Discrete Surface Evolution and Mesh Deformation for Aircraft Icing Applications

David Thompson*, Xiaoling Tong†, Qiuhan Arnoldus‡, Eric Collins§,
David McLaurin¶, and Edward Luke‖
Mississippi State University, Mississippi State, MS, 39762, USA
Colin Bidwell**
NASA Glenn Research Center, Cleveland, OH, 44315, USA

Robust, automated mesh generation for problems with deforming geometries, such as ice accreting on aerodynamic surfaces, remains a challenging problem. Here we describe a technique to deform a discrete surface as it evolves due to the accretion of ice. The surface evolution algorithm is based on a smoothed, face-offseting approach. We also describe a fast algebraic technique to propagate the computed surface deformations into the surrounding volume mesh while maintaining geometric mesh quality. Preliminary results presented here demonstrate the efficacy of the approach for a sphere with a prescribed accretion rate, a rime ice accretion, and a more complex glaze ice accretion.

I. Introduction

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 mesh must evolve in response to the growth of the ice shape. Assuming a loosely coupled ice accretion strategy, such as that used in *LEWICE3D*,[1] a sequence of quasi-static accretion steps is performed to generate the final ice shape. Since the ice shape changes, each accretion step requires a new mesh. However, a full mesh regeneration may be expensive for complex configurations. An alternative strategy is to deform the mesh in response to the ice growth. The second challenge is that accreted ice shapes can become quite complex. While the current state of the art in ice accretion prediction does not produce shapes with exceedingly complicated geometries, the predicted ice shapes can nevertheless present significant challenges for meshing software. Additionally, as the fidelity of ice accretion prediction increases, the complexity of the numerically-generated ice shapes will increase. Any mesh generation strategy will necessarily require the ability to handle such complex geometries if it is to represent a viable, long-term solution.

Currently, there is no automated mesh generation process designed to work with *LEWICE3D*. The resulting capability gap precludes routine grid-based, multi-time-step simulations of ice accretion on complex configurations. As part of the NASA Atmospheric Environment Safety Technology Project, an ongoing effort at Mississippi State University seeks to facilitate routine simulation of ice accretion on realistic, three-dimensional configurations by developing a suite of meshing tools that will produce unstructured, mixed element (hybrid) meshes for evolving ice shapes in an automatic, efficient, and robust manner. Such automated mesh generation is a necessary step in the enhancement of existing ice accretion prediction tools as well as in the development of the next generation of these tools.

*Associate Professor, Department of Aerospace Engineering, PO Box A, Associate Fellow.
†Assistant Research Professor, Center for Advanced Vehicular Systems, PO Box 5405, Associate Member.
‡Research Associate II, Center for Advanced Vehicular Systems, PO Box 5405, Nonmember.
§Postdoctoral Associate, Center for Advanced Vehicular Systems, PO Box 5405, Member.
¶Assistant Research Professor, Center for Advanced Vehicular Systems, PO Box 5405, Member.
‖Associate Professor, Department of Computer Science and Engineering, PO Box 9637, Senior Member.
**Aerospace Engineer, Icing Branch, 21000 Brookpark Road, Member.

Any approach that has the potential to make grid-based ice accretion simulations for realistic configurations commonplace occurrences must have the following characteristics:

- Automation: Simulating the evolving ice shape necessarily requires generating a new mesh for each ice shape. For this approach to be routine, it is necessary that the mesh generation process be as automated as possible. Once an initial mesh is generated, the user should be removed from the loop, even in cases where the surface evolution initially produces an invalid mesh or a mesh of poor quality due to the complexity of the ice shape.
- Efficiency: Although a new mesh must be generated for each ice shape, the process must be efficient. The simplest approach, completely regenerating the mesh, is potentially a time-consuming task for complex aircraft configurations and not appropriate for ice accretion simulations.
- 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 mesh retains sufficient quality as the ice surface evolves.

In this paper, we briefly describe the algorithms we employ to evolve the discrete surface mesh that represents the accreting ice and to project these deformations into the volume mesh. We are developing a solver-neutral interface propagation tool that computes the position of a discrete surface as it evolves under an accretion rate map specified by *LEWICE3D*. The surface evolution 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. The output from *iceSurf* provides input, in the form of a surface displacement file, to the mesh deformation tool *gridMover* to produce a new volume mesh. We include preliminary results for a sphere with a prescribed accretion rate field, a relatively simple rime ice accretion, and a more complex glaze ice accretion.

II. Background: Surface Evolution

One of the challenges associated with evolving a faceted, discrete surface 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 centroid in the direction of the face normal as shown for a two-dimensional surface in figure 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 these offset planes (not parallel), intersect in a line while three non-parallel 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 share a given node in a typical triangular surface mesh is usually more than three and, consequently, the node determination is overspecified. This results in an ambiguity in how the nodal positions are defined in the new surface.

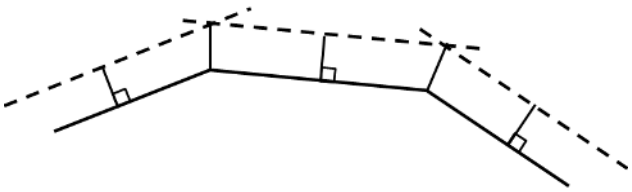


Figure 1. Face offsetting produces unambiguous nodal positions in two dimensions.

One approach that has shown promise for evolving a surface mesh while conserving volume is the method developed by Jiao.[2,3] Jiao employs a singular value decomposition (SVD) to solve a least square problem and then applies an eigenvalue/eigenvector analysis at each node to resolve its normal motion, which generates the surface geometry, and its tangential motion, which maintains mesh quality.

The first step is to propagate each evolving face in its normal direction. 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 $\mathbf{p}$ with unit normal $\mathbf{n}$ can be expressed by a linear equation $\mathbf{n^T x} = \delta$, where $\delta = \mathbf{n^T p}$. If there are m faces passing through a node, an $m \times 3$ linear system will be formed

$$\mathbf{Nx} = \mathbf{a}. \tag{1}$$

Here, each row of the system corresponds to one of the m faces that are incident on the node and elements of $\mathbf{a}$ are the offset distances for each incident face. The linear system given by Eq. 1 can be under-determined or over-determined 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:

$$\mathbf{Ax} = \mathbf{b} \tag{2}$$

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

Since the matrix $\mathbf{A}$ in Eq. 2 is symmetric and positive semi-definite, it has an eigenvalue decomposition $\mathbf{A} = \mathbf{V \Lambda V^T}$, where $\mathbf{\Lambda}$ is the diagonal matrix consisting of the eigenvalues of $\mathbf{A}$, which are real and non-negative, and the corresponding eigenvectors are the columns of $\mathbf{V}$. Since $\mathbf{A} = \mathbf{N^T W N}$, the following singular value decomposition can be derived

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

where $\mathbf{U}$ is a $m \times 3$ matrix.

The vector space spanned by the eigenvectors corresponding to the larger eigenvalues of $\mathbf{A}$ 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 Equation 2 represents an advective motion in which the resulting surface is the intersection of the propagated face planes. For wavefront 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 wavefront 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 is computed, Jiao improves mesh quality by performing a null space smoothing by computing a tangential motion $\mathbf{t}$ at each vertex $\mathbf{v}$. $\mathbf{t}$ is a weighted average of the neighborhood of $\mathbf{v}$ projected onto the null space. This smoothing scheme has been shown to preserve sharp features and to introduce only very small volume errors. Jiao suggests repeating this step without displacement in the primary space to incorporate a global smoothing into the algorithm that preserves the accreted volume.

III. Approach

Generating an ice shape for a specified accretion time t_{ice} is accomplished by performing a series of quasi-static, loosely-coupled, ice accretion/flow simulation steps. This approach is necessary because, as the ice shape evolves, it changes the flow field, which, in turn impacts the local ice accretion rate. We term each of these quasi-static steps an "ice accretion step" with an associated time interval Δt. As noted below, each time interval Δt may be further subdivided into subintervals Δt_s. For each ice accretion step: (1) a CFD simulation is performed to compute the flow field about the current ice shape, (2) a *LEWICE3D* computation is performed to determine the new ice shape, or alternatively, the accretion rate map, and (3) the surface and volume meshes are evolved based on the ice accretion rate using *iceSurf* and *gridMover*, respectively. Here, we focus on the the process employed to compute the deformation of the computational mesh in response to the ice accretion, which can be divided into three distinct phases:

- Generate a volume accretion rate map on the wetted surface
- Evolve the surface mesh based on the accretion rate map using *iceSurf*
- Deform the volume mesh by projecting the surface deformations into the volume mesh using *gridMover*

Each of these processes is described in the sections that follow.

III.A. Generate Accretion Rate Map using Lofting

Currently, *LEWICE3D* does not provide an accretion rate map. *LEWICE3D* generates ice shapes using the strip-based strategy employed in *LEWICE2D*,[4] which is based on the Messinger icing model.[5] In future generations of *LEWICE3D*, a fully three-dimensional approach will be employed to generate an ice accretion rate map that will be used by the surface mesh evolution algorithm. Two different strategies were employed to circumvent this shortcoming.

The first approach, which is applicable only for cold, rime icing accretions, uses the collection efficiency to estimate a surface icing rate. The underlying assumption employed in this approach is that the droplets freeze on impact producing a pure deposition problem. The icing rate for each surface element is calculated assuming that no evaporation or runback occurred and is given by

$$\frac{dv_{ice}}{dt} = \frac{V_\infty \times \beta \times LWC \times A}{\rho_{ice}} \tag{4}$$

where V_∞ is the freestream airspeed, β is the local collection efficiency, LWC is the free stream liquid water context, A is the area of the surface element under consideration, and ρ_{ice} is the ice density.

The second approach, which is applicable for warmer, glaze icing conditions, uses a lofting method to generate the icing rate map. The lofting algorithm assumes that the ice thickness varies linearly along spanwise lines for wings and circumferentially for bodies of revolution (e.g. inlets, spinners and radomes). Lofting information is used to generate transformations that facilitate interpolation of ice thickness from the strip-based ice accretions to the surface. This method is flexible and robust and allows interpolation on wings with taper, twist, and leading edge curvature and bodies-of-revolution.

The ice patch lofting scheme uses the ice thickness values for the surface nodes, which are interpolated from the *LEWICE3D* ice shape values, along with the surface normal at the nodes to generate the new iced surface. Volume elements are formed from the original surface element and the new displaced iced surface element. The icing rate is then determined by calculating the volume of these iced volume elements and then dividing this volume by the icing time. The use of ice thickness interpolated from the *LEWICE3D* ice shapes allows a convenient, robust method for generating three-dimensional iced surfaces which have run-back and evaporation effects.

Two types of lofts are available to describe various surfaces of interest. The first lofting type is a wing-type lofting. This lofting requires the input of the leading edge and trailing edge of the wing. The second lofting type is a body of revolution-type lofting. The body of revolution lofting requires the line of rotation and the leading edge center of rotation. A two-dimensional coordinate system (S,T) is employed for the loftings for which S is the axial coordinate and T is either the spanwise coordinate for wing-type lofts or the circumferential angle for body of revolution-type lofts (figure 2).

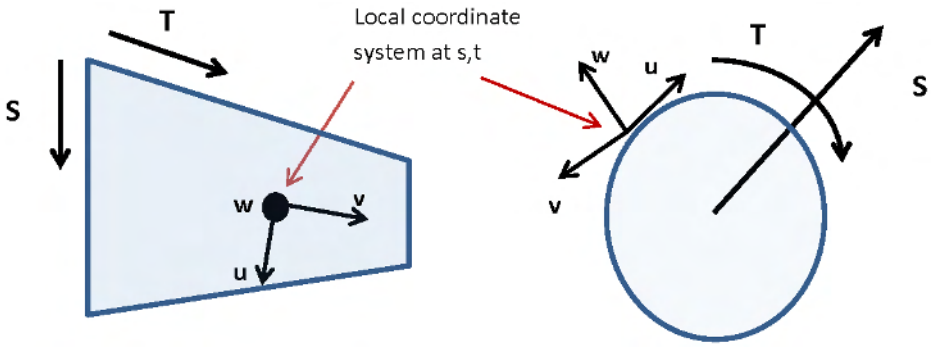


Figure 2. ***LEWICE3D*** **planform types and their associated local coordinate systems – wing(left) and body of revolution (right).**

The position (s,t), and the local coordinate system (u,v,w) at a position (x,y,z) on the lofting are generated using an iterative process. The up-vector w is the planform normal vector at (S,T). For wing-type lofts, the planform normal is generated by taking the cross product of the leading edge direction vector at T and a vector formed from a line at T between the leading edge and trailing edge lines. For the body of

rotation type geometries the planform normal is the radial vector at T. The spanwise vector, v is generated by interpolation from the leading edge and trailing edge lines at (S,T) for wing-type planforms. For body of rotation type planforms the spanwise vector is the tangential vector at T. The axial vector, u is generated by taking the cross product of the up-vector and the spanwise vector (figure 2). The local coordinates systems generated for the surface point and icing cut points are used to transform the local surface points into the local ice cut point coordinate system for the interpolation of the ice thickness (figure 3).

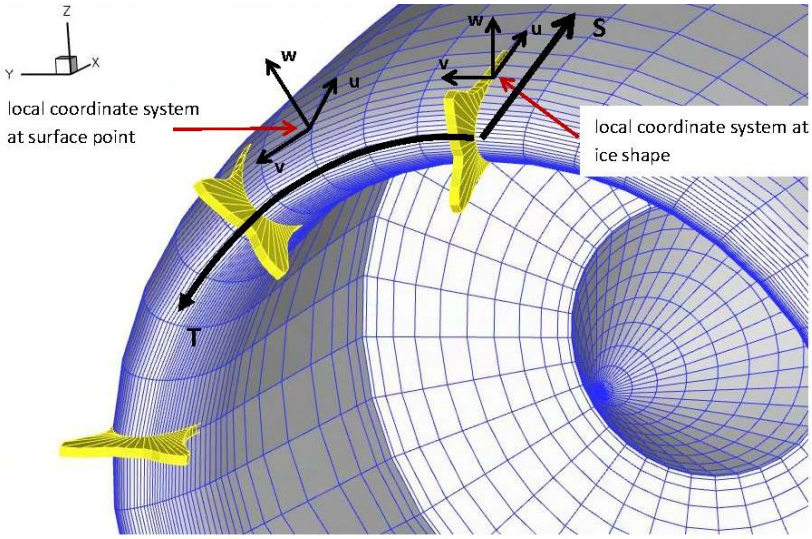


Figure 3. Local coordinate systems for surface point and ice shape.

Two lines are formed in the local ice cut point system (figure 4). One line segment connects neighboring ice cut points ($l1$). The other line is formed using the local ice cut spanwise vector as the slope and the transformed surface point as the intercept ($l2$). A set of tests is performed to determine if the minimum distance between the $l1$ and $l2$ occurs within the endpoints of $l1$ and if this minimum distance is reasonably small value. If both tests are positive the ice thickness is interpolated linearly from the two ice cut thickness values at the ice cut endpoints. The local coordinates systems generated for the surface point and icing cut points are used to transform the local surface points into the local ice cut point coordinate system for the interpolation of ice thickness (figure 3).

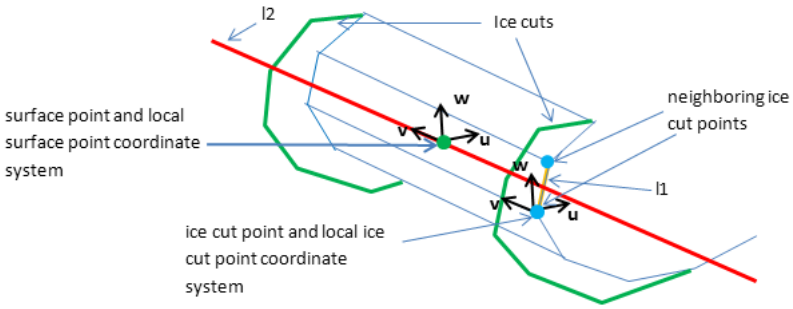


Figure 4. Ice thickness interpolation scheme for surface point.

This procedure is repeated for all of the ice cuts associated with the local surface point. If more than one intercept is found for the local surface point then a linear interpolation on T is performed from the two surrounding intercepts (larger T and smaller T than the surface point T). If the value of T of the surface point is either greater than the T's of all of the ice cut intercepts or is less than the T's of all of the ice cut intercepts the value of ice thickness is set to zero. If an intercept has been found and only one cut has been associated with the local surface point then an extrapolation is assumed and the ice thickness at the local

surface point is set to the interpolated value of ice thickness. The above procedure is repeated for all surface points and all ice cuts associated with the surface point.

The new iced surface is generated using the ice thickness and surface normal information at each node (figure 5). The iced surface nodes are determined by adding the ice thickness in the surface normal direction to the clean surface nodes. The ice accretion rate for each surface element is calculated from the volume of ice formed at each surface element divided by the icing time.

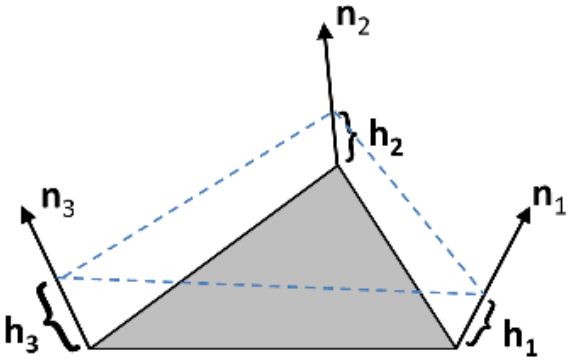


Figure 5. Ice addition scheme for surface element.

III.B. Surface Mesh Evolution

We have developed a computational tool (*iceSurf*) that employs elements of Jiao's algorithm[2,3] to evolve a discrete surface mesh in response to the ice accretion rate. Like Jiao's algorithm, the algorithm used in *iceSurf* emphasizes conservation of the accreted volume. However, unlike Jiao's algorithm, which utilizes a face velocity, *iceSurf* utilizes the volume growth rate to drive the surface evolution. *iceSurf* uses the offset direction in the primary space defined by Jiao's method as the initial nodal displacement direction and then employs global and local smoothing algorithms to maintain mesh quality.

III.B.1. Identify Geometrical Features

A procedure similar to that described by Jiao[2] is employed to identify geometrical features such as edges and corners. *iceSurf* provides special treatment for the nodes associated with these features. If the angle between the two faces that share a given edge is greater than a threshold, then this edge is considered to be a geometrical edge. If a node has two associated edges, the node is marked as an edge node. If a node has three or more associated edges, it is marked as a corner node.

III.B.2. Define Nodal Offset Direction

The next step in the process is to generate an initial nodal offset direction. To determine the displacement direction for a node, *iceSurf* computes the direction of the displacement using Jiao's primary space. However, there are two significant differences. First, as noted above, Jiao uses a face velocity while we use a volume accretion rate. This impacts Jiao's algorithm because the face offset distance, i.e., $\mathbf{b}$, in Equation 1, is unknown. The approach employed here is to assume a uniform displacement in the determination of the offset direction

$$\mathbf{N}\mathbf{x} = \mathbf{1} \tag{5}$$

where $\mathbf{1}$ is the $m \times 1$ vector that has elements that are unity. This temporarily circumvents the need for knowledge of the face displacement. An additional change is that the weight matrix $\mathbf{W}$ in Jiao's algorithm (see Equation 3) is based on the face areas of the triangles incident on the node under consideration while our weight matrix is based on the included angles of the faces incident to the node.

Once the primary direction is defined, it is held fixed throughout the remainder of the process.

III.B.3. Determine Height Field

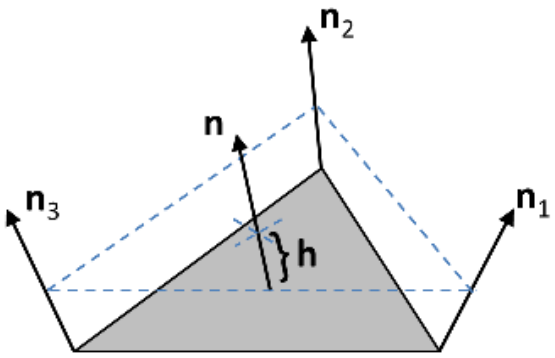


Figure 6. Accreted volume based on height field.

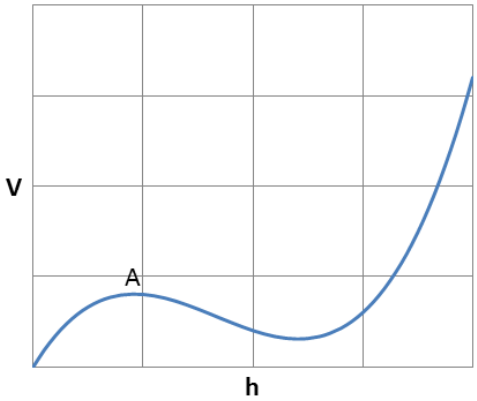


Figure 7. Volume of accreted ice as a function of height.

It is now necessary to determine the height field that gives the appropriate volume accreted on each face. From this height field, the nodal displacements can be determined. As illustrated in Figure 6, this computation is complicated by the fact that the nodal normals are not parallel to the face normal except in special circumstances. The accreted volume is a nonlinear function of height h given by

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

where a, b, and c are functions of the geometry of the face and the nodal "normals" $\mathbf{n_1}$, $\mathbf{n_2}$, and $\mathbf{n_3}$. In general, this cubic function may not be monotonic (see Figure 7). The first maximum value of volume, indicated by point A, is the volume at which the faces cross and the volume begins to decrease (due to the addition of negative volume). The time at which this occurs can be obtained by dividing the maximum volume by the accretion rate and represents the maximum time step $\Delta t_{V_{max}}$ that can be taken for a given set of conditions. Jiao[3] describes a procedure to compute a scale factor $0 < \alpha < 1$ for the time step to prevent self intersections. The maximum allowable time step, which we denote as Δt_{max} is then given by

$$\Delta t_{max} = \alpha \min\left(\Delta t, \Delta t_{V_{max}}\right). \tag{7}$$

If the resulting time increment Δt_{max} is less than the desired time increment for the ice accretion step Δt, multiple steps are required for the time step. These steps are referred to as subinterval steps with a time step of $\Delta t_s = \Delta t_{max}$.

III.B.4. Smooth the Height Field

In general, the heights for two triangular faces that share an edge will not be equal. A height field smoothing that conserves volume is then employed to redistribute the volume. Assume that we have 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 $\Delta V = (h_1 - h_2) A_1$ where A_1 is the area of T_1 at the height $(h_1 + h_2)/2$. The volumes associated with the two faces are then modified using

$$\begin{aligned} V_1 &= V_1 - \beta \Delta V \\ V_2 &= V_2 + \beta \Delta V \end{aligned} \tag{8}$$

where $0 < \beta < 1/2$. We have found that $\beta = 0.2$ produces good results. 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, 10-15 height field smoothing iterations are employed. The resulting accreted volume is then compared to the value obtained by multiplying the accretion rate by the time increment Δt_s. The resulting volume residue is converted to a rate and then added or subtracted during the next subinterval step as appropriate.

III.B.5. Compute Nodal Positions

The next step is to determine the nodal positions using the smoothed height field. As discussed by Jiao,[3] the nodal position should be the intersection point of the offset planes for advective motion. In contrast, for wavefront motion, the node should reside on a smooth nonlinear patch. To simplify the computation, it is assumed that the direction of the wavefront nodal displacement is the same as the advective displacement, and only the length of displacement is adjusted so that the node will be on the nonlinear patch. Referring to Figure 1, recall that the height field represents the offset distance in the direction of the face normal so it is not the equivalent to the nodal displacement. If h is the offset for the face and the face is contracting at this node, the nodal displacement h_n is given by

$$h_n = \frac{h}{\mathbf{n} \cdot \mathbf{n_n}} \tag{9}$$

where $\mathbf{n}$ and $\mathbf{n_n}$ are the face and node normals, respectively. If the face is diverging at this node, $h = h_n$. Of course, there are multiple faces associated with this node. The final nodal displacement is given by a weighted average of the displacement from the faces incident on the node. The weighting is based on the included angles of the faces at that node.

III.B.6. Smooth the Evolved Surface Mesh

The nodal positions are then smoothed using the null space smoothing described by Jiao.[2,3] According to Jiao, the null space "... is a plane, a line, or the empty set, tangential to the surface at the vertex." Null space smoothing moves nodes in the tangent plane or in the direction of minimum curvature of the surface and therefore, tends to preserve the volume better than other forms of smoothing, such as Laplacian. It also tends to preserve sharp features or regions of large curvature. Unless noted otherweise, null space smoothing was performed for all of the results presented here.

III.C. 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*. *gridMover* is based on the method developed by Luke et al.[6] to perform a volume mesh deformation in response to a surface mesh deformation. Their approach takes boundary displacements as input and returns a deformed volume mesh. A robust direct interpolation, which is based on an inverse distance weighted (IDW) technique, is 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 $\mathbf{s}_i(\mathbf{r})$ that is computed using

$$\mathbf{s}(\mathbf{r}) = \mathbf{M}_i\mathbf{r} + \mathbf{b}_i - \mathbf{r}, \tag{10}$$

where $\mathbf{M}_i$ is the rotation matrix, $\mathbf{b}_i$ is the displacement vector associated with the i^{th} node, and $\mathbf{r}$ 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:

$$\mathbf{s}(\mathbf{r}) = \frac{\Sigma w_i(\mathbf{r})\,\mathbf{s}_i(\mathbf{r})}{\Sigma w_i(\mathbf{r})} \tag{11}$$

We chose a function of the reciprocal of distance for the weight function $w_i(\mathbf{r})$. A two-exponent weight function is employed so that we can preserve near-boundary deformations, which is critical for 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 Equation 11 in $n \log n$ time. This accelerated IDW approach has been shown to be competitive with the considerably more expensive radial basis function (RBF) proposed by deBoer *et al.*[7] In fact, the IDW approach does a better job preserving

the orthogonality of the mesh in the near-wall viscous regions, which makes it an ideal candidate for ice accretion prediction. More details are given in Luke et al.[6]

IV. Results

We now present results that demonstrate the efficacy of the mesh evolution and deformation algorithms. First, we consider a prescribed ice accretion on a sphere. The intent is to illustrate salient features of the algorithms. We then employ the algorithms in a loosely coupled approach utlizing *Loci/Chem* and *LEWICE3D* for a rime icing condition and a more challenging glaze icing condition.

IV.A. Test Cases: Sphere with Prescribed Accretion Map

To illustrate the effectiveness of the surface evolution algorithm employed in *iceSurf*, the volume accretion on a sphere of radius 1 m was computed for an analytically-specified accretion rate map in which faces that were located at $x > 0$ were given a uniform accretion velocity, $u_a = 0.1$ m/s. A face volume accretion rate was then computed by multiplying the face area by the accretion velocity. The evolving surface in time can be computed analytically as a sphere with a radius offset of $h = \sqrt[3]{3 * t * u_a + 1} - 1$. For this test case we computed the accretion at a time of 2 s, which corresponded to an increase in the radius of the sphere by $h = 0.16961$ m. This was performed by *iceSurf* using six substeps to produce the surface shown in figure 8. The resulting accretion closely matched the expected analytical results. No smoothing was used for this case. The surface evolution algorithm employed in *iceSurf* produced a valid surface mesh even for a discontinuous rate map and no smoothing.

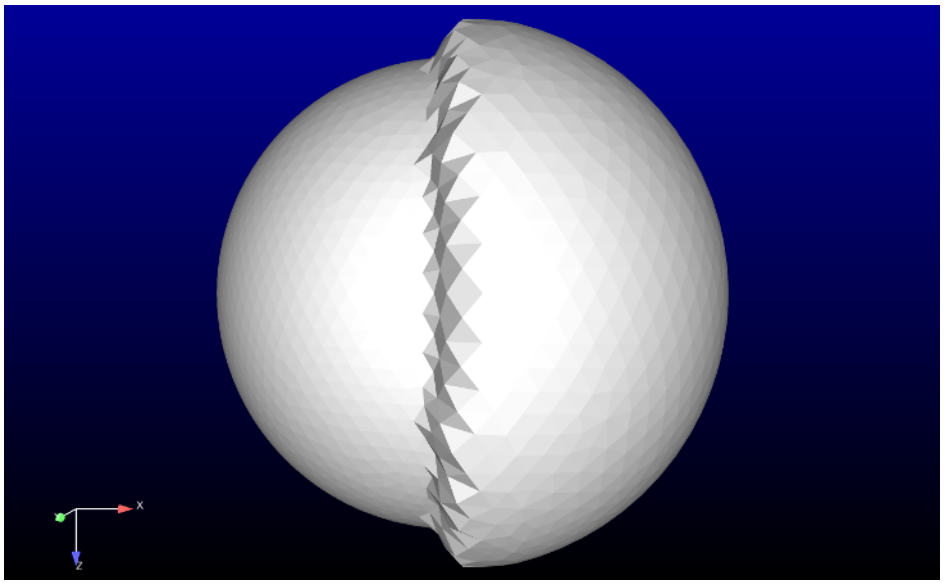
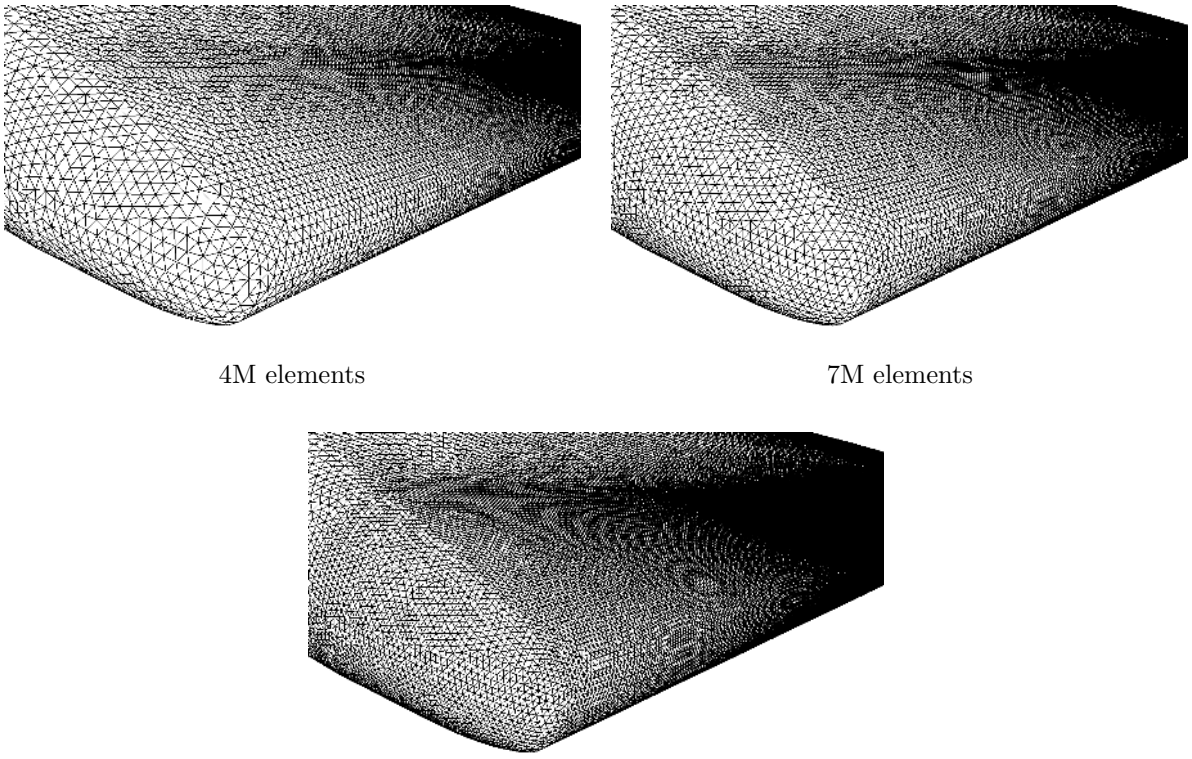


Figure 8. Sphere with specified discontinuous accretion rate map.

IV.B. Rectangular Planform Wing with GLC305 Cross Section

Loosely-coupled *Loci/CHEM-LEWICE3D* simulations of ice accretion on a rectangular planform wing with a constant GLC305 airfoil section were performed and the results compared with available *LEWICE2D* and *LEWICE3D* simulation results and experimental data. The chord length of the wing was 0.9144 m and the span was 1.8288 m. The flow field was computed using the *Loci/CHEM* flow solver.[8] The flight conditions and icing conditions were chosen to match cases documented in the *LEWICE2D* validation report.[9] In this study, the freestream velocity and pressure were 90 m/s and 1 atm, respectively. The angle of attack was 4.5 deg. Two different icing conditions were selected with different air temperatures producing a rime accretion and a glaze accretion. A far field boundary condition was imposed approximately ten chords in front 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,[10] which is a blend of $k-\omega$ and $k-\epsilon$ models, was adopted to model the effects of turbulence in the RANS simulation. A high Reynolds number mixed element (hybrid) mesh was generated using the *SolidMesh* mesh generation tool.[11] In order to study the effects of different mesh resolution on the resulting ice shape, three meshes were generated with 4 million, 7 million and 12 million elements. Starting from the 4 million element mesh, the subsequent meshes represent a doubling of the number of triangular elements in a region of the surface mesh near the leading edge of the wing over the first 15 percent of the chord (see figure 9). We also investigated the effects of subdividing the icing time into multiple time steps on the resulting ice shape.



4M elements

7M elements

12M elements

Figure 9. Meshes employed for refinement study.

After the flow field was obtained, we used the tool *fluentensightcnv*, provided by NASA GRC as part of the *LEWICE3D* distribution, to convert *Ensight* output files generated by *Loci/CHEM* into a format that is compatible with the *LEWICE3D* ice accretion code. 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 wing span 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 Monte-Carlo collection efficiency method (IMNTCL=1) was utilized to compute the droplet collection efficiency.

As noted in Section III.B, *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 achieved can be dramatically affected by the number of substeps employed in the integration. Additionally, to achieve volume conservation, the actual volume swept out by each face during the integration is tracked. A volume conservative height smoothing can be employed at each substep of *iceSurf* to improve stability and surface smoothness. The effects of employing substeps and height smoothing in *iceSurf* will now be discussed. In the comparisons that follow, we note that only single-step ice accretions can be generated by

LEWICE3D.

IV.B.1. Results for Rime Ice Conditions

The icing conditions considered for this case were a liquid water content (LWC) of 0.405 gm/m^3, an ambient temperature and pressure of 257.88 K and 1 atm, respectively, and a relative humidity of 100 percent. At this temperature, the water droplets freeze on impact without runback, so the simplified method for estimating the icing rate based on collection efficiency described in Section III.A was appropriate. Water droplets with a diameter of 20 μm were released at a speed of 90 m/s on a plane approximately eight chord lengths upstream of the wing leading edge. Icing times of 2 and 4.4 min were chosen, which correspond to Case 210 and Case 211 in the *LEWICE2D* validation report,[9] respectively. All rime ice results were generated on the 12 million element mesh.

Figures 10 and 11 show the ice surfaces predicted by *iceSurf* and *LEWICE3D* compared with experimental data at spray times of 2 and 4.4 min, respectively. The black solid line in the figures denotes the clean wing surface. In both cases, a single accretion step was used with no smoothing. The simulated ice shapes agree with the experimental data fairly well. Once observed difference was that the numerical scheme does not predict the inflection on the upper surface of the accretion. To investigate the effects of employing multiple subintervals, figure 11 also shows results from a two-step ice accretion cycle. Note that the differences between the single-step prediction and two-step prediction are minimal. The failure to predict this inflection in the ice shape is currently under investigation. A second difference is the under prediction of icing limit on the lower surface. The simulation used only a single droplet size that represented the average droplet in the spray cloud. The inclusion of larger droplets, which would impact further aft, would have shifted the icing limit further aft. In general, rime cases are more sensitive to collection efficiency because there is excess convective heat transfer available and the drops freeze where they impact. Glaze shapes are less sensitive to collection efficiency because the local convective heat transfer controls the amount of ice at location.

These figures also show a comparison of ice shapes generated by *iceSurf* and *LEWICE3D* for two different icing times. The shapes suggest that the volume under the surface by *iceSurf* is less than *LEWICE3D*. The computed volume under the surface from *iceSurf* agrees with the provided volume rates better than 1%. We are currently investigating this discrepancy. It can can possibly be attributed to the simplistic model used to compute the accretion rate map for the rime cases. We have not observed a similar volume difference in computed glaze ice shapes. The three-dimensional ice shape at time of 4.4 min produced by *iceSurf* is shown in figure 12 where the extruded ice is shaded gray.

IV.B.2. Results for Glaze Icing Conditions

The icing conditions for this case corresponded to Case 072604 in the *LEWICE2D* validation report.[9] The conditions were identical to those employed for the rime ice cases (Cases 210 and 211 above) except that the ambient temperature was 263.2 K, which would allow runback to occur. Under these icing conditions, the accretion rate method based on the collection efficiency in not appropriate and the lofting method described in Section III.A was employed.

The first case considered was for a spray time of 6 min. A comparison of ice shapes generated by *iceSurf*, *LEWICE3D*, and *LEWICE2D* and experimental data is illustrated in figure 13. The results generated by *iceSurf* were obtained using one icing step and 20 steps of smoothing on the 12 million element mesh. The ice shapes produced by *iceSurf* and *LEWICE3D* compare very favorably, although the *iceSurf* results is smoother due to height smoothing. The three-dimensional ice shape produced by *iceSurf* is shown in figure 14.

We now discuss the ripples that appear in the surface as shown in figure 14. These ripples were smoothed to a great extent by application of the height smoothing method. The ripples apparently occurred at surface mesh irregularities and appeared to be caused by a discrepancy in surface normal computation between *iceSurf* and the *LEWICE3D*-lofting scheme. The relationship between the volume assigned to a surface element and the volume of the extruded prism was strongly dependent on computed surface normals. Differences in the normal computations between the two codes resulted in height irregularities that were associated with irregularities in the unstructured surface mesh. We are currently investigating solutions to this problem that may include standardizing the surface normal computation methods between the two algorithms. For the present simulations, these differences produced some unwanted irregularities that were

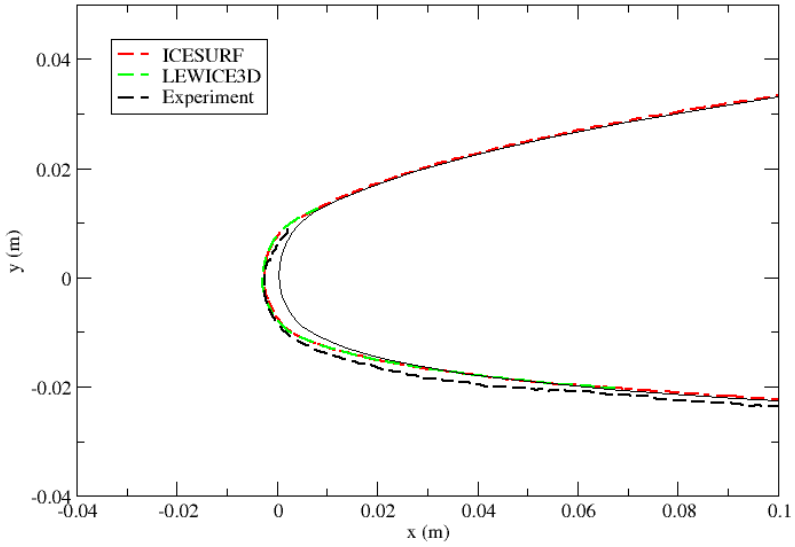


Figure 10. Comparison of computed ice shapes for t_{ice}=2 min with experimental data (black) for Case 210[9] – *iceSurf* (red) and *LEWICE3D* (green).

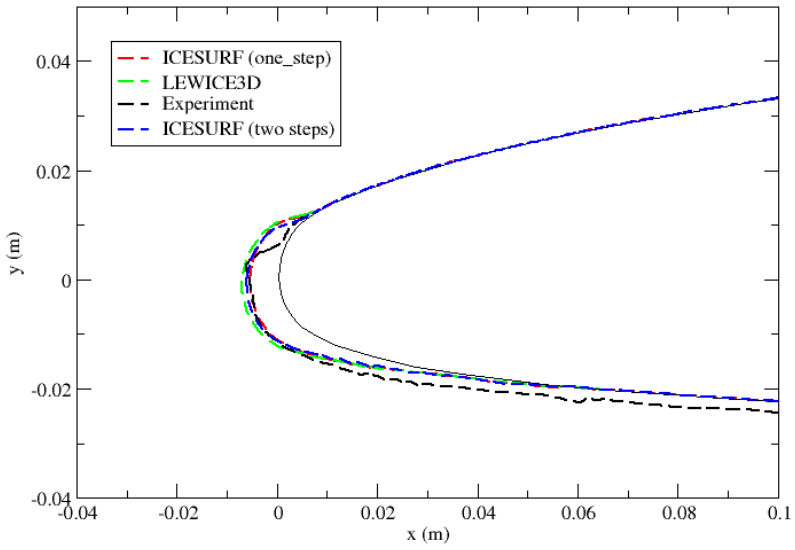


Figure 11. Comparison of computed ice shapes for t_{ice}=4.4 min with experimental data (black) for Case 211[9] – *iceSurf*-single step (red), *LEWICE3D* (green), and *iceSurf*-multi step (blue).

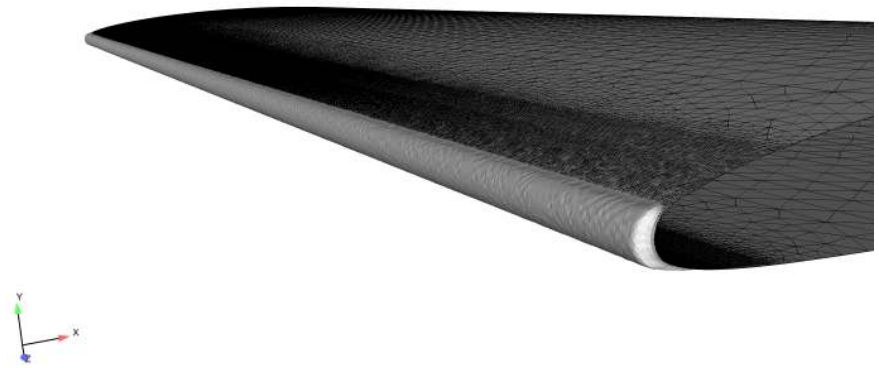


Figure 12. Three-dimensional ice shape for t_{ice}=4.4 min generated by *iceSurf* for Case 211.[9]

mitigated to some extent by utilizing a number of height smoothing steps; however, this may have introduced excessive smoothing.

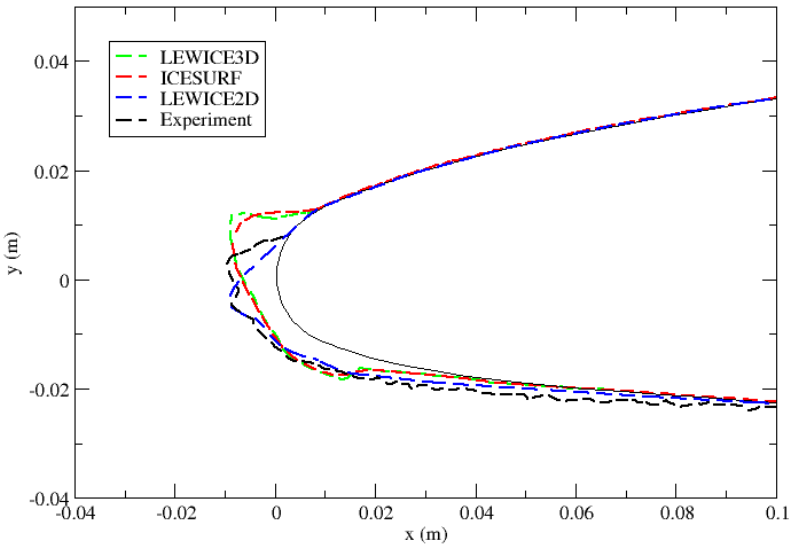


Figure 13. Comparison of the computed ice shapes for t_{ice}=6 min with experimental data (black) for Case 072604[9] – *LEWICE3D* (green), *iceSurf* (red), and *LEWICE2D* (green).

The flow field surrounding the airfoil changes in response to the ice accretion, which, in turn, produces a change in the ice accretion behavior, including the collection efficiency. Therefore, computing the ice shape with the initial ice accretion rate based on the clean wing is not accurate over long periods of time. To evaluate how the value of the time step affects the ice shape, we divided the 6-min icing time into three 2-min intervals. That is, the flow field and ice accretion rates were re-computed every 2 min based on newly deformed mesh obtained from *iceSurf*. Figure 15 shows the development of ice shape at 2-min intervals. Growth of ice on the leading edge combined with the change of the surrounding flow field would likely shift the stagnation point further aft, causing upper portion of the horn to move downward, assuming the same runback distance, as depicted in the figure. The difference between the ice shapes predicted using the one-

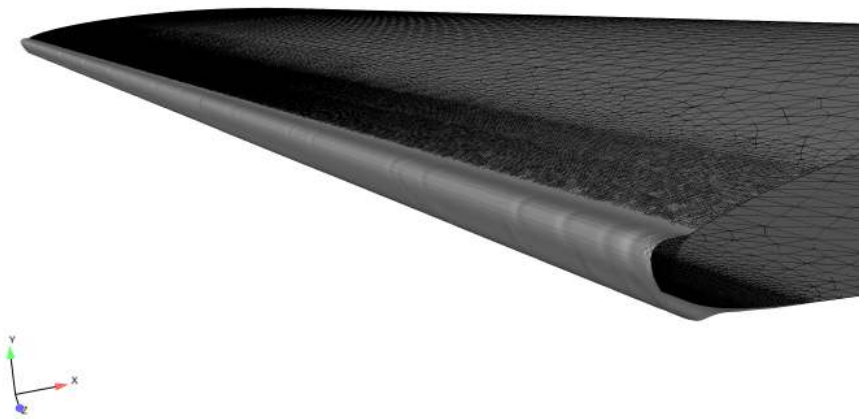


Figure 14. Three-dimensional ice shape for t_{ice}=6 min generated by *iceSurf* for Case 072604.[9]

step run and the three-step run is shown in figure 16. Again the downward shift of the upper portion of the ice in multi-step run is observed, and exhibits closer agreement with the experimental data. No attempt was made to reduce the subinterval time step below 2 min.

To study the effects of the height smoothing technique described in Section III.B, various numbers of smoothing steps were utilized during the computation of a single 6-min accretion step. The results are shown in figure 17. It is obvious that using more smoothing steps tend to "round" the ridge at the top of the horn. The choice of the appropriate number of smoothing steps is a balance between ensuring good quality of the deformed mesh and preserving the dominant features of the ice shape.

To examine the effects of mesh resolution on the computed ice shape, 6-min icing-time simulations were performed using three 2-min accretion steps for the three meshes shown in figure 9. As can be observed in figure 18, the results obtained using the 12 million and 7 million element meshes are very similar, while the ice shape obtained using the 4 million element mesh is more diffused.

Starting from the single-step, 6-min ice accretion, we intended to advance the ice shape for a total icing time of 22.5 min using multiple 2-min intervals, which would correspond to Case 072605 in the *LEWICE2D* validation report.[9] However, our current tool was unable to advance the surface to a time of 18 min due to self-intersection of the surface mesh. The surface generated by *iceSurf* at a time of 16 min, which is just before failure, is shown in figure 19 with a comparison to a *LEWICE2D* run. It can be seen that the classic horned glaze ice shape developed during this period.

The failure of the ice shape calculation at 18 min was caused by accumulated errors due to the ripples on the ice surface depicted in figure 20. Zooming into the region containing the ripples, we observe that the main reason for the failure is due to a folding of the surface on itself in a region where two "bumps" merge as shown in figure 21. While this could be resolved with topological editing of the surface (which is planned for the future for the *iceSurf* tool), the present tool is unable to continue surface evolution due to local self intersection. This issue will be resolved, at least for simpler ice shapes, once we correct problems related to the transfer of volume rates between *LEWICE3D* and *iceSurf*.

To illustrate the effectiveness of the volume mesh deformation tool *gridMover*, figure 22 shows "crinkle-cut" images of the mesh in a cutting plane located at z=0.9 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 maintains mesh quality up to the point that the calculation failed due to the self-intersection of the surface mesh. It should be noted that no attempt was made to tune the volume mesh deformation parameters.

V. Conclusion

In this paper, we describe a meshing strategy designed for simulating the accretion of ice on aerodynamic surfaces. Our approach is based on a discrete surface mesh evolution algorithm that employs an eigenvalue

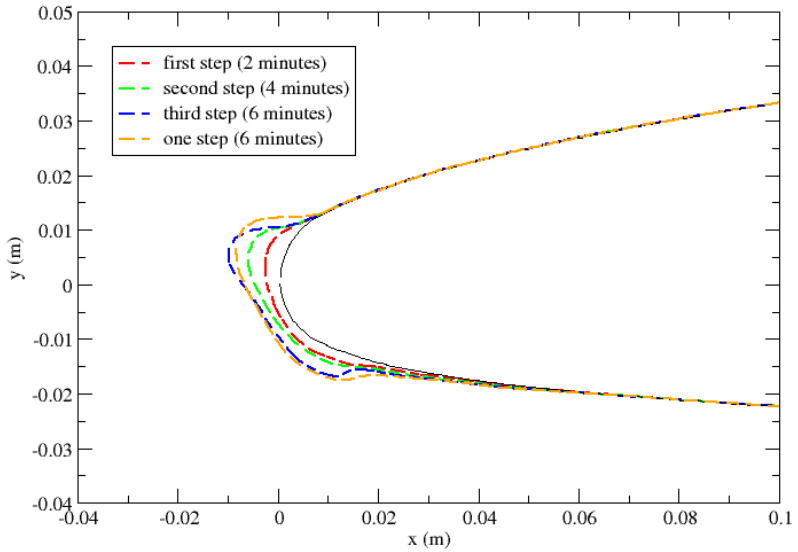


Figure 15. Comparison of ice shapes for t_{ice}=6 min generated by *iceSurf* for three 2-min intervals and a single 6-min interval for Case 072604.[9]

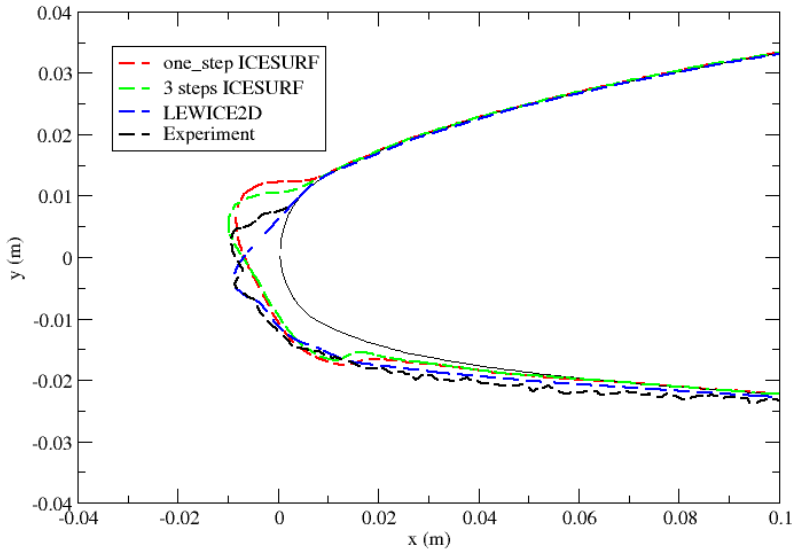


Figure 16. Comparison of ice shapes for t_{ice}=6 min generated by *iceSurf* for a multi-step run, a single-step run with *LEWICE2D* and experimental data for Case 072604.[9]

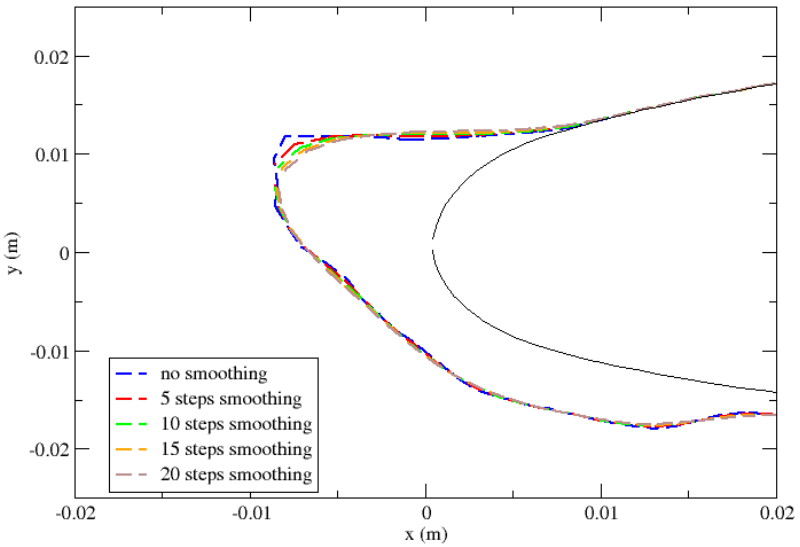


Figure 17. Comparison of ice shapes for t_{ice}=6 min generated by *iceSurf* for different numbers of height smoothing steps for Case 072604.[9]

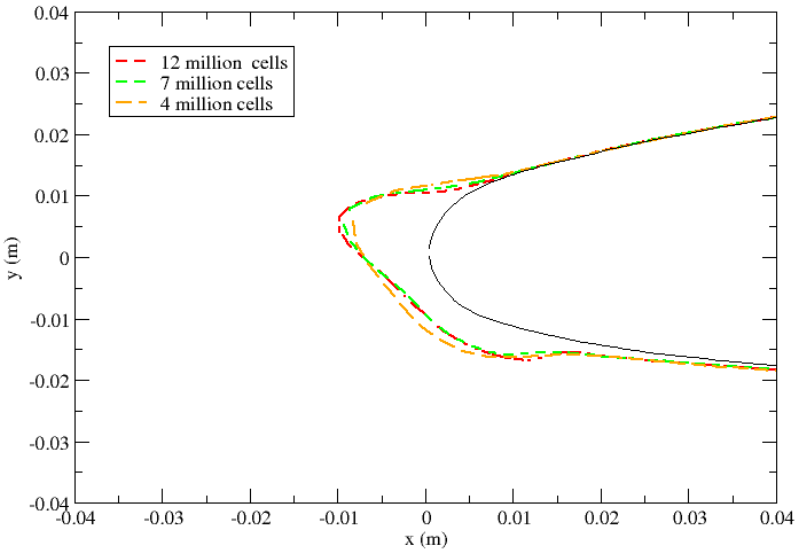


Figure 18. Comparison of ice shapes for t_{ice}=6 min generated by *iceSurf* for different mesh resolutions for Case 072604.[9]

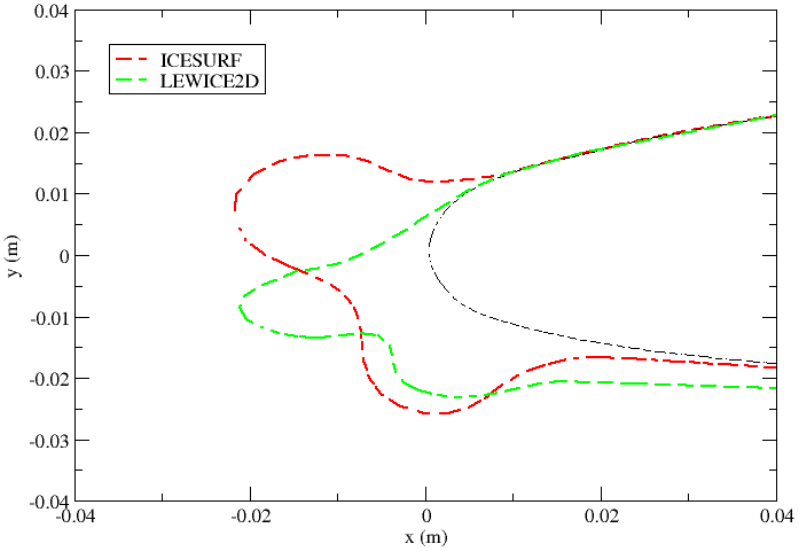


Figure 19. Comparison of ice shapes for t_{ice}=16 min generated by *iceSurf* and *LEWICE2D*.

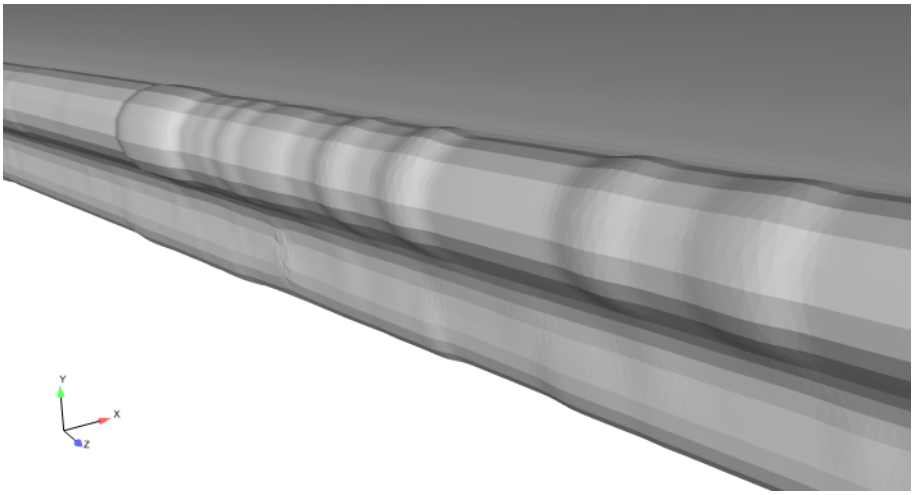


Figure 20. Ice surface for t_{ice}=16 min generated by *iceSurf*.

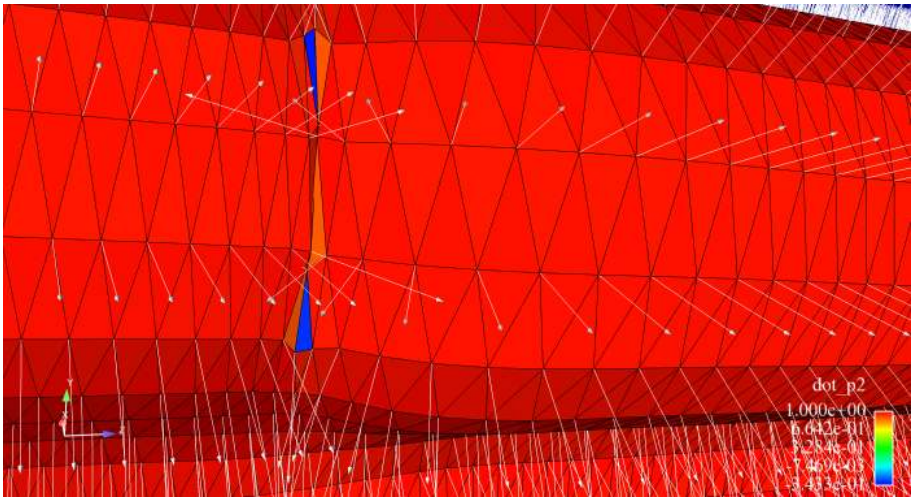


Figure 21. Ice surface for t_{ice}=16 min generated by *iceSurf* that highlights the region where the surface mesh is invalid.

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. Additionally, we employ a fast algebraic approach to project the surface deformation into the volume mesh. These techniques were demonstrated by application to sphere with an analytically-specified, synthetic accretion rate map and coupled *LEWICE3D–Loci/CHEM* simulations for rime and glaze ice accretions on a rectangular planform wing with a constant GLC305 airfoil section. Numerical results for the sphere show good agreement with the analytical solution. Results obtained for the rime ice case, for which the accretion rate map is based on collection efficiency, show similar trends to results predicted using *LEWICE3D*; however, discrepancies exist that can be attributed to the manner in which the accretion rate is computed. Results for the glaze ice case demonstrate that our approach can handle more complex shapes. Remaining issues can be attributed, in part, to the differences between surface normal evaluation that is employed in *iceSurf* and the lofting approach employed by *LEWICE3D* to generate the accretion rate map.

The progress reported here represents a part of an ongoing effort to develop the tools needed for the next generation of NASA's icing software. We are concurrently developing tools that employ local quality improvement operations, such as edge swaps, etc., and local mesh regeneration to maintain geometric mesh quality as the accreted surface and volume mesh evolve. We believe that this strategy of robust mesh generation coupled with local mesh quality improvement will provide a path forward for automated ice accretion prediction on complex configurations.

Acknowledgments

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

References

[1]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.

[2]Jiao, J., "Volume and Feature Preservation in Surface Mesh Optimization," *Proceedings,* 15^{th} *International Meshing Roundtable*, Springer-Verlag, Birmingham, AL, 2006, pp. 359–374.

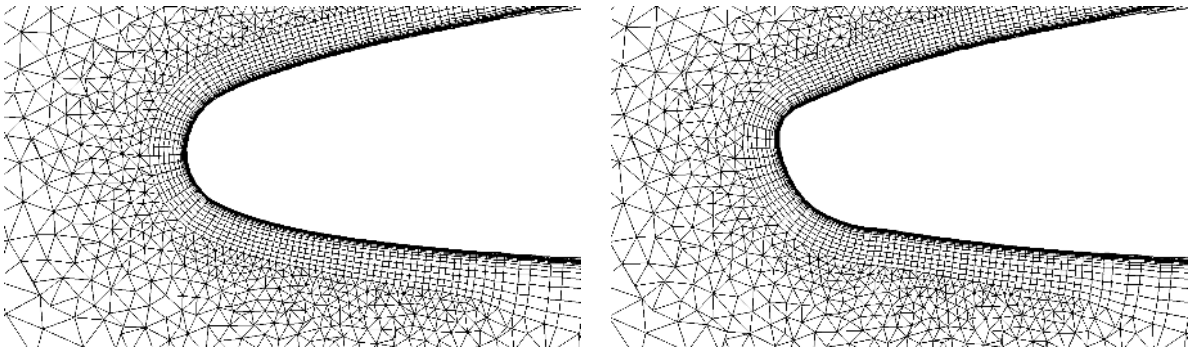
[3]Jiao, J., "Face Offsetting: A Unified Approach for Explicit Moving Interfaces," *Journal of Computational Physics*, Vol. 220, 2007, pp. 612–625.

[4]Wright, W., "User Manual for the NASA Glenn Ice Accretion Code LEWICE Version 2.2.2," Nasa cr-2002-21179, NASA, Cleveland, OH, Aug. 2002.

[5]Messinger, B., "Equilibrium Temperature of an Unheated Icing Surface as a Function of Airspeed," *Journal of the Aeronautical Sciences*, Vol. 20, 1953, pp. 29–42.

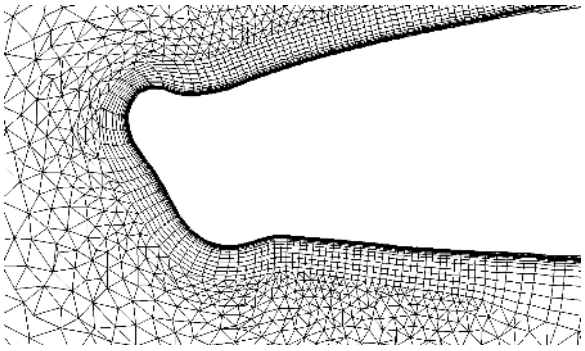
[6]Luke, E., , Collins, E., and Blades, E., "A Fast Mesh Deformation Method using Explicit Interpolation," *Journal of Computational Physics*, Vol. 231, 2012, pp. 586–601.

[7]de Boer, A., van der Schoot, M., and Bijl, H., "Mesh Deformation Based on Radial Basis Function Interpolation,"

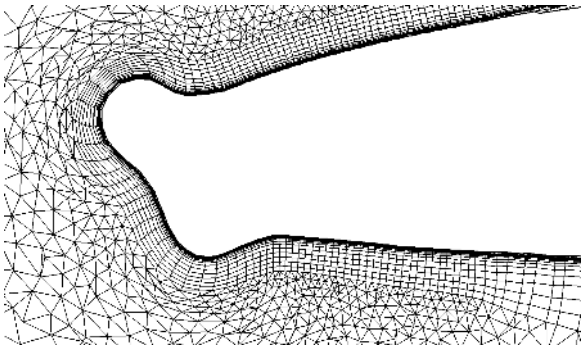


clean wing

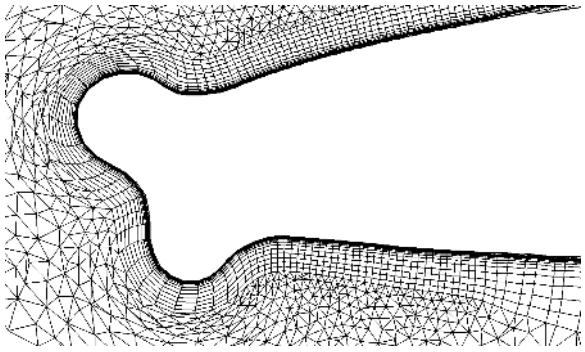
4-min accretion



8-min accretion



12-min accretion



16-min accretion

Figure 22. Evolution of the volume mesh in midspan plane.

Computers and Structures, Vol. 85, 2007, pp. 784–795.

[8]Luke, E., , and Cinnella, P., "Numerical Simulations of Mixtures of Fluids using Upwind Algorithms," *Computers and Fluids*, Vol. 36, 2007, pp. 1547–1566.

[9]Wright, W. and Rutkowski, A., "Validation Results for LEWICE 2.0," Naca cr-1999-208690, NASA, Cleveland, OH, Jan. 1999.

[10]Menter, F., "Two-Equation Eddy-Viscosity Turbulence Models for Engineering Applications," *AIAA Journal*, Vol. 32, 1994, pp. 1598–1605.

[11]Gaither, J., Marcum, D., and Mitchell, B., "SolidMesh: A Solid Modeling Approach to Unstructured Grid Generation," *Proceedings* 7^th *International Conference on Numerical Grid Generation in Computational Field Simulations*, Whistler, BC, 2000.