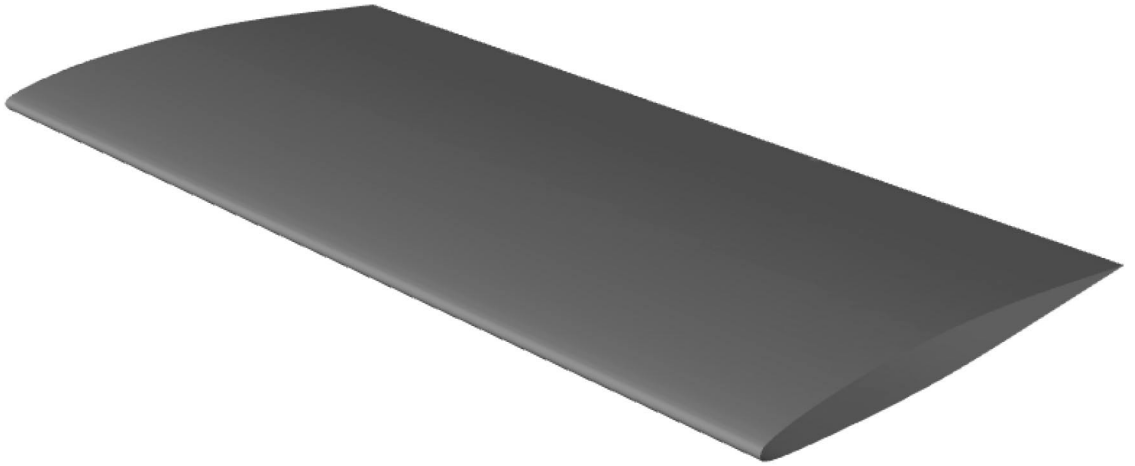
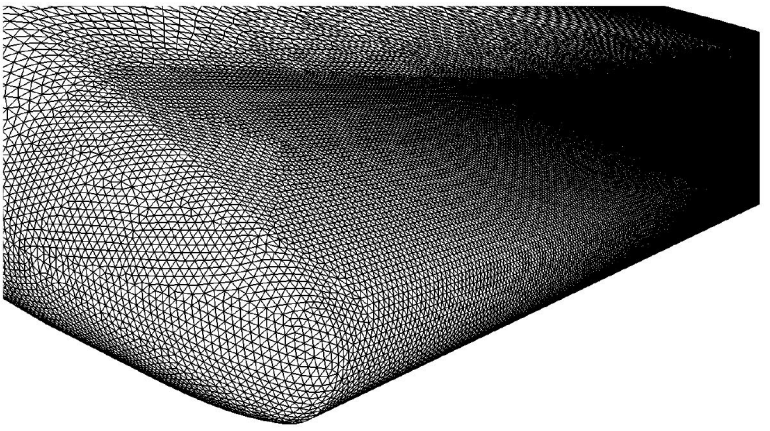


Fig. 7 Comparison of the computed single-step ice shapes for LEWICE3D (gray) and iceSurf (blue) at $t_{\text{ice}} = 22.5$ min for case 072605 [11].



a) Geometry



b) Surface mesh near wingtip

Fig. 6 Geometry and mesh for rectangular-planform wing.

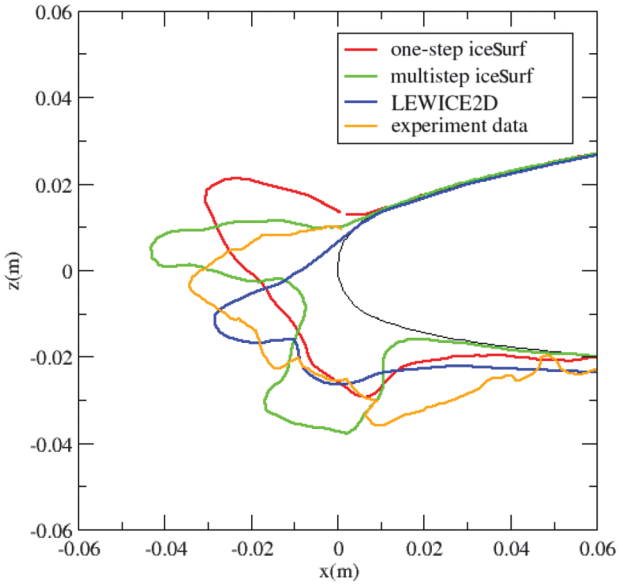


Fig. 8 Comparison between the computed ice shapes and experimental data at $t_{ice} = 22.5$ min for case 072605 [11].

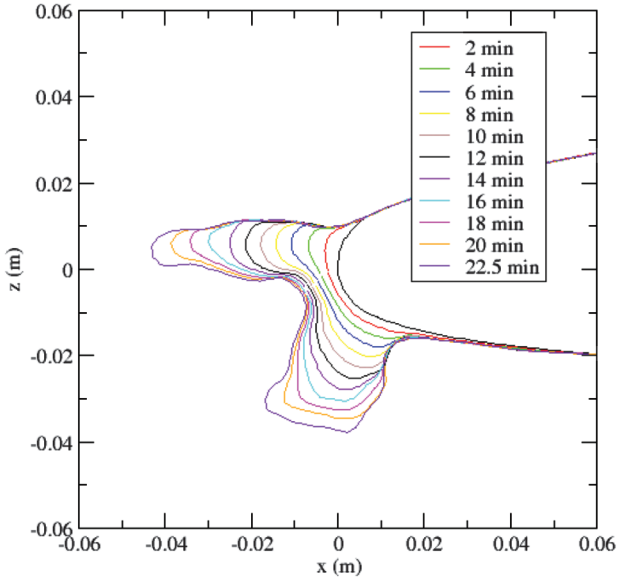


Fig. 9 Development of ice shapes generated by iceSurf using multiple steps for case 072605 [11].

fluxes to exchange between facets as they are relocated in the null-space smoothing procedure.

The interpolation of V_f is computed by looping over edges of the surface, where each edge is defined by two nodes and a left and right facet. A surface-normal direction is assigned to the edge using a simple average of the neighboring facet normals. The null-space smoothing provides a deflection for the edge nodes in which an area and normal vector are computed, as shown in Fig. 5. This edge is topologically consistent with the left face. Thus, if the swept area normal is in the same direction as the edge normal, the edge is moving into the right facet and the volume flux must be transferred from the right facet to the left. This edge volume flux is defined as

$$V_{\text{flux}} = \begin{cases} n_s \cdot n_e V_r \frac{A_s}{A_r} & : n_s \cdot n_e >= 0 \\ n_s \cdot n_e V_l \frac{A_s}{A_l} & : n_s \cdot n_e < 0 \end{cases} \tag{19}$$

where V_{flux} is the volume flux exchanged between facets; n_s is the normal vector of the swept edge area; $n_e = 1/2(n_l + n_r)$ is the edge surface normal; and A_s is the swept area, whereas A_l and A_r are the left and right facet areas, respectively. The volume flux is then added to the left facet and subtracted from the right facet. This formulation amounts to a first-order upWIND scheme for material advection. We have also implemented a second-order upWIND scheme by replacing V_l and V_r with a limited reconstruction based on surface gradients of local facet volumes. This approach yielded similar results to the first-order scheme. We suggest that the first-order scheme be preferred on the basis of robustness.

11. Final Volume Correction Step

When the value of $\alpha_{\Delta t}$ is unity, the iterations for the surface evolution are complete. However, the assigned facet volume V_f may be nonzero due to differences between the prismatoid volume and the final polyhedral volume. This remaining volume residue is then added by a final correction step to improve volume conservation. It might be tempting to iterate on this process by again computing a residue and repeating, but it was found that the resulting iteration eventually became unstable. Generally, one final corrections step is enough to bring the already small-volume conservation error to an even smaller value. Generally, this value will be well below the uncertainty present in current icing models, so it is not considered to be a major source of error.

B. Volume Mesh Deformation

Assuming a valid surface mesh of reasonable quality has been evolved, the next step in the process is the deformation of the volume mesh using gridMover, which is based on the method developed by Luke et al. [8] to perform a volume mesh deformation in response to a surface mesh deformation. Their approach took boundary displacements as input and returned a deformed volume mesh. A robust direct interpolation, which was based on an inverse distance weighted

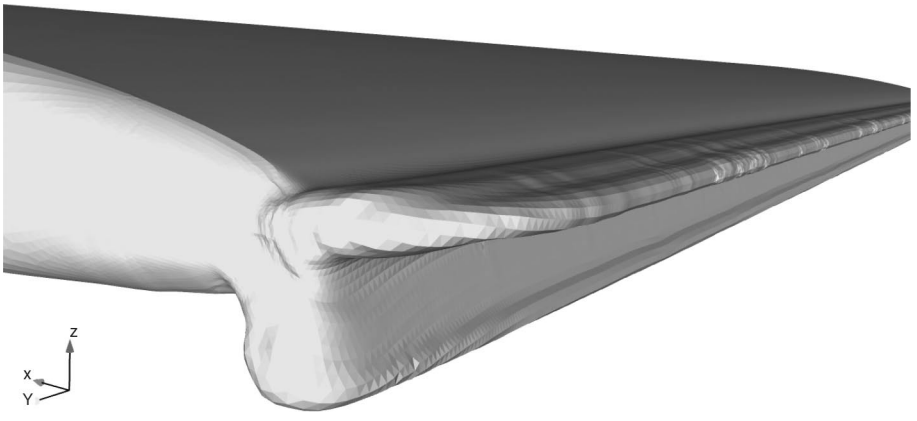


Fig. 10 Three-dimensional ice shapes generated by iceSurf at $t_{ice} = 22.5$ min for case 072605 [11].

(IDW) technique, was employed to produce the mesh motion. A unique feature of this approach is the specification of both a displacement and a local rotation for each node of the surface mesh. The local rotation for a given node is computed by using a least-squares fitting to determine the rotation about the node that best matches the displacements of all edges and normals from surface facets that reference the given node. In this method, node i on the deforming surface produces a displacement field $\boldsymbol{s}_i(\boldsymbol{r})$ that is computed using

$$\boldsymbol{s}_i(\boldsymbol{r}) = M_i\boldsymbol{r} + \boldsymbol{b}_i - \boldsymbol{r} \tag{20}$$

where M_i is the rotation matrix, $\boldsymbol{b}_i$ is the displacement vector associated with the ith node, and $\boldsymbol{r}$ is the coordinate vector for the original mesh. The displacement field in the volume mesh is computed using a weighted average of all boundary node displacement fields:

$$\boldsymbol{s}(\boldsymbol{r}) = \frac{\Sigma w_i(\boldsymbol{r})\boldsymbol{s}_i(\boldsymbol{r})}{\Sigma w_i(\boldsymbol{r})} \tag{21}$$

We chose a function of the reciprocal of distance for the weight function $w_i(\boldsymbol{r})$. A two-exponent weight function is employed so that we can preserve near-boundary deformations, which is critical for

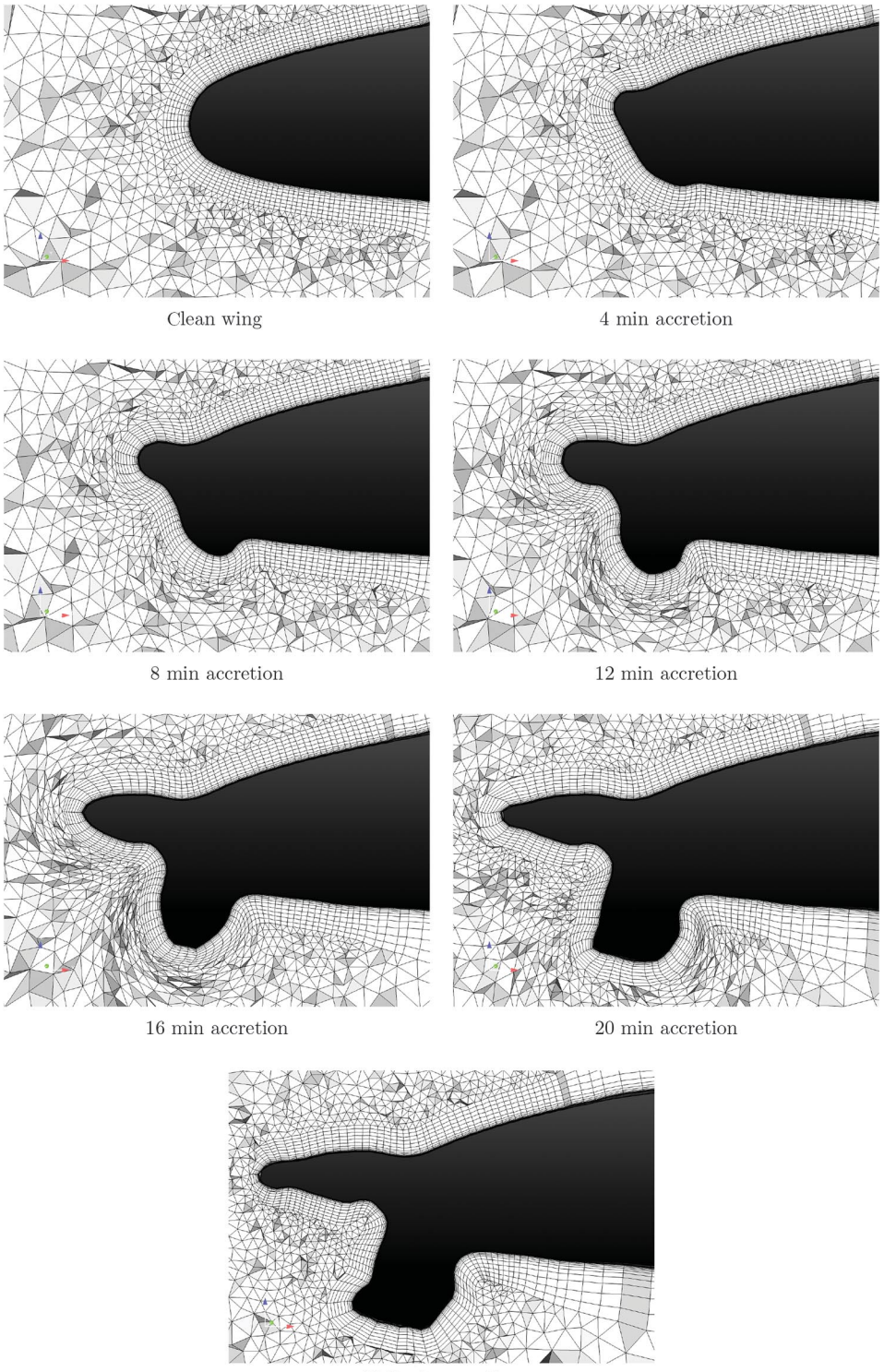


Fig. 11 Evolution of the volume mesh in the midspan plane.

high-aspect-ratio viscous meshes, while providing a smooth transition to the interior of the domain. We also use the area of the surface facet in the weighting function so that mesh refinement of a region does not increase its influence in the interpolation.

A tree-code-based fast approximation algorithm is used to evaluate Eq. (21) in $n \log n$ time. This accelerated IDW approach has been shown to be competitive with the considerably more expensive radial basis function proposed by deBoer et al. [47]. In fact, the IDW approach does a better job of preserving the orthogonality of the mesh in the near-wall viscous regions, which makes it an ideal candidate for ice accretion prediction. More details were given by Luke et al. [8].

When the surface mesh evolves very complex geometric features, gridMover may fail to generate a volume mesh with acceptable quality. Typically, this occurs when there is shearing of the mesh induced by null-space smoothing in regions of high curvature, such as indentations. In these cases, it is necessary to regenerate the volume grid.

C. Robustness of Meshing for Icing Applications

The algorithm presented in this paper seeks to provide a robust foundation for surface evolution without introducing topological change in the surface (that is, edges, nodes, or facets are not created or destroyed). Ultimately, this will limit the robustness of the algorithms because defeaturing, mesh refinement, or edge-feature alignment will be necessary to evolve an arbitrarily complex surface. In addition to topological changes in the surface mesh, topological changes in the volume mesh may be required to accommodate very large deformations of the surface, even when no topological change has occurred on the surface. We observe that degradation of volume mesh quality is the critical limiting factor during the course of an icing simulation. Although null-space smoothing improves the surface mesh quality, there is a tradeoff with volume mesh quality, which can be degraded because the surface smoothing introduces shearing in the volume mesh. Generally, once the volume mesh quality degrades and

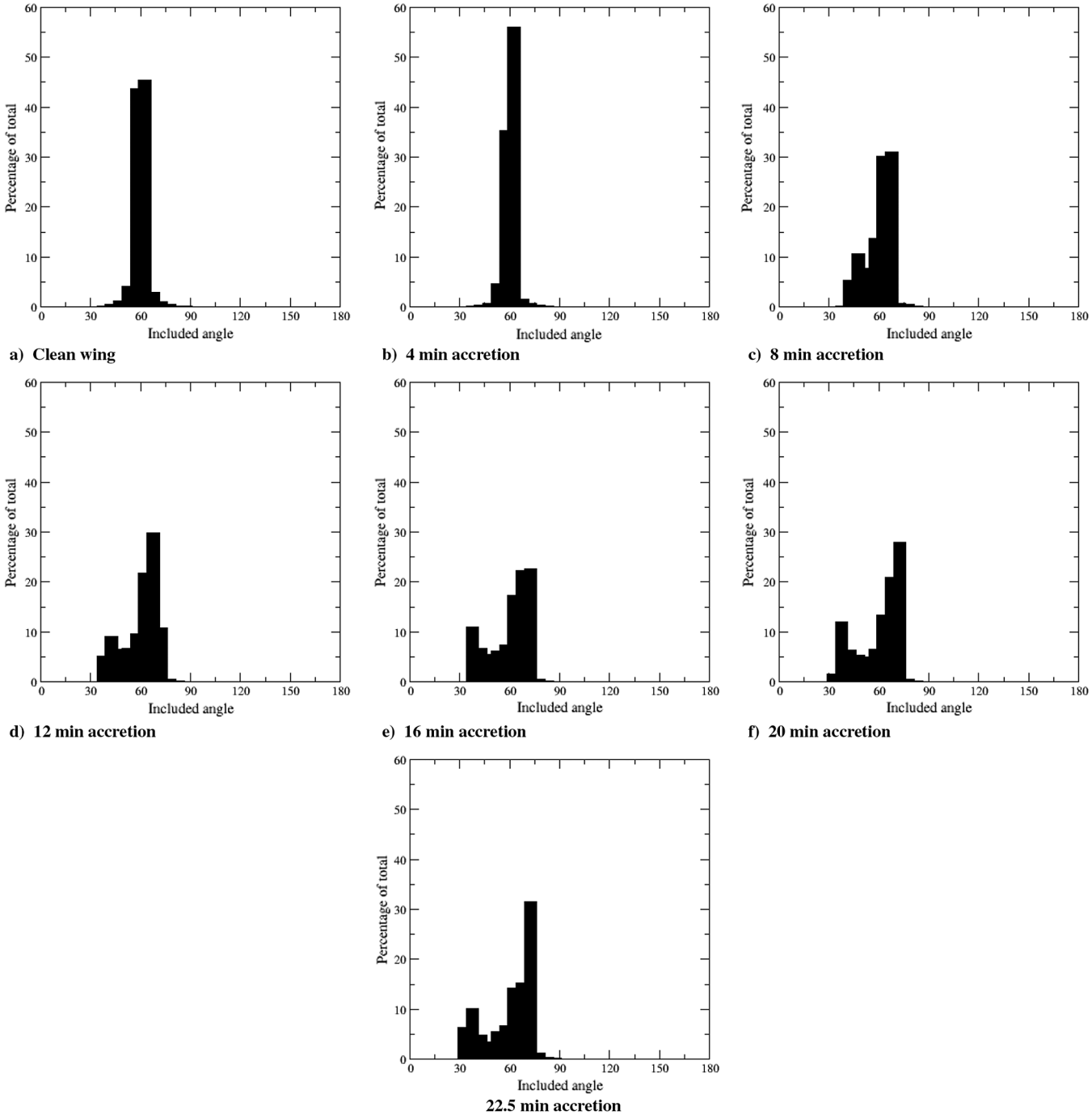


Fig. 12 Evolution of surface mesh quality.

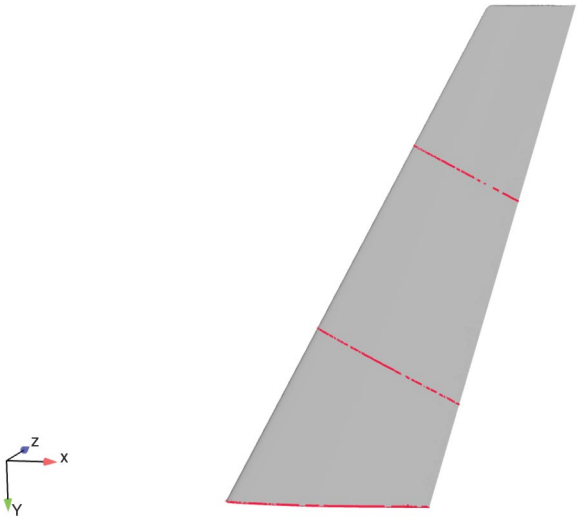


Fig. 13 Geometry of the GLC305 swept wing [3]. Red line segments show locations where the ice sections are extracted.

a new mesh is regenerated around the resulting complex shape, the regenerated mesh is usually less deformable because the null-space smoothing pulls the mesh through troughs and valleys to which the regenerated mesh structure specifically conforms. Therefore, it would be more productive to start with a clean wing with the surface mesh refined in the general area where icing is expected to occur. This strategy can delay expensive volume remeshing to only the final steps of a complex ice shape. Generally, we have found that the ability of the volume deformation algorithm to conform the clean wing mesh to the ice shape dramatically exceeded our initial expectations. In our tests, volume remeshing has only been required after significant evolution of complex glaze-ice shapes.

With regard to the robustness of the surface evolution, most of the parameters employed in the icing simulations were developed through application to a suite of test shapes where a uniform accretion rate was imposed everywhere. These shapes included convex, concave, and saddle-point geometries, as well as ridgelines and corners. Some of the shapes had analytical solutions for their shape evolution (e.g., a sphere that evolved to a progressively larger and larger sphere), whereas others had specific topological difficulties, such as the emergence of ridge lines where accreting surfaces collide. In nearly all cases, the values of the parameters were not critical to robustness but were optimized using these test cases to provide the best results, e.g., agreement with analytical solutions where available. Often, the parameters only had an effect (by design) on very localized regions where problematic evolution of the surface was observed. For surfaces that did not contain these problem features, the values of these parameters had no effect.

With regard to icing simulations, the main parameter used to control robustness was the number of height smoothing steps. In a sense, the height smoothing was used as a defeaturing tool. It was a low-pass filter that removed high-frequency content produced by the icing model. Generally, if sufficient height smoothing was employed, the resulting algorithm could become very robust with the cost that the resulting ice shapes were excessively smoothed and contained little local detail. When the surface evolution was one that tended to generate smooth ice shapes, such as under rime-icing conditions, it was possible to eliminate height smoothing altogether. However, for icing conditions that tended to produce complex shapes, such as those that occurred under glaze-icing conditions, employing height smoothing at the early stages of the simulation was usually needed to obtain robust outcomes. On the other hand, ice shapes with even greater detail than those presented in this paper could be obtained by using less height smoothing and manually changing the smoothing parameter as the ice shape evolved. However, it should be noted that the roughness of the surface has a cumulative effect. Therefore, once surface irregularity developed, height smoothing would not remove it because the smoothing was only applied to the volume accreted during subsequent icing steps. Thus, it may be necessary to recompute a few steps in the process and manually increase the number of smoothing steps in order to avoid the evolution of a problematic surface detail. As a general rule of thumb, height smoothing is most productive to include in the initial stages of an icing simulation and can be relaxed as the simulation proceeds to the target time. Height smoothing is generally not required to evolve a single icing step in a robust manner.

We now suggest possible approaches for further enhancing robustness. Typically, a lack of robustness in surface evolution arises when high-frequency features evolve for which there is insufficient resolution to provide an unambiguous representation in the discrete representation of the surface. One strategy that might be effective for problems of this type is topological change. For example, mesh refinement is one such method that could yield improved robustness. However, without any control of the refinement, the problem can quickly become intractably large. Explicit filtering of high-frequency features in the surface using height smoothing has been demonstrated to be an effective defeaturing tool. Unfortunately, it is relatively scale indiscriminate as currently implemented. A more desirable capability would be to eliminate features smaller than a specified spatial scale while only minimally altering those at larger scales. Therefore, more work is needed to identify a quantifiable relationship between height smoothing and scale resolution. Filtering of this type would work in concert with an icing model for which the scale at which features could be reliably resolved was well understood. Features at a smaller scale (i.e., unresolvable features) could be removed via the improved height smoothing strategy.

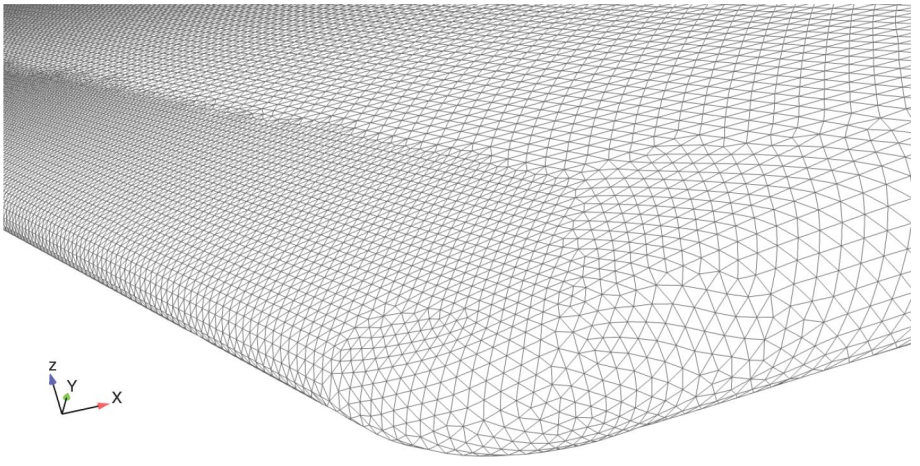


Fig. 14 Surface mesh distribution near the wingtip at the leading edge for the swept, tapered, and twisted GLC-305 wing.

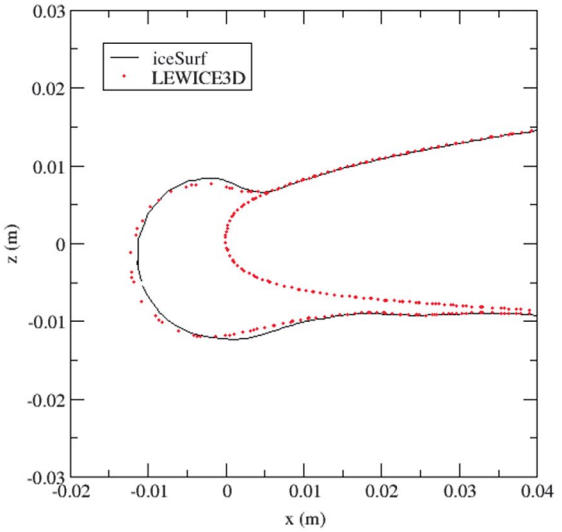
V. Results

In this section, we present results that demonstrate the efficacy of the mesh evolution and deformation algorithms. The algorithms are employed in a loosely coupled approach using LEWICE3D [4] and Loci/CHEM [5] to simulate ice accretion on two different configurations. We first consider ice accretion on a rectangular-planform wing with a constant GLC-305 cross section for a two-dimensional, but nevertheless challenging, glaze-icing condition [11]. Then, we present rime-ice results for a truly three-dimensional accretion on a swept and tapered wing with a GLC-305 section [3]. Comparisons between the simulation results and experimental ice shapes are included. However, as noted previously, the purpose of these comparisons is to demonstrate that our mesh evolution and deformation strategies are robust, even for complex ice shapes, rather than to evaluate the overall coupled strategy.

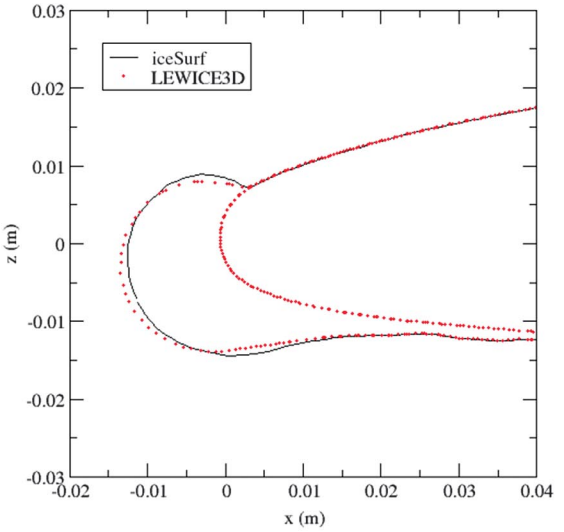
The results in this section were produced using an automated script that ran the computational fluid dynamics (CFD), icing models, surface evolution, and mesh deformation steps sequentially. Generally, user intervention was limited to steps where the quality of the volume mesh became unsuitable for the CFD solver, in which case the volume mesh was regenerated and the script was restarted from that step. The glaze-ice simulation used height smoothing in the initial steps in anticipation of a more complex ice shape, whereas the rime-ice case did not employ height smoothing. Otherwise, the example cases used the default settings of parameters identified in the algorithm description.

A. Rectangular-Planform Wing with GLC-305 Cross Section

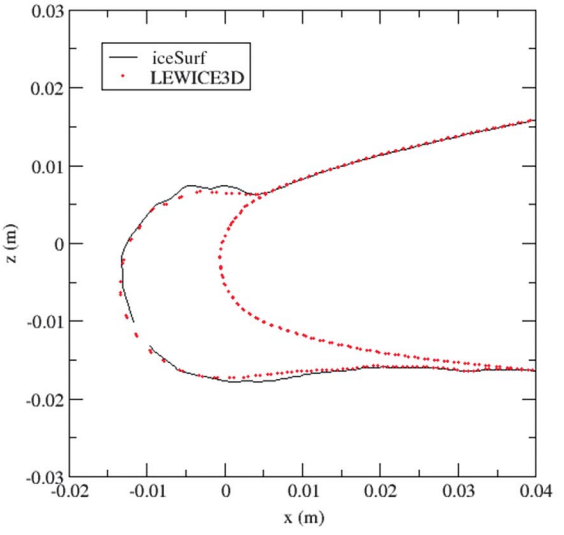
Loosely coupled Loci/CHEM-LEWICE3D simulations of ice accretion on a rectangular-planform wing with a constant GLC-305 airfoil section were performed, and the results were compared with available LEWICE and LEWICE3D simulation results and experimental data [11]. The chord of the wing was 0.9144 m, and the span was 1.8288 m. The wing planform is shown in Fig. 6a. The initial mesh consisted of approximately 12 million elements. The initial mesh on the surface near the wingtip is shown in Fig. 6b. The flight conditions and icing conditions were chosen to match cases documented in the LEWICE validation report [11]. The freestream velocity and pressure were 90 m/s and 1 atm, respectively. The angle of attack was 4.5 deg. A far-field boundary condition was imposed approximately 10 chords in front of and behind the wing. A far-field boundary condition was also applied at the top, bottom, and outboard side boundaries of the computational domain, all of which were located approximately five chords from the wing. A symmetry boundary condition was employed at the midspan symmetry plane. Menter's baseline turbulence model [48], which is a blend of $k-\omega$ and $k-\epsilon$ models, was adopted to model the effects of turbulence in the Reynolds-averaged Navier–Stokes simulation.



Near tip of the wing ($Y = -1.1214$ m)



Near middle of the wing ($Y = -0.5606$ m)



Root of the wing ($Y = 0$ m)

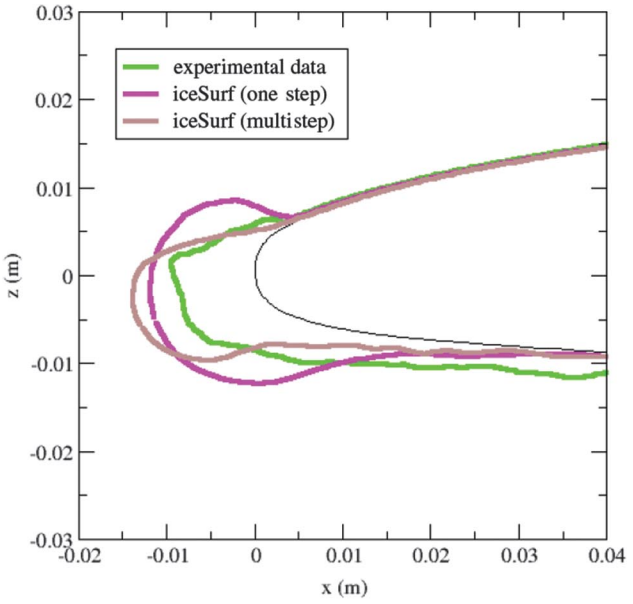
Fig. 15 Comparison of the ice shape generated from LEWICE3D and one-step iceSurf at $t = 5$ min for condition 3 [3].

To avoid resolving the droplet collection efficiency on virtual boundary surfaces, the far-field and symmetry planes were removed from the surface element generation process. The droplet release box covered the wingspan direction and a significant portion of the domain in the vertical direction. A droplet tracking window was specified near the airfoil covering the whole span. The parcel-based collection efficiency method was used to compute the droplet collection efficiency.

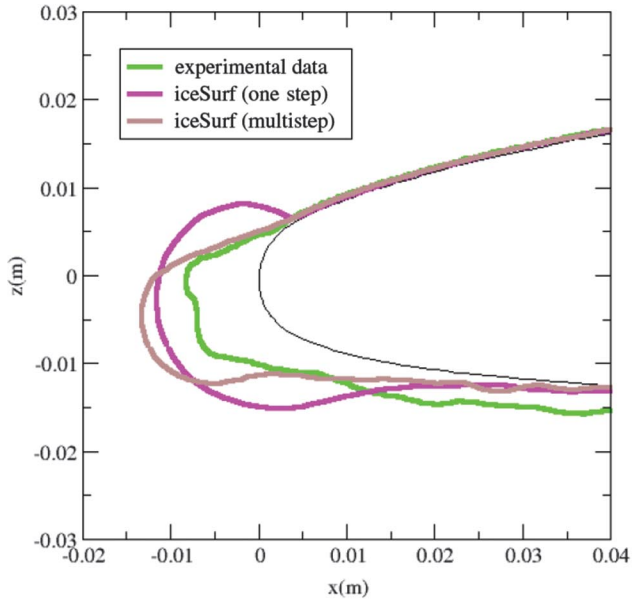
As mentioned in the Sec. IV.A, iceSurf may subdivide the desired icing time into a number of subintervals, which may be needed to ensure stability in the surface evolution algorithm. Since it is assumed that accretion occurs normal to the surface and the surface normals evolve with accretion substeps, the resulting shape can be dramatically affected by the number of substeps employed for the time integration. Additionally, to ensure volume conservation, the actual volume swept out by each face during the integration is tracked. In the comparisons that follow, we reiterate that only single-step ice accretion calculations can be performed using the baseline LEWICE3D capability.

We chose a glaze-ice case that corresponded to case 072605 in the LEWICE validation report [11] to show the efficacy of mesh evolution and deformation for ice accretion on the rectangular-planform wing. The icing conditions were a liquid water content (LWC) of 0.43 g/m^3; an ambient temperature and pressure of 263.2 K and 1 atm, respectively; and a relative humidity of 100%. The droplet diameter was 20 μm. At these conditions, runback occurred and a glaze-ice shape was accreted. Even though the ice shape was two-dimensional, this case was very challenging due to the severe deformation of the final ice shape caused by the long icing time (22.5 min) and glaze-icing conditions.

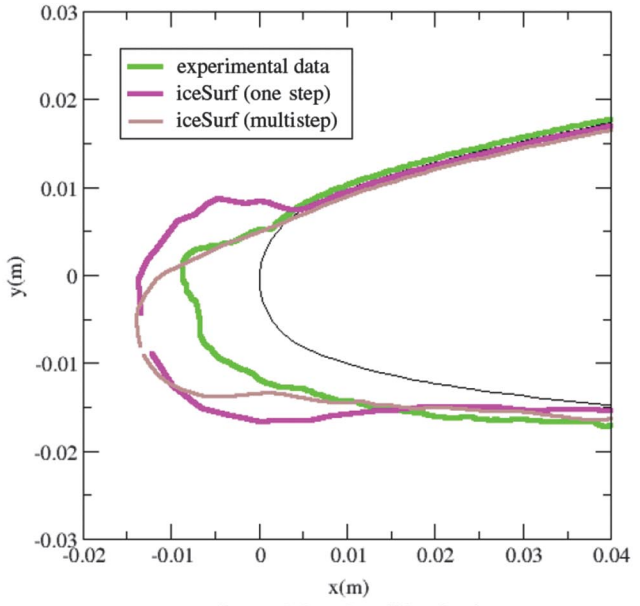
Since the flowfield surrounding the airfoil changed in response to the ice accretion that, in turn, produced a change in the collection efficiency, ice accretion rate, etc., computing the ice shape with the initial ice accretion rate computed for the clean wing was not appropriate for long icing times. Instead, we divided a majority of 22.5 min total icing time into 2 min intervals. That is, the flowfield and ice accretion rates were recomputed every 2 min based on the newly deformed mesh obtained from iceSurf and gridMover. During



Near tip of the wing ($Y = -1.1214$ m)



Near middle of the wing ($Y = -0.5606$ m)



Root of the wing ($Y = 0$ m)

Fig. 16 Comparison of the ice shape generated from iceSurf with experimental data at $t = 5$ min for condition 3 [3].

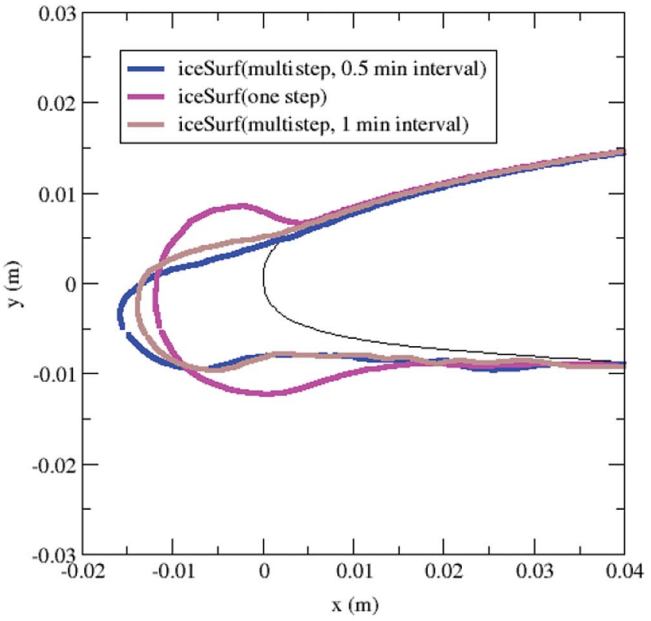
the first 14 min, height smoothing was applied to prevent the formation of high-frequency surface features. To maintain surface mesh quality while preserving the shape of the iced geometry, null-space smoothing was employed from the beginning of the simulation. The height smoothing technique was removed after 14 min, and the final ice shape was achieved by applying normal smoothing and regenerating the volume grid whenever the quality of the volume grid deteriorated.

First, we show a comparison of ice shapes computed using LEWICE3D and a single-step iceSurf calculation in Fig. 7 to demonstrate the efficacy of the mesh evolution strategy. In both cases, a single accretion step was employed. The results from iceSurf were obtained using 500 null-space smoothing steps. The ice shapes compare very favorably, indicating that the mesh evolution algorithm can successfully reproduce a discrete geometry in response to the specified ice growth obtained from LEWICE3D.

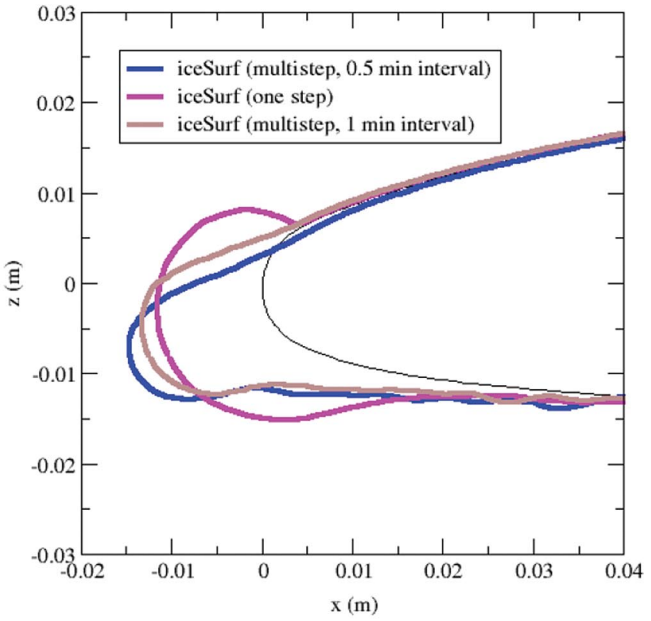
A comparison of ice shapes between single-step iceSurf, multistep iceSurf, LEWICE, and experimental results is shown in Fig. 8. Growth of ice on the leading edge combined with the changes in the surrounding flowfield shift the stagnation point aft along the lower surface of the wing, causing the upper part of the upper horn to shift downward. Compared with the single-step solution, the ice shape generated by the multistep iceSurf computation exhibits more glazelike characteristics.

Figure 9 shows a time history of the ice shape evolution. It can be seen that the classic horned glaze-ice shape developed gradually during each time interval, producing a highly deformed surface. The three-dimensional ice shape at $t = 22.5$ min is displayed in Fig. 10, which clearly shows the ice shape that emerges at the end cap of the airfoil. It demonstrates the robustness of the surface mesh generation tool for three-dimensional geometries.

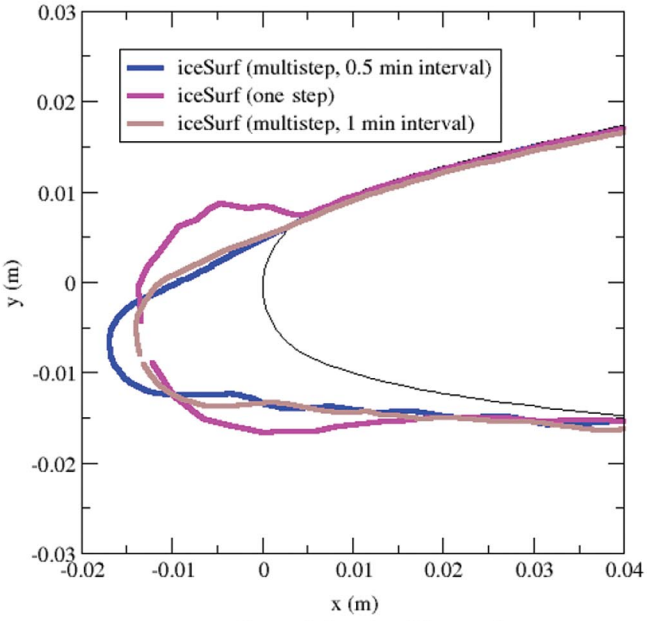
Figure 11 shows the temporal evolution of the mesh in a cutting plane located at $y = 0.9144$ m, near the midspan of the wing. These images show that, as the surface mesh evolved, the volume mesh was deformed in a manner that maintained visual mesh quality.



Near tip of the wing ($Y = -1.1214$ m)



Near middle of the wing ($Y = -0.5606$ m)



Root of the wing ($Y = 0$ m)

Fig. 17 Comparison of the ice shape at $t = 5$ min generated from iceSurf between different time intervals for condition 3 [3].

Figure 12 quantitatively demonstrates that the geometrical quality of the discrete surface description is maintained during the evolution of the ice shape. In this set of images, mesh quality is evaluated in terms of the included angles of the triangular surface elements. In the mesh for the clean wing, the included angles are clustered around 60 deg, i.e., the included angles of an equilateral triangle. Null-space smoothing is employed from the initiation of the ice evolution. Although there is some degradation of the surface mesh quality as it evolves, these figures illustrate that null-space smoothing maintains reasonable quality of the surface mesh, even through large deformations. For this case, the final total volume error of the ice shape versus the icing model predicted volume accumulation is just 3.65%, showing that the scheme maintains a low volume error over many icing steps, despite relatively high geometric complexity.

B. Swept, Tapered Wing with GLC-305 Cross Section

Results are now presented for ice accretion on a more geometrically complex three-dimensional swept and tapered wing with a GLC-305 section [3], as shown in Fig. 13. The half-span of the wing is 1.52 m, and the chord lengths are 0.64 and 0.256 m at the root and tip, respectively. The wing has a 28 deg leading-edge sweep. The wing has an aspect ratio of 6.80, a taper ratio of 0.4, and a geometric twist of 0 deg at the root and −4 deg at the tip. The wing twist is not evident in the figure. The flight and icing conditions are chosen to match a case documented in the report by Vargas et al. [3]. The flight and icing conditions are selected to create a rime-ice accretion. Imposed boundary conditions, numerical models, and execution procedure are the same as the rectangular wing case described in the previous section. The mesh in the region of the leading edge near the tip of the wing is shown in Fig. 14. The mesh resolution is approximately 20 million elements in the computational domain.

The flight and icing conditions for the swept wing case corresponded to condition 3 in the work of Vargas et al. [3]. The freestream velocity and pressure were 89.5 m/s and 93,382 Pa, respectively. The angle of attack was 6 deg at the root of wing. The icing conditions considered in this case were a LWC of 0.51 g/m^3, an ambient temperature of 257 K, and a relative humidity of 100%. The median volumetric diameter (MVD) of the droplets was 14.5 μm, and the spray time was 5 min. This case was chosen because it should have only minimal runback. This was necessary because LEWICE3D does not currently have a three-dimensional runback model.

Figure 15 shows a comparison of ice shapes generated from LEWICE3D and iceSurf at three different spanwise locations along the leading edge: 0.46 m (tip), 1.09 m (midspan), and the root

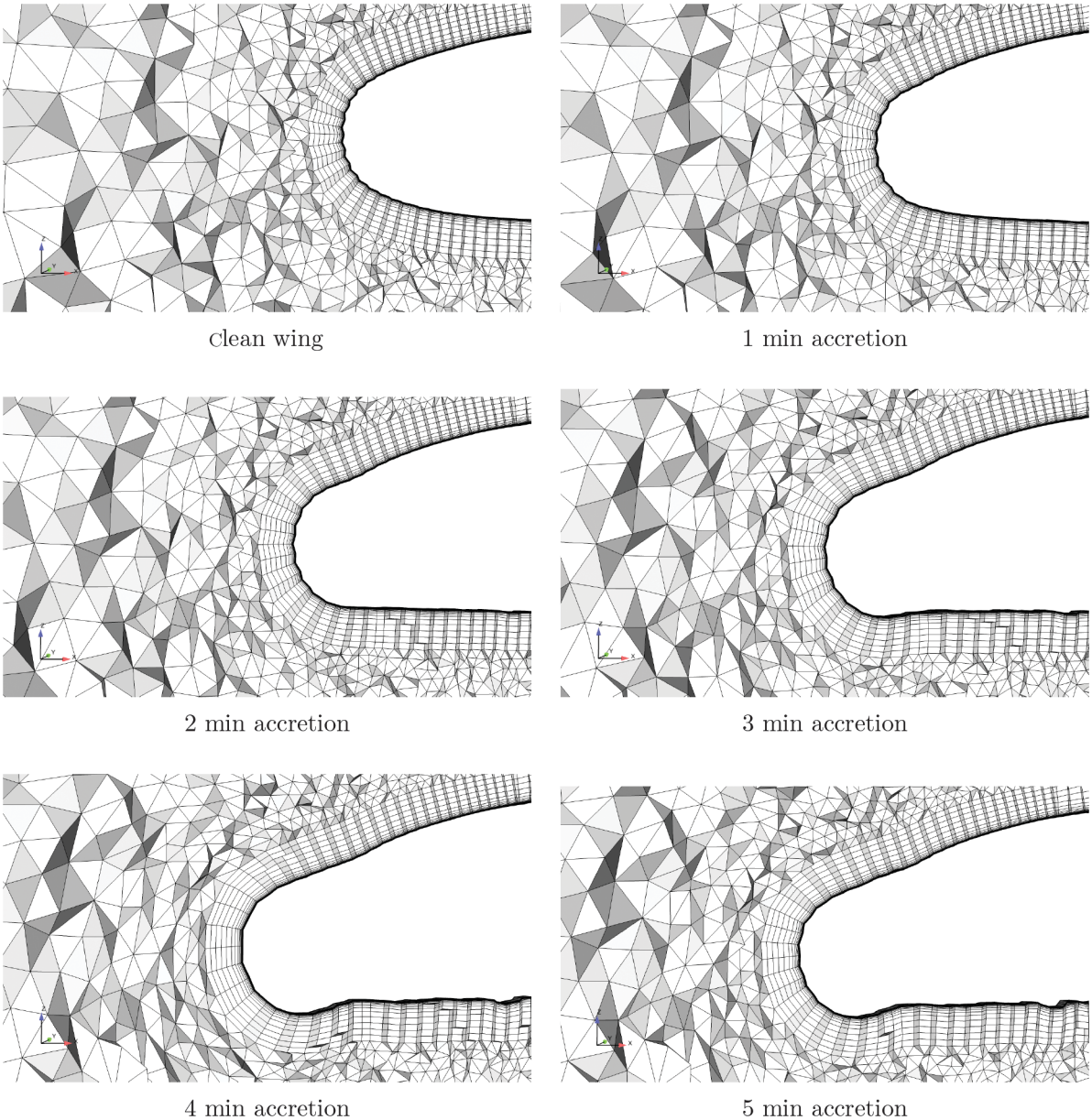


Fig. 18 Evolution of the volume mesh near the midspan plane ($Y = -0.5606$ m).

(as shown in Fig. 13). The results from iceSurf shown in Fig. 15 were obtained using a single-step run. The ice shapes produced by iceSurf and LEWICE3D agree favorably, demonstrating the efficacy of the mesh evolution tool to reproduce the shape predicted by LEWICE3D.

A comparison of ice shapes at the three cutting planes that were obtained using the single-step iceSurf and multistep iceSurf methods and experimental data is shown in Fig. 16. The results from iceSurf were advanced at 1 min intervals. Due to the general smoothness of shape in comparison to the glaze-ice shape described in the previous section, neither height smoothing nor normal smoothing was needed. Null-space smoothing was employed throughout the simulation to maintain the surface mesh quality. Three-dimensional effects were apparent in the differences between the ice shapes at the different spanwise locations. The variation in the ice shape was caused by the different thicknesses and effective angles of attack at each cross section. It appeared that the multistep simulation using iceSurf successfully predicted the inflection on the upper surface of the accretion. The general shape for the multistep run compared more favorably with the experimental results than did results from the single-step run.

Figure 17 illustrates that the size of the time interval employed in the icing simulation can greatly affect the final ice shape. Issues related to a multiple-step run in LEWICE3D need to be addressed before more accurate results for a multiple-step run can be achieved. Also, iceSurf is successful at reproducing the geometry of the ice surface provided by LEWICE3D.

The evolution of the volume mesh around the leading edge at 1 min intervals for the three-dimensional wing case is illustrated in Fig. 18. The cutting plane is near the midspan of the wing ($Y = -0.5606$ m), and the cutting line is normal to the leading edge. It is shown that the quality of the deformed mesh is maintained as the ice shape evolves. In addition, the volume error for this case is just 1.83%, showing close agreement with the icing model volume rates.

VI. Future Work

As noted in Sec. I, our overall meshing strategy is based on maintaining the quality of the surface mesh. Since the volume mesh generation process is highly automated (provided a valid high-quality surface mesh exists), the key to successful automation of the mesh generation process for ice accretion simulations is maintaining the quality of the discrete geometry representation, i.e., the surface mesh. As described in Sec. IV.A, the null-space smoothing employed in iceSurf serves to mitigate surface mesh quality issues to some degree. However, if the quality of the surface mesh becomes significantly degraded, mesh repair may be needed. Future work will be focused on surface repair algorithms to repair highly deformed surface mesh while still preserving geometric features. Techniques such as edge flipping, edge splitting, and edge contraction [49] will be adopted and tested on icing cases with highly deformed surface elements. Another challenging issue is how to reduce sharp, unphysical noise on the discrete surface definition. Finally, volume mesh smoothing or untangling [50] is needed to repair the volume mesh in some cases with large deformations.

VII. Conclusions

In this paper, a meshing strategy appropriate for simulating the accretion of ice on aerodynamic surfaces is described. The approach is based on a discrete surface mesh evolution algorithm that employs an eigenvalue analysis to define the primary space, which is the direction of growth and a null-space in which the surface mesh may be smoothed to maintain mesh quality while preserving the accreted volume. Local surface smoothing, which prevents folded faces, is also applied for more challenging geometries. A fast algebraic approach is employed to project the surface deformation into the volume mesh. These techniques are demonstrated by application to coupled LEWICE3D–Loci/CHEM simulations for a complex ice shape on a rectangular-planform constant section GLC-305 wing, as well as rime-ice accretions on a three-dimensional GLC-305 swept wing. Results show that the mesh generation tools are capable of handling relatively complex shapes and capturing simple three-dimensional ice accretion features.

Acknowledgment

This effort was supported by the NASA John H. Glenn Research Center under Cooperative Agreement NNX12AC22A.

References

[1] Jung, S., Dorrestijn, M., Raps, D., Das, A., Megaridis, C., and Poulikakos, D., "Are Superhydrophobic Surfaces Best for Icephobicity?" *Langmuir*, Vol. 27, No. 6, 2011, pp. 3059–3066. doi:10.1021/la104762g

[2] Bragg, M., Broeren, A., and Blumenthal, L., "Iced-Airfoil Aerodynamics," *Progress in Aerospace Sciences*, Vol. 41, No. 5, 2005, pp. 323–362. doi:10.1016/j.paerosci.2005.07.001

[3] Vargas, M., Papadakis, M., Potapczuk, M., Addy, H., Sheldon, D., and Giriunas, J., "Ice Accretions on a Swept GLC-305 Airfoil," NASA TM-2002-211557, April 2002.

[4] Bidwell, C., "Ice Particle Transport Analysis with Phase Change for the E3 Turbofan Engine Using LEWICE3D Version 3.2," AIAA Paper 2012-3037, June 2012.

[5] Luke, E., and Cinnella, P., "Numerical Simulations of Mixtures of Fluids Using UpWIND Algorithms," *Computers and Fluids*, Vol. 36, No. 10, 2007, pp. 1547–1566. doi:10.1016/j.compfluid.2007.03.008

[6] Thompson, D., Tong, X.-L., Arnoldus, Q., Collins, E., McLaurin, D., Luke, E., and Bidwell, C., "Discrete Surface Evolution and Mesh Deformation for Aircraft Icing Applications," *5th AIAA Atmospheric and Space Environments Conference*, AIAA Paper 2013-2544, June 2013.

[7] Tong, X., Thompson, D., Arnoldus, Q., Luke, E., and Bidwell, C., "Robust Surface Evolution and Mesh Deformation for Three Dimensional Aircraft Icing Applications on a Swept GLC-305 Airfoil," *6th AIAA Atmospheric and Space Environments Conference*, AIAA Paper 2014-2201, June 2014.

[8] Luke, E., Collins, E., and Blades, E., "A Fast Mesh Deformation Method Using Explicit Interpolation," *Journal of Computational Physics*, Vol. 231, No. 2, 2012, pp. 586–601. doi:10.1016/j.jcp.2011.09.021

[9] Jiao, J., "Volume and Feature Preservation in Surface Mesh Optimization," *Proceedings of the 15th International Meshing Roundtable*, Springer–Verlag, Berlin, 2006, pp. 359–374.

[10] Jiao, J., "Face Offsetting: A Unified Approach for Explicit Moving Interfaces," *Journal of Computational Physics*, Vol. 220, No. 2, 2007, pp. 612–625. doi:10.1016/j.jcp.2006.05.021

[11] Wright, W., and Rutkowski, A., "Validation Results for LEWICE 2.0," NACA CR-1999-208690, Jan. 1999.

[12] Gent, R., "TRAJICE2: A Combined Water Droplet and Ice Accretion Prediction Code for Aerofoils," Royal Aircraft Establishment TR-90054, Farnborough, U.K., 1990.

[13] Brahimi, M., Tran, P., Paraschivoiu, I., and Tezok, F., "Ice Accretion on Supercritical and Multi-Element Airfoils," AIAA Paper 1995-0754, Jan. 1995.

[14] Mingione, G., Brandi, V., and Saporiti, A., "A 3D Ice Accretion Simulation Code," AIAA Paper 1999-0247, Jan. 1999.

[15] Verdin, P., Charpin, J., and Thompson, C., "Multistep Results in ICECREMO2," *Journal of Aircraft*, Vol. 46, No. 5, 2009, pp. 1607–1613. doi:10.2514/1.41451

[16] Montreuil, E., Chazottes, A., Guffond, D., Murrone, A., Caminade, F., and Catris, S., "Enhancement of Prediction Capability in Icing Accretion and related Performance Penalties—Part I: Three-Dimensional CFD Prediction of the Ice Accretion," AIAA Paper 2009-3969, June 2009.

[17] Wright, W., "User Manual for the NASA Glenn Ice Accretion Code LEWICE Version 2.2.2," NASA CR-2002-21179, Aug. 2002.

[18] Pendenza, A., Habashi, W. G., and Fossati, M., "A 3D Mesh Deformation Technique for Irregular In-flight Ice Accretion," *International Journal for Numerical Methods in Fluids*, Vol. 79, No. 5, Oct. 2015, pp. 215–242. doi:10.1002/fld.4049

[19] Szilder, K., and Lozokowski, E., "Progress Towards a 3D Numerical Simulation of Ice Accretion on a Swept Wing Using the Morphogenetic Approach," SAE Paper 2015-01-2162, June 2015.

[20] Szilder, K., and Lozokowski, E., "Novel Two-Dimensional Modeling Approach for Aircraft Icing," *Journal of Aircraft*, Vol. 41, No. 4, 2004, pp. 854–861. doi:10.2514/1.470

[21] Messinger, B., "Equilibrium Temperature of an Unheated Icing Surface as a Function of Airspeed," *Journal of the Aeronautical Sciences*, Vol. 20, No. 1, 1953, pp. 29–42. doi:10.2514/8.2520

[22] Anon., "WIND User's Guide: Introduction," NASA Glenn Research Center, 2002, http://www.grc.nasa.gov/www/winddocs/user/intro.html.

[23] Thompson, D., and Soni, B., "ICEG2D: A Software Package for Ice Accretion Prediction," *41st Aerospace Sciences Meeting and Exhibit*, AIAA Paper 2003-1070, Jan. 2003.

[24] Bourgault, Y., Beaugendre, H., and Habashi, W., "Development of a Shallow-Water Icing Model in FENSAP-ICE," *Journal of Aircraft*, Vol. 37, No. 4, 2000, pp. 640–646. doi:10.2514/2.2646

[25] Fossati, M., Khurram, R., and Habashi, W., "An ALE Mesh Movement Scheme for Long-Term In-Flight Ice Accretion," *International Journal for Numerical Methods in Fluids*, Vol. 68, No. 8, 2014, pp. 958–976. doi:10.1002/fld.v68.8

[26] Sethian, J. A., "Fast Marching Methods," *SIAM Review*, Vol. 41, No. 2, 1999, pp. 199–235. doi:10.1137/S0036144598347059

[27] Hirt, C. W., and Nichols, B. D., "Volume of Fluid (VOF) Method for the Dynamics of Free Boundaries," *Journal of Computational Physics*, Vol. 39, No. 1, 1981, pp. 201–225. doi:10.1016/0021-9991(81)90145-5

[28] Sussman, M., Smereka, P., and Osher, S., "A Level Set Approach for Computing Solutions to Incompressible Two-Phase Flow," *Journal of Computational Physics*, Vol. 114, No. 1, 1994, pp. 146–159. doi:10.1006/jcph.1994.1155

[29] Rudman, M., "Volume-Tracking Methods for Interfacial Flow Calculations," *International Journal for Numerical Methods in Fluids*, Vol. 24, No. 7, 1997, pp. 671–691. doi:10.1002/(ISSN)1097-0363

[30] Mittal, R., and Iaccarino, G., "Immersed Boundary Methods," *Annual Review of Fluid Mechanics*, Vol. 37, No. 1, 2005, pp. 239–261. doi:10.1146/annurev.fluid.37.061903.175743

[31] Fedkiw, R. P., Aslam, T., Merriman, B., and Osher, S., "A Non-Oscillatory Eulerian Approach to Interfaces in Multimaterial Flows (the Ghost Fluid Method)," *Journal of Computational Physics*, Vol. 152, No. 2, 1999, pp. 457–492. doi:10.1006/jcph.1999.6236

[32] Rauschenberger, P., Criscione, A., Eisenschmidt, K., Kintea, D., Jakirć, S., Tuković, Ž., Roisman, I., Weigand, B., and Tropea, C., "Comparative Assessment of Volume-of-Fluid and Level-Set Methods by Relevance to Dendritic Ice Growth in Supercooled Water," *Computers and Fluids*, Vol. 79, June 2013, pp. 44–52. doi:10.1016/j.compfluid.2013.03.010

[33] Juric, D., and Tryggvason, G., "A Front-Tracking Method for Dendritic Solidification," *Journal of Computational Physics*, Vol. 123, No. 1, 1996, pp. 127–148. doi:10.1006/jcph.1996.0011

[34] Tryggvason, G., Bunner, B., Esmaeeli, A., Juric, D., Al-Rawahi, N., Tauber, W., Han, J., Nas, S., and Jan, Y., "A Front-Tracking Method for the Computation of Multiphase Flow," *Journal of Computational Physics*, Vol. 169, No. 2, 2001, pp. 708–759. doi:10.1006/jcph.2001.6726

[35] Glimm, J., Grove, J., Li, X., and Tan, D., "Robust Computational Qlgorithms for Dynamic Interface Tracking in Three Dimensions," *SIAM Journal on Scientific Computing*, Vol. 21, No. 6, 2000, pp. 2240–2256. doi:10.1137/S1064827598340500

[36] Enright, D., Fedkiw, R., Ferziger, J., and Mitchell, I., "A Hybrid Particle Level-Set Method for Improved Interface Capturing," *Journal of Computational Physics*, Vol. 183, No. 1, 2002, pp. 83–116. doi:10.1006/jcph.2002.7166

[37] Du, J., Fix, B., Glimm, J., Jia, X., Li, X., Li, Y., and Wu, L., "A Simple Package for Front Tracking," *Journal of Computational Physics*, Vol. 213, No. 2, 2006, pp. 613–628. doi:10.1016/j.jcp.2005.08.034

[38] Loehner, R., and Yang, C., "Improved ALE Mesh Velocities for Moving Bodies," *Communications in Numerical Methods in Engineering*, Vol. 12, No. 10, 1996, pp. 599–608. doi:10.1002/(SICI)1099-0887(199610)12:10<>1.0.CO;2-R

[39] Stein, K., Tezduyar, T. E., and Benney, R., "Mesh Moving Techniques for Fluid-Structure Interactions with Large Displacements," *Transactions of the ASME*, Vol. 70, No. 1, 2003, pp. 58–63. doi:10.1115/1.1530635

[40] Stein, K., Tezduyar, T. E., and Benney, R., "Automatic Mesh Update with the Solid-Extension Mesh Moving Technique," *Computer Methods in Applied Mechanics and Engineering*, Vol. 193, Nos. 21–22, 2004, pp. 2019–2032. doi:10.1016/j.cma.2003.12.046

[41] Farhat, C., Degand, C., Koobus, B., and Lesoinne, M., "Torsional Springs for Two-Dimensional Dynamic Unstructured Fluid Meshes," *Computer Methods in Applied Mechanics and Engineering*, Vol. 163, Nos. 1–4, 1998, pp. 231–245. doi:10.1016/S0045-7825(98)00016-4

[42] Degand, C., and Farhat, C., "A Three-Dimensional Torsional Spring Analogy Method for Unstructured Dynamic Meshes," *Computers and Structures*, Vol. 80, Nos. 3–4, 2002, pp. 305–316. doi:10.1016/S0045-7949(02)00002-0

[43] Helenbrook, B. T., "Mesh Deformation Using the Biharmonic Operator," *International Journal for Numerical Methods in Engineering*, Vol. 56, No. 7, 2003, pp. 1007–1021. doi:10.1002/(ISSN)1097-0207

[44] de Boer, A., van der Schoot, M., and Bijl, H., "Mesh Deformation Based on Radial Basis Function Interpolation," *Computers and Structures*, Vol. 85, Nos. 11–14, 2007, pp. 784–795. doi:10.1016/j.compstruc.2007.01.013

[45] Rendall, T., and Allen, C., "Efficient Mesh Motion Using Radial Basis Functions with Data Reduction Algorithms," *Journal of Computational Physics*, Vol. 228, No. 17, 2009, pp. 6231–6249. doi:10.1016/j.jcp.2009.05.013

[46] Witteveen, J. A., and Bijl, H., "Explicit Mesh Deformation Using Inverse Distance Weighting Interpolation," *19th AIAA Computational Fluid Dynamics Conference*, AIAA Paper 2009-3996, June 2009.

[47] de Boer, A., van der Schoot, M., and Bijl, H., "Mesh Deformation Based on Radial Basis Function Interpolation," *Computers and Structures*, Vol. 85, Nos. 11–14, 2007, pp. 784–795. doi:10.1016/j.compstruc.2007.01.013

[48] Menter, F., "Two-Equation Eddy-Viscosity Turbulence Models for Engineering Applications," *AIAA Journal*, Vol. 32, No. 8, 1994, pp. 1598–1605. doi:10.2514/3.12149

[49] Jiao, X., Colombi, A., Ni, X., and Hart, J. C., "Anisotropic Mesh Adaptation for Evolving Triangulated Surfaces," *Engineering with Computers*, Vol. 26, No. 4, Aug. 2010, pp. 363–376.

[50] Karman, S., "Adaptive Optimization-Based Smoothing for Tetrahedral Meshes," AIAA Paper 2015-2038, June 2015.