Advances in Distance-Based Hole Cuts on Overset Grids

William M. Chan*, Shishir A. Pandya†
NASA Ames Research Center, M/S 258-2, Moffett Field, CA 94035

An automatic and efficient method to determine appropriate hole cuts based on distances to the wall and donor stencil maps for overset grids is presented. A new robust procedure is developed to create a closed surface triangulation representation of each geometric component for accurate determination of the minimum hole. Hole boundaries are then displaced away from the tight grid-spacing regions near solid walls to allow grid overlap to occur away from the walls where cell sizes from neighboring grids are more comparable. The placement of hole boundaries is efficiently determined using a mid-distance rule and Cartesian maps of potential valid donor stencils with minimal user input. Application of this procedure typically results in a spatially-variable offset of the hole boundaries from the minimum hole with only a small number of orphan points remaining. Test cases on complex configurations are presented to demonstrate the new scheme.

I. Introduction

STRUCTURED overset grid technology has been successfully applied to perform modeling and simulation analysis on a wide variety of complex aerospace applications.[1–6] Over the years, various methods have been developed for domain connectivity or overset grid assembly.[7–15] Since both the surface and volume grids are allowed to overlap arbitrarily, grid points that fall inside solid bodies or outside the computational domain need to be identified and excluded from the process of solving the governing field equations. This step is sometimes called hole-cutting or grid-point blanking, and is the first step in the domain connectivity process. At grid boundaries where a flow solver boundary condition is not specified, and at the boundaries of holes from the hole-cutting process, the solution needs to be interpolated from neighboring overlapping grids. The second step in domain connectivity involves the search for donor cells (interpolation stencils) for the fringe points on such boundaries. The number of layers of fringe points N_f that requires interpolation from neighboring grids is dependent on the order of the flow solver differencing stencil. For example, for a five-point stencil in the flow solver, $N_f = 2$ is needed so that the first point from the fringe boundary where the flow solution is computed can have a complete 5-point stencil with two points on each side.

The process of hole-cutting involves the identification of grid points that will not be solved by the governing equations of the solver. At a minimum, grid points that fall inside solid boundaries of the geometry need to be removed from the computation. The set of blanked points identified in this step is sometimes called the minimum hole. The hole boundaries after this step may be located very close to the geometry walls. In a viscous computation, the fringe points at such hole boundaries typically have donor stencils that are in the viscous layers. For a fringe point that is not near a viscous wall of its own grid, a large discrepancy usually exists between its cell attributes and the donor-stencil cell attributes. This mismatch in cell attributes, such as cell volume, will usually result in poor inter-grid information transfer when solution gradients exist in the interpolation region. The cell attribute mismatches can be reduced by displacing the hole boundaries away from the solid walls. As the hole boundaries move away from the minimum hole, the amount of overlap between neighboring grids reduces from a maximum until there is no overlap between grids. Between the extreme states of maximum grid overlap and no grid overlap, there exist numerous acceptable locations of the hole boundaries from different grids.[15] It is the hole-cutting software's primary function to find and settle on one such hole boundary location.

*Computer Scientist, AIAA Senior Member
†Aerospace Engineer, AIAA Senior Member

Most hole-cutting algorithms in use today belong to one of two types: explicit or implicit hole-cutting. In explicit hole-cutting, the user is required to specify how the holes are cut. These could be in the form of user-defined surfaces as in the PEGASUS4 software,[7] or fast look-up reference maps of the minimum holes plus offset distances as in the X-rays approach.[8] Due to the tedious user inputs required in these explicit methods, the implicit hole-cut method,[9] which requires no explicit user inputs other than flow solver boundary conditions, has gained popularity in a number of domain connectivity software today.[10–13] Such methods involve searching for all possible donor stencils for every grid point in the volume grid system. The resulting fringe boundaries are then iterated further to match grid attributes such as cell volume, aspect ratio, and orientation between donor and receiver cells in the grid overlap region. While the manual effort is low for implicit hole cut methods, the computational time could be expensive. Implicit hole cut methods require a donor stencil search for every grid point in the volume grid, which involves order N^3 number of searches where N is a representative number of grid points in one dimension. On the other hand, explicit hole cut methods require a donor stencil search only at N_f layers of grid points at the grid outer boundaries and hole boundaries which involves only an order N^2 number of searches. Given the same donor stencil search procedure, explicit hole cut methods have an order of magnitude less points to search than implicit hole cut methods. Hence, explicit hole cut methods are typically much less expensive computationally than implicit hole cut methods, and thus are more suitable for relative motion problems where hole-cutting needs to occur at every time step.

Various criteria can be used to determine the location of the hole boundaries. As long as flow gradients are accurately transferred between grids, there appears to be no unique location that has been proved to be superior to others.[15] Depending on the cell attributes used in the hole-boundary determination process, different implicit hole-cut methods will produce different hole boundary locations. In explicit hole cut methods such as the original X-rays method, the final hole boundaries are based on a user-specified constant offset distance from user-defined surfaces that bound a closed volume. Such surfaces are typically the solid walls of geometric components, or surfaces that bound a special region in the flow field such as a wake or a plume. In the improved approaches in Ref. 15 and in the method described in this paper, the final hole boundaries are based on automatically-determined variable offset distances from the solid walls. A study from Ref. 15 suggests that variations in hole boundary locations away from solid walls have smaller or equal effects on aerodynamic loads than variation in other numerical scheme parameters such as turbulence models.

In the original X-rays approach to hole cutting,[8] a two-dimensional Cartesian map, together with surface pierce points in the third dimension, are used to represent the surface geometry of each component in the computation. This scheme requires significant manual effort to construct an X-ray map for each geometric component, specify a constant distance offset from the geometry wall for hole-cutting, and the list of grids that are cuttable by each X-ray map. Moreover, the resulting holes are limited to a constant distance offset from the geometry surface, while in cases where components are in close proximity, a spatially-varying distance offset from the geometry surface is needed to provide better quality inter-grid communication.

An effort to significantly reduce the user's effort and to improve the hole cut quality was initiated in the automated X-rays approach.[14,15] A volume grid system typically consists of near-body grids where a grid boundary conforms to a solid wall of the flow domain, and off-body grids that do not contain a solid wall boundary. In this approach, each geometric component of a complex configuration may be modeled by one or more near-body grids. The user only has to specify the flow solver boundary conditions for each grid along with a component tag for each solid wall grid surface that identifies its geometric component. Hole cutting or determination of the final hole boundary then proceeds in three steps. First, component X-ray maps are automatically generated, together with a list of grid subsets that are cuttable by each X-ray. These X-ray maps are then used to create a minimum hole by identifying grid points that are inside solid boundaries. Second, a hole boundary estimate using an automatic variable-distance offset from the minimum hole is generated using heuristic distance rules. An unsatisfactory number of orphan points typically remains after this step. Finally, orphan-points removal iterations are performed by local perturbations of the hole boundaries.

The current work continues the effort in improving the robustness of the improved X-rays approach by enhancing the first two steps described above. Previously, open boundaries of a component's surface triangulation were closed using an approximate procedure.[14] Minimum holes created from X-ray maps based on such approximately closed triangulations were frequently inaccurate. In the current work, an accurate closure of the open boundaries on a component's surface triangulation is introduced which leads to a precise determination of the minimum hole. For step two, deficiencies in the previous scheme included

an assumption of constant outer boundary extent of the near-body grids, and that blanked points were not accounted for in the heuristic distance rules. These assumptions typically resulted in a large number of orphan points after the initial hole boundary offset from the minimum hole. Multiple iterations of local hole boundary relocation were then needed to reduce the number of orphan points. In the current work, such deficiencies are eliminated by accounting for the spatially-varying outer boundary extents of the near-body grids after minimum hole cuts are performed, and by utilizing donor stencil maps in grid-point blanking decisions.

The heuristic distance rules discussed in this paper rely on efficient computation of component-based wall distances. The concept of using wall-distances to determine hole boundary locations has been explored on unstructured meshes.[16] Each geometric component is modeled by an unstructured near-body grid. Unstructured near-body grids from different components are allowed to overlap each other arbitrarily. The distance of any volume grid point to the wall of its own grid is first determined. The distance of a volume grid point to the wall from another grid is then found by interpolation from the grid point's donor stencil from the other grid. This scheme requires donor stencil search for all volume grid points prior to determining the hole boundaries and can therefore be expensive for the same reason that implicit hole cut methods are expensive. In the current work presented in this paper, the required wall distances are efficiently computed using Cartesian reference maps. More elaborate rules are established for the treatment of near-body and off-body grids with special attention paid to collar grids[17] that are commonly utilized at junctions of intersecting components in structured grid systems.

The use of Cartesian maps in a fast look-up procedure for distance to a component wall and locations of valid donor cells is discussed in Section II. These techniques are utilized in various steps of the procedures described in the subsequent sections. A robust procedure for determining an accurate minimum hole using cut cells on surface grids of the given grid system is presented in Section III. This enables the construction of more accurate local rules for near-body and off-body hole boundary placement presented in Section IV. Application of the new scheme to a number of complex configuration test cases is presented in Section V. Summary and conclusions are given in Section VI.

II. Fast Look-Ups Using Cartesian Maps

A Cartesian map offers a very fast but approximate way to perform a number of tasks related to grid searches. It can be used to determine the approximate location of a grid point relative to the curvilinear cells in a grid system, as well as the approximate value of a field variable in the computational domain that would otherwise have been more expensive to compute. The task of determining the curvilinear volume grid cell that contains an arbitrary point P is an expensive process that typically involves building a tree, traversing the tree and performing stencil searches. By relating the curvilinear grid cells to cells in a Cartesian map, the list of curvilinear grid cells that may contain P can be rapidly determined by a direct look-up of the Cartesian map cell that contains P. Similarly, values of a field variable at the vertices of a curvilinear mesh can be mapped to the vertices of the Cartesian map. The approximate value of the field variable at P can then be obtained by a direct look-up of the Cartesian map cell that contains P followed by trilinear interpolation from the Cartesian cell vertices.

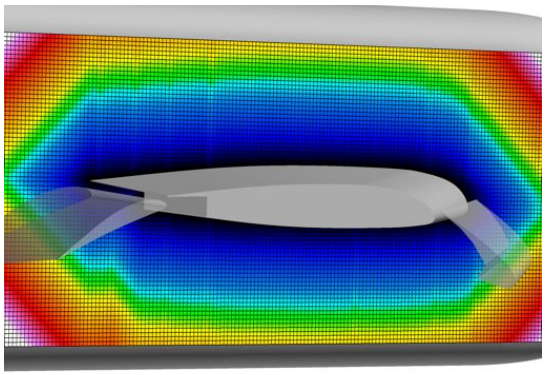


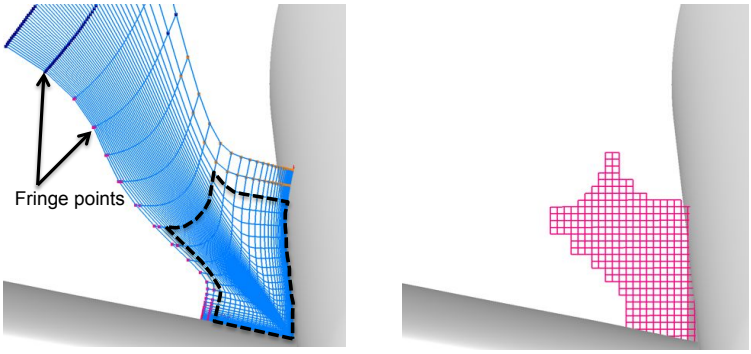
Figure 1. Cartesian map for distance to middle element wall of three-element wing system (color map from small to large distance: black, blue, cyan, green, yellow, red, magenta, white).

In the current work, Cartesian maps are used for two tasks: determining the approximate distance to a component's wall, and the approximate location of valid donor stencils in a grid. The approximate nature of the Cartesian maps is recognized and accepted since the algorithms described in this paper do not require exact values of the variables involved.

For the component wall distance Cartesian map, a uniform Cartesian mesh is generated encompassing the surface geometry of the component. The average surface grid spacing of the component is used to determine

the spacing of the map in all three directions. Cartesian map cells that are cut by the surface geometry of the component are first identified. An accurate distance-to-the-wall computation is applied to the vertices from such cells by direct projection to the surface. The distances to the wall for the remaining vertices in the Cartesian map are determined by a fast marching scheme from the vertices whose wall distances are already computed.[15] Given a point P, efficient methods for computing the approximate distance to the component wall for various locations of P relative to the Cartesian map are given in Ref. 15. Fig. 1 shows the wall distance Cartesian map for the middle element of a three-element high-lift system.

A Cartesian map of the approximate available donor stencils of a volume grid is constructed using a similar method as introduced by Meakin in Ref. 18, and utilized by Sitaraman in Ref. 12. Again, the average surface grid spacing of the volume grid is used to determine the grid spacing of the map in all three directions. A valid donor stencil is defined to be a grid cell where none of its vertices is a blanked point or a grid point whose solution value is interpolated from another grid cell (fringe point). By adding one or more extra layers of cells between valid donor stencils and available donor stencils, the approximation error in using a Cartesian map for look-up of an available donor stencil can be significantly reduced. Further constraints on the availability of donor stencils are discussed in Section IV. Once the list of available donor stencils is identified, the bounding box of these cells are then used to mark up Cartesian cells in the map via a logical array. The logical is set to true for Cartesian cells that contain an available donor stencil, and set to false otherwise. For a given point P, the Cartesian cell in the donor stencil map that contains P can be quickly determined. If it falls inside a marked cell in the donor stencil map, then it is inside an available donor stencil. Fig. 2a shows a volume grid slice of a curvilinear mesh. Fringe points are marked by symbols while the region of available donor stencils is marked by the dotted line. This region is retracted several layers below fringe cells at the outer boundary opposite to the wall to remove some orphan points due to insufficient overlap between the near-body grids away from the walls. Fig. 2b shows a slice of the available donor stencil map that lines up close to the curvilinear grid slice.



(a) (b)

Figure 2. Cartesian map of available donor stencils of curvilinear near-body grid. (a) Volume grid slice with fringe points marked by symbols and available donor stencils region marked by black dashed line. (b) Slice of Cartesian map of available donor stencils in the same vicinity.

III. Accurate Minimum Hole Cut

The minimum hole cut is defined by the hole boundaries formed after grid points that fall inside a solid body are blanked. An accurate determination of the minimum hole is needed prior to deploying the distance rules discussed in Section IV to push the hole boundaries away from the geometry surface. Grid point blanking for the minimum hole is accomplished using a closed surface triangulation of each geometric component, derived from the structured surface meshes in the configuration. For a configuration with two disjoint components, such as two spheres, the surface grid cells on each component form a closed surface. The X-ray map of each component can be automatically generated using the average surface grid spacing as the image plane spacing. For a slightly more complex configuration such as a wing/fuselage combination, the surface grid cells on the fuselage component might form a closed surface, but the wing component surface is typically open at the wing root. In order to construct a robust X-ray map for the wing, the collection of surface grids used to build the X-ray map needs to form a closed surface. An accurate procedure for closing the open surface is presented in Section III.A below. The open boundary curve of a component is mapped to existing surface grids from other components in the grid system. Surface cells that straddle the

boundary curve are cut and all cells that fall in the interior of the boundary curve, including the cut cells, are then used to fill the hole bounded by the open boundary curve. After a closed surface triangulation of a component has been created, a precise method for generating the minimum hole using a combination of standard X-rays and direct ray-casting is described in Section III.B.

III.A. Automatic Closure of a Component Open Surface

At the junction of two intersecting components, overset surface meshes from the components may be closed or left open, while a collar grid is used to resolve the geometry of the junction. In the wing/body example shown in Fig. 3a, the fuselage surface grid is typically constructed disregarding the presence of the wing and is usually a simple closed surface. The wing surface grid, including part of the collar grid used to connect the wing to the fuselage, is usually open at the wing root (see Fig. 3b). Using an X-ray map derived from this open wing cutter surface will not be able to reliably blank points from other grids that fall inside the wing root region, thus causing an error in the minimum hole cut process. Various options have been tried to solve this problem unsatisfactorily in the past including construction of an unconstrained triangulated surface from the open boundary curve,[19] and an approximate surface to generate a closing triangulation.[14] In the wing/body example, the triangulated surface formed by both methods does not necessarily match the fuselage surface at the wing root, especially when the open boundary is non-planar. This usually results in inaccurate hole cuts in the wing/body junction. These weaknesses, however, can be easily addressed if the open boundary can be closed using an existing surface definition from another component. In the wing/body example, we use the surface of the fuselage to close the open wing root.

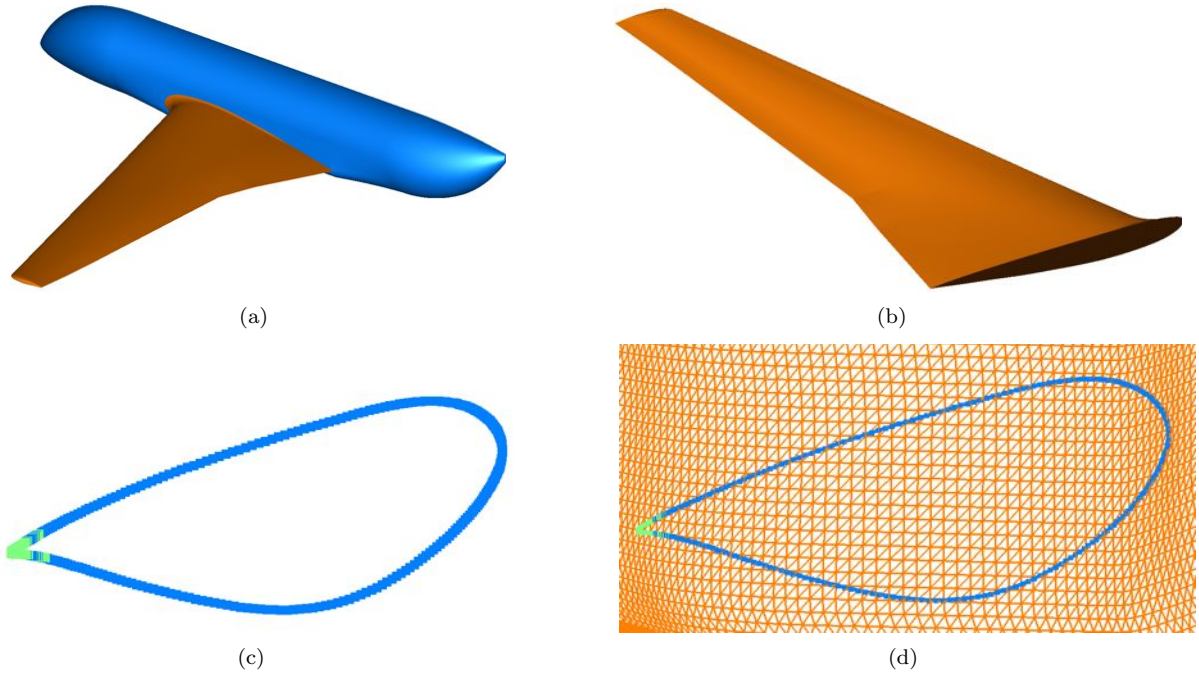


Figure 3. Wing with open root that needs to be closed using a subset of the fuselage. (a) Wing/body geometry. (b) Wing with open root. (c) Curves that bound open boundary at wing root. (d) Open boundary curves from wing root and fuselage surface grid cells.

Three scenarios can arise at a component's open boundary: (1) The open boundary falls on the surface of another component and the other component's surface grid cells completely cover the opening. (2) The open boundary falls on the surface of another component and the other component's surface grid cells only partially cover the opening. (3) The open boundary resides completely inside another component. The procedure described below has been designed to robustly handle the first scenario which is typically the most commonly occurring. With the resulting closed triangulated surface for the open component, the method guarantees that grid points from other components that fall inside the open component are properly blanked. The current work does not address scenarios two and three. In the absence of available surface grid cells to completely close the open boundary, the best that can be done is to fall back to an unconstrained triangulation of the open boundary as described in Ref. 19, or a closed triangulation provided by the user.

Such a closed triangulation of a component may be readily available from the CAD definition, or may have to be manually created.

The process of closing the open component surface begins by identifying and extracting the curves that bound the open boundary. For each surface grid subset that makes up the open component, a test is performed at the minimum and maximum boundaries in the J and K directions, where J and K are the structured grid indices on the surface mesh. At a given boundary, if a neighboring point from another surface subset from the same component is not found, the boundary curve is considered to be part of the component's open boundary. Next, the boundary curves that form a closed loop are grouped into an ordered loop set. The points along the curves on the loop are directly related to vertices in the open component triangulation. A component may have one or more open boundary loops, e.g., a pipe that is connected to two different components at its two ends.

For each ordered loop set, surface grid cells from another component (reference surface) are used to fill the interior. In the wing/body example, Fig. 3c shows the open boundary loop at the wing root. Triangulated surface grid cells from the fuselage can be used to fill the interior of the loop (see Fig. 3d). The possibility that a component's surface grid cells will help close all or part of the open loop only exists if the bounding box of the open loop is contained by the bounding box of the component surface grids. Thus, component surfaces that do not meet this criterion are not considered. It is, however, possible that multiple surface grids cover some or all of the open loop region. In these cases, each of these surface grids is used to obtain a closing surface; possibly resulting in overlapping closures of the open loop. The points from each curve in the ordered loop are projected and connected to the reference surface triangulation. Grid points in the overlapped regions between consecutive curves are automatically eliminated.

III.A.1. Connecting the Loop with the Reference Surface Triangulation

A major step in the process is to connect the closed loop curves to the reference triangulation. For each curve on the loop, a new curve is constructed that consists of all points from the original curve plus all points that are intersections between the segments on the original curve with edges of the reference triangulation. The process of creating this new curve is started by identifying the location of the first point in the first curve on the reference triangulation (See Fig. 4). A point on a triangle or an edge is projected onto the triangulation and added to the new curve. A point that coincides with a vertex is simply added to the new curve. Finally, if the point from the curve is not in the triangulation at all, the search goes to the next point on the curve until a point projects to the triangulation within a tolerance.

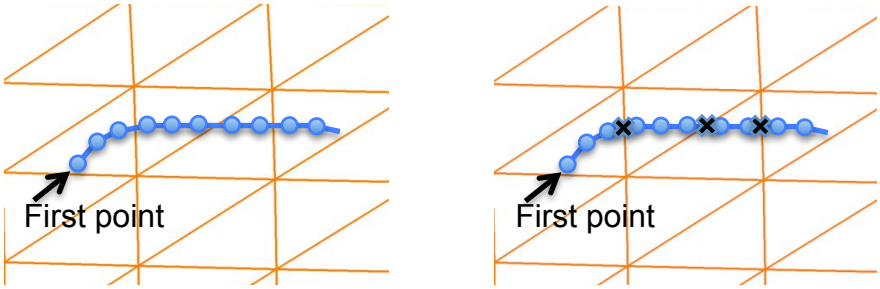


Figure 4. The first point on the open boundary curve and its neighboring reference triangles. (a) Original curve points marked by blue symbols. (b) New curve with new sites added (black crosses) at intersections at triangle edges.

Once the location of the first projectable point is determined, a "walk" to the next point on the curve is initiated. If the next point is in the same triangle as the previous point (e.g. first 3 points in Fig. 4), it is projected to the triangle and added to the new curve. If it is in another triangle, the point that cuts the edge as we "walk" to the new triangle is added to the new curve and we continue to "walk" to the next point on the original curve from the newly added point. If the next point is on an edge or a vertex of the current triangulation, that point is added to the new curve. If we step out of the triangulation across an edge during the walking procedure, this indicates that a part of the curve is not on the reference triangulation. In this case, we attempt to capture the other end of the curve by searching backwards. The two pieces of the curve are then concatenated. Note that we do not treat the case where a loop enters and leaves the reference surface more than once.

As the "walk" is executed, the type and location of each point on the new curve is stored for use in cutting the triangles on the reference component in which the loop resides. Each edge of a triangle that straddles the loop (e.g., the vector connecting the vertices of the edge, $\overrightarrow{V_1V_2}$ in Fig. 5) is compared against a vector $\hat{n}$ where $\hat{n}$ is the local surface normal on the open boundary of the open component. The dot product between $\overrightarrow{V_1V_2}$ and $\hat{n}$ is used to determine if vertex V_1 or V_2 is inside the loop. The inside vertex will be utilized later for cutting the triangles that straddle the loop. At the end of the walk around the loop, if there are multiple curves bounding the loop, they are concatenated into a single curve by eliminating points in the overlap regions.

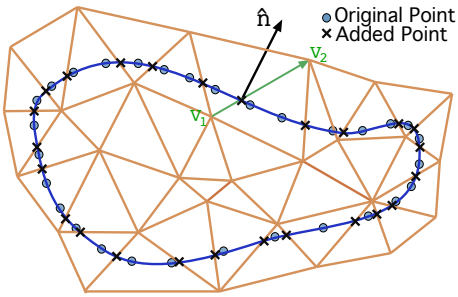


Figure 5. Determination of inside vertex on an edge of the reference triangulation that straddles the open boundary loop.

III.A.2. Filling the Loop Interior with Reference Surface Cells

Having determined which vertex of each straddling edge is inside, we are ready to cut the reference triangulation in the vicinity of the new single curve bounding the loop. First, a point belonging to the new curve that is on an edge entering a triangle and the corresponding point leaving the same triangle are identified. For convention, these bounding points are labeled N_i and N_e respectively (see Fig. 6). Note that because N_i and N_e are on edges of the reference triangulation, we can look up the inside vertices of those two edges and label them V_i and V_e. Now, when there is only one inside vertex or $V_i = V_e$ (see Fig. 6a), new triangles are introduced by simply connecting each point of the new curve between N_i and N_e to the inside vertex. The second case is that there are two inside vertices or $V_i \neq V_e$ (see Fig. 6b). In this case, each point between N_i and N_e can be connected to either V_i or V_e. New triangles are introduced by connecting to the vertex that makes a smaller included angle, keeping with the principles of a Delaunay triangulation. The resulting triangulation is shown in Fig. 6c.

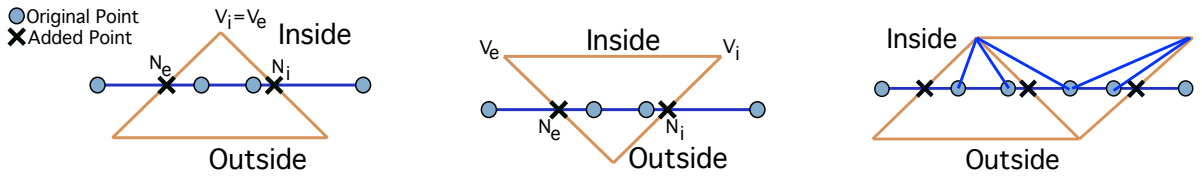


Figure 6. Cutting the boundary triangles. (a) One vertex inside. (b) Two vertices inside. (c) Cut triangles formed by connecting the inside vertices to the vertices on the new single curve bounding the open loop.

Special logic is required to determine proper cutting in two situations: when one (or both) of the points (N_i or N_e) is at a vertex of the triangulation instead of on an edge, or when both points are on the same edge of a triangle. Once all points on the new curve have been connected to a vertex of the reference triangulation, the new curve is bounded by the triangles that have been cut as shown in Fig. 7a,b.

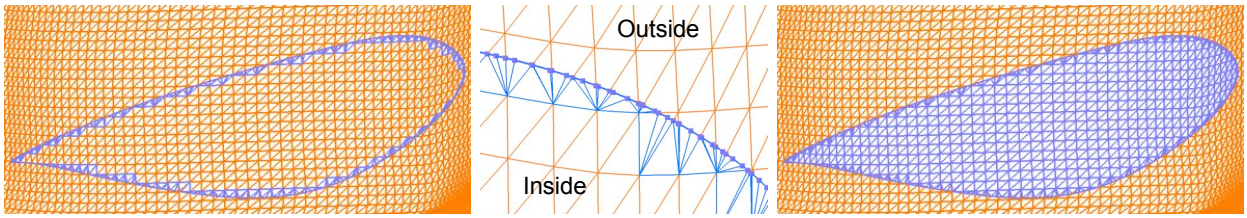
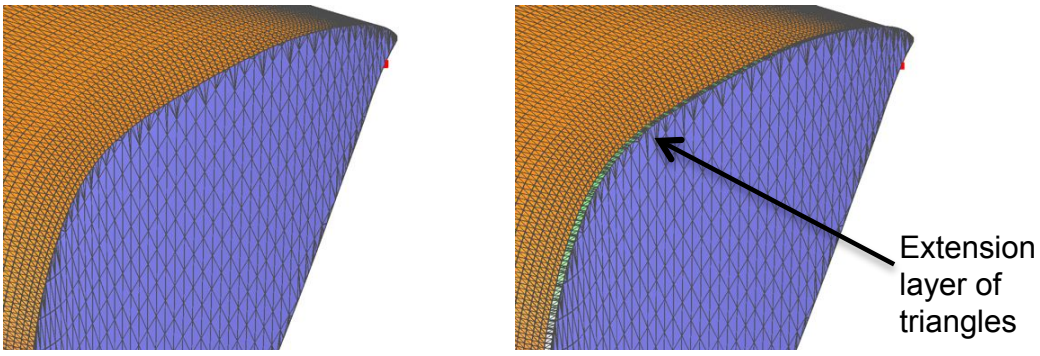


Figure 7. Cut and paint algorithm for filling the open boundary loop interior. (a) Cut cells along open boundary loop (far view). (b) Cut cells along open boundary loop (close-up view).(c) Triangles used to fill open boundary loop interior after applying painter's algorithm from the cut cells.

To identify the triangles inside the open hole that do not need to be cut, a boundary triangle with an entire edge inside the loop becomes the starting point. The triangle on the other side of that edge is inside the hole. From this seed triangle, we find all other inside triangles using a painter's algorithm.[20] Appending

the triangles identified by this process to the cut triangles at the new curve boundary gives the set of triangles that close the hole (see Fig. 7c).

The new filler triangulation (Fig. 8a) is now shifted a small distance in the direction normal to the new surface and towards the inside of the reference surface. This is to ensure proper cutting of the surface and volume meshes from the reference surface. The boundary vertices of the new triangulation are connected to their respective parent on the new curve and the resulting quadrilaterals are bisected into triangles. These new triangles are appended to the hole triangulation and to the original open boundary component triangulation resulting in a new cutter shown in Fig. 8b. If the reference surfaces cover the hole completely as is the situation in a majority of cases, the new cutter is a completely closed surface.



(a) (b)

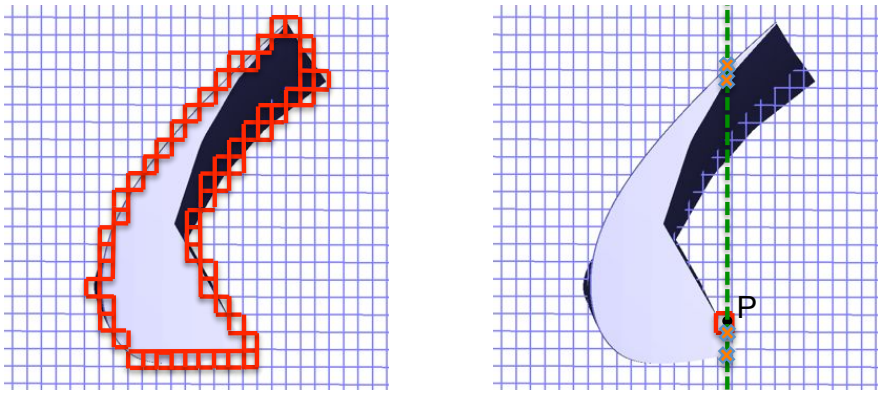
Figure 8. Extension of hole filler triangulation. (a) Unextended triangulation used to fill open boundary at wing root. (b) Extended triangulation used to fill open boundary at wing root.

In half-body configuration cases, reference surfaces are not available to fill open boundaries that end at the symmetry plane, e.g., open boundary on the half fuselage at the symmetry plane. In these cases, the entire loop is guaranteed to be on a single plane. Here, the loop is duplicated and shifted to the other side of the symmetry plane. Connecting this shifted loop to the original loop and bisecting the resulting quadrilaterals results in a new triangulation on the other side of the symmetry plane which is connected to the open boundary component triangulation to form the extended cutter. This extended cutter can then be used to build robust X-ray maps for the open boundary component.

III.B. Precise Minimum Hole Cut Using Closed Triangulation of Component

An X-ray map provides an approximate discrete representation of the surface of a component. As one refines the image plane spacing of the map, a more and more accurate representation of the component surface is formed. However, in the determination of whether a grid point P is inside or outside of the component surface (minimum hole), an erroneous conclusion from the inside/outside test is sometimes possible when P falls between discrete points on the X-ray map where a linear representation may not accurately model the surface geometry. In the current work, grid points that are located close to the component surface will receive a more accurate test. Any cell in the component wall distance Cartesian map discussed in Section II that is cut by the component surface triangulation is marked (see red cells in Fig. 9a) and is used to indicate if a point is close to the component surface. For an arbitrary grid point P, the cell in the wall distance Cartesian map that contains P can be quickly determined. If P falls in a marked cell, a ray is cast in the Z direction through P and the pierce points on the component surface triangulation are determined (see orange cross symbols in Fig. 9b). The Z-coordinate of P relative to the sorted Z coordinates of the pierce points is used to determine the inside/outside status of P in a similar manner as the standard X-rays approach.

This exact ray casting scheme does introduce more computational work but is applied to only a small fraction of the total number of test points where an accurate determination of the inside/outside status is needed. The majority of grid points to be tested should fall outside of the marked cells and will receive the more efficient inside/outside test using standard X-rays. For the test cases presented in Section V, the fraction of grid points that requires exact ray casting ranges from 0.1% to 0.7%. The extra wall clock time incurred by using exact ray casting for this small fraction of points ranges from a few tenths of a second for the small cases to about 1.5 seconds for the larger cases. This is a trivial increase when compared to the total wall clock time for performing domain connectivity.



(a) (b)

Figure 9. Component surface and slice of distance to wall Cartesian map. (a) Cartesian cells cut by surface triangulation are marked by red cells. (b) Grid point P (black symbol) falls inside a red cell. The minimum hole cut test is performed by casting a ray in the Z-direction (green dashed line) and comparing the Z coordinate of P relative to the sorted Z coordinates of the pierce points (orange crosses) on the component surface triangulation.

IV. Hole Boundary Estimation Using Distance Rules and Donor Stencil Maps

Given that a minimum hole has been determined, heuristic rules are used to offset the hole boundary away from the geometry surface. Volume grids in a grid system are classified as either a near-body grid or an off-body grid. A near-body grid is defined to be a grid that contains a solid wall boundary and typically has tight grid-point clustering adjacent to the solid wall to capture the viscous layer. All other grids are defined to be off-body grids. These may include Cartesian grids that are used to fill the space around the surface geometry, and various specialized grids employed to resolve flow features of interest such as shear layers, wakes, and plumes.

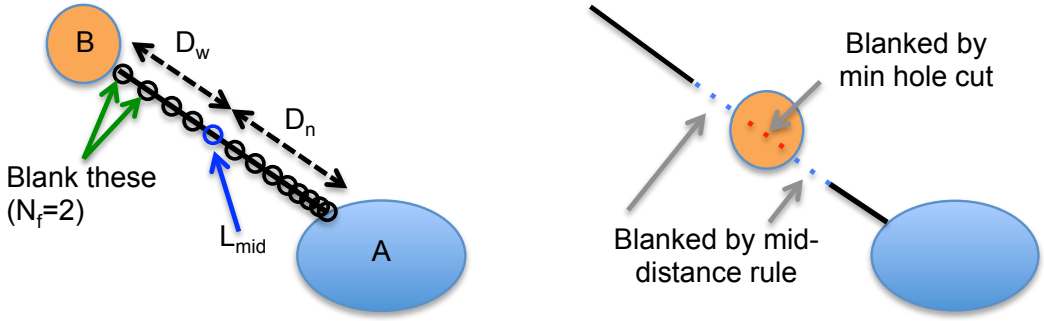
After the minimum hole cut, grid points from near and off-body grids that lie inside a solid wall boundary have been blanked. The resulting hole boundaries reside very close to the solid walls. The distance rules described in the subsections below are designed to displace the hole boundaries away from the solid walls using appropriate spatially-variable offsets. The intention is to move the hole boundaries so that the fringe points do not receive donor interpolations from cells that are vastly different in size to those of the fringe points while maintaining sufficient overlap between grids. Since the rules described in this paper do not refer to vertex connectivity within each grid, they are equally applicable to both structured and unstructured overset grids. However, only structured grid examples will be presented in this paper.

With stretched curvilinear and Cartesian grids in various relative orientations overlapping arbitrarily, it is difficult to define an optimal location of the hole boundary away from the wall. It has been demonstrated in Ref. 15 that aerodynamic loads computed from grid systems with various hole boundary locations away from the wall do not vary much for hole boundary offset distances that fall between 25% - 75% of the distance from the wall to the grid outer boundary in the wall-normal direction. On first thought, one might impose that the hole boundaries be placed away from the walls such that there are no orphan points or that the number of orphan points is as small as possible. If there is sufficient overlap between adjacent near-body surface grids to avoid the presence of orphan points on the surface, orphan points in the volume mesh can always be avoided by allowing the off-body grid hole boundaries to remain close to the wall surface. However, this would mean grid communication between grid points with large discrepancies in grid attributes such as cell sizes. If the near-body grids do not support an orphan-point-free status among themselves due to local large discrepancies in grid spacings, should off-body grid cells be allowed to remain unblanked close to the wall? In the current work, this decision is left for the user to make and is discussed in more detail in Section IV.D.

IV.A. Blanking Between Near-Body Grids

A complex configuration is constructed from multiple geometric parts or components. For example, a subsonic aircraft may contain components such as the fuselage, right and left wings, pylons, nacelles, vertical and horizontal tails. Each component may be modeled by one or more grids. At the junction between intersecting components, a collar grid is typically used to model the geometry of the junction. Hence, a

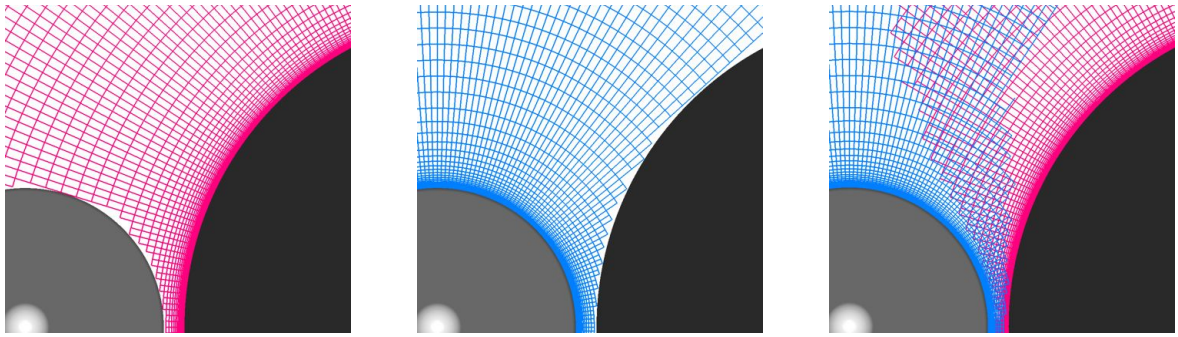
collar grid is made up of two parts where each part resides on a different parent component. In the current work, each component of the geometry is assigned an unique name. As a result, each near-body surface grid point inherits an associated component name of the geometric part that it resides on. Similarly, an associated component name can be assigned to each near-body volume grid point by tracing it to the surface via a grid line and picking up the same associated component name as the surface point.



(a) (b)

Figure 10. Mid-distance rule for grid point blanking between near-body grids from components A and B. (a) Near-body grid line emanating from component A (D_w = distance to nearest wall from un-associated component, D_n = distance to wall from associated component, L_{mid} = grid index on grid line at about mid-distance to component B). (b) Grid line from component A piercing component B and continuing on through the other side (dotted red line: points blanked by minimum hole cut, dotted blue line: points blanked by mid-distance rule, black solid line: field points).

For each grid point in a non-collar near-body grid, let D_w be the distance to the nearest wall belonging to a component that is different from the associated component of the grid point. Also let D_n be the distance to the nearest wall on its associated component. Now consider a grid point along a grid line emanating from the surface with index direction L. Starting from the surface, the first grid point in L that satisfies $D_w < \epsilon_c D_n$, where ϵ_c is a user-controllable parameter, is identified and let this grid index be L_{mid} (see Fig. 10a). For most cases, a default value of $\epsilon_c = 1.0$ is used which corresponds to the case when L_{mid} is at approximately the mid-point between the two components. In more difficult cases, it is sometimes necessary to reduce ϵ_c to a value less than one in order to make hole cuts closer to the unassociated component wall.



(a) (b) (c)

Figure 11. Grid point blanking between non-collar near-body grids. (a) Minimum hole on grid slice from right component. (b) Minimum hole on grid slice from left component. (c) Hole boundary estimate after application of near-body blanking rule.

Next, grid points on this L-direction grid line that satisfy (1) $L > L_{mid} + N_f$, and (2) $D_w < \epsilon_c D_n$ are then blanked, where N_f is the number of layers of fringe points specified (see Fig. 10a). The addition of N_f to the test is needed to provide proper overlap between neighboring grids. The second condition $D_w < \epsilon_c D_n$ is needed for cases where the grid line continues through the unassociated component and comes out on the other side (see Fig. 10b). In such cases, grid points on the other side that do not satisfy $D_w < \epsilon_c D_n$ are left unblanked if there are at least $2N_f + 1$ points left. Fig. 11a,b shows the minimum hole between two disjoint components. The result of applying the near-body blanking scheme described above is shown in Fig. 11c.

A similar mid-distance rule is also applied to collar near-body grids. Since collar grids are used in the junction between two parent components, the closest distance to one of the parent components is very small at the components junction near the wall surface. The distance to a wall search along grid lines normal

to the surface cannot begin at the surface. Instead, it begins at the first non-blanked point from the outer boundary opposite to the wall and heads backwards towards the wall. The main goal here is to blank points near the outer boundary that are too close to any wall.

IV.B. Blanking Between Collar Grids that Share the Same Parent Component

A collar grid is used in the intersection region between two parent components. Let J be the grid direction on the collar grid that wraps around one of the components, K be the grid direction where part of a grid line lies on one parent while the other part lies on the other parent, and L be the grid direction normal to the surface. The next step involves grid point blanking between neighboring collar grids that share a common parent component. For example, in a high-lift system shown in Fig. 12, the collar grids from the slat, wing, and flap all share the fuselage as a common parent component. Fig. 13 shows the region between the wing/fuselage collar grid and the slat/fuselage collar grid after the minimum hole cut. Surface grid points on grid lines in the K direction from both collar grids need to be retracted away from the minimum hole since there are points on these grid lines that lie close to the wall of another component. Similarly, volume grid points from both collars along grid lines in both the K and L directions may lie close to the wall of another component and need to be retracted (see Fig. 13c).

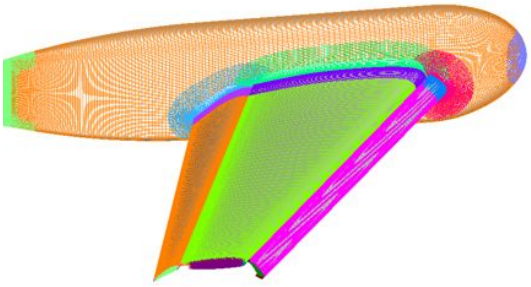
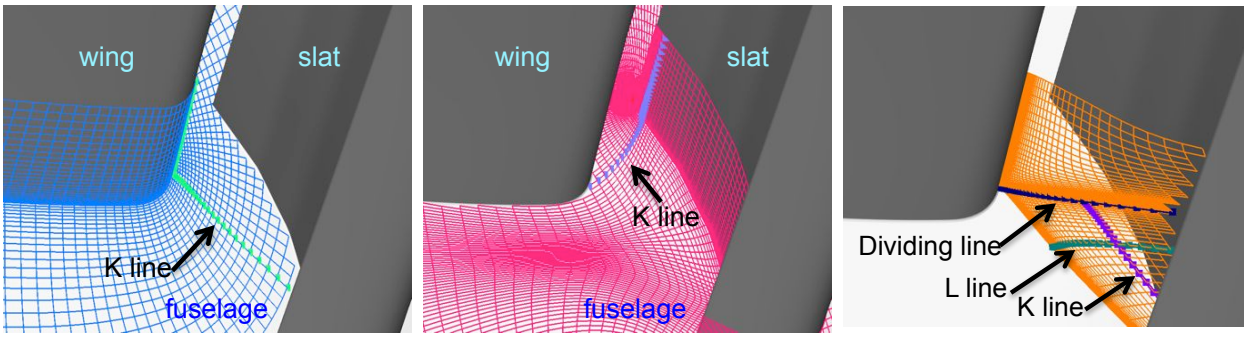


Figure 12. Trap Wing high lift system with fuselage, slat, wing, and flap.



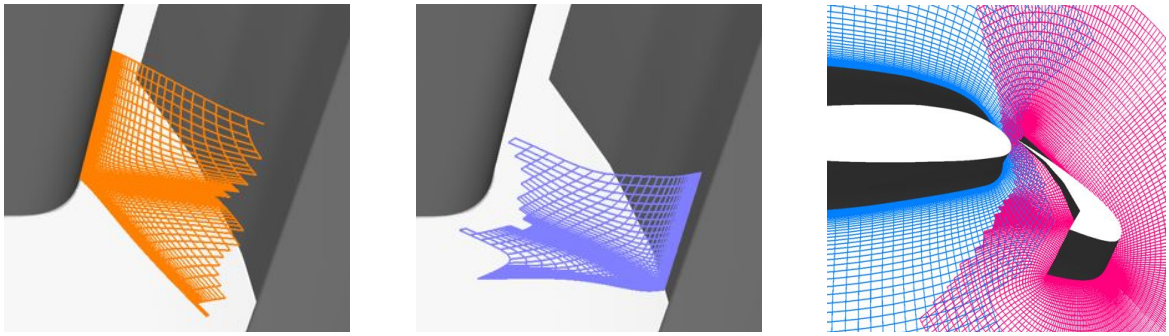
(a) (b) (c)

Figure 13. Collar grids that share the same parent component after minimum hole cut. (a) Wing/fuselage collar surface grid. (b) Slat/fuselage collar surface grid. (c) Wing/fuselage collar volume grid slice.

For each pair of collar grids that share the same parent component, a mid-distance blanking rule is applied to the K and L grid lines emanating from a reference location towards the wall of one of the components. In the L direction, the reference location is the wall from which the L direction grid line emanates. In the K direction, the reference location is the dividing surface between the two parent components of the collar volume grid (see Fig. 13c). After applying the mid-distance blanking rule to the L and K families of grid lines, the resulting volume and surface grid slices of both collar grids are shown in Fig. 14.

IV.C. Blanking Between a Collar Grid and Its Parent Components

The next step involves grid point blanking on grids that belong to the two parent components of a collar grid in the component intersection region. In the high-lift system example above, three collar grids are utilized, one at each of the junctions between the fuselage and its three attached elements (slat, wing, and flap). This grid-point blanking step will work on grid points on the main fuselage, slat, wing and flap grids that are in the vicinity of the respective collar grids. Since no further grid point blanking on the collar grids is needed beyond this step, the valid donor stencils on the collar grids will not change from here on. It is therefore safe to use the donor stencil maps of the collar grids discussed in Section II to determine grid point blanking on the grids that belong to the parent components of the collars.

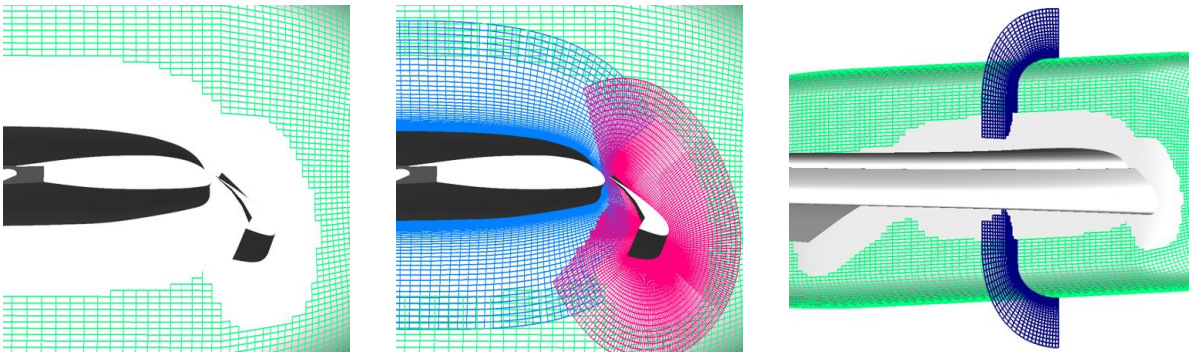


(a) (b) (c)

Figure 14. Collar grids that share the same parent component after application of collar-to-collar blanking rule. (a) Volume grid slice from wing/fuselage collar. (b) Volume grid slice from slat/fuselage collar. (c) Surface grids from both collars.

Let J and K be the grid index directions of the surface grids, and let L be the grid index direction normal to the wall. If the volume grids of the collar and its parents are generated with proper overlap in the J and K directions all the way from the wall to the outer boundary opposite to the wall, then the full extent of the collar volume grid in J and K can be valid donor cells at each layer in L provided the cell does not contain any fringe points. In this best scenario, available donor cells in the collar grid extend from $L = 1$ to $N_L - N_f - 1$, where N_L is the number of grid points in the L-direction. In cases where there is insufficient grid overlap between the collar and its parents away from the wall, the upper limit in L of available donor stencils has to be reduced using a user-controllable parameter L_{up}. The default value of L_{up} is set to $N_L - N_f - 1$. Fig. 2a shows a collar grid slice where L_{up} has been reduced below its default value. The corresponding donor stencil map around this grid slice is shown in Fig. 2b.

After the available donor stencil map on the collar grid has been constructed, grid point blanking on the parent grids proceeds by ensuring proper overlap with the collar grid. A parent grid point can be blanked if itself and N_{look} points in the $\pm J, \pm K$, and $\pm L$ directions reside inside an available donor stencil of the collar grid, where N_{look} is usually set to N_f, the number of required fringe points. Occasionally where there are large discrepancies in grid spacings between grids, N_{look} may have to be increased to maintain proper overlap. The donor stencil look-up test is efficiently performed since the available donor stencil map for the collar grid is Cartesian. Fig. 15 shows the result of applying the above scheme in the slat and wing collar grid regions of the high-lift system example.



(a) (b) (c)

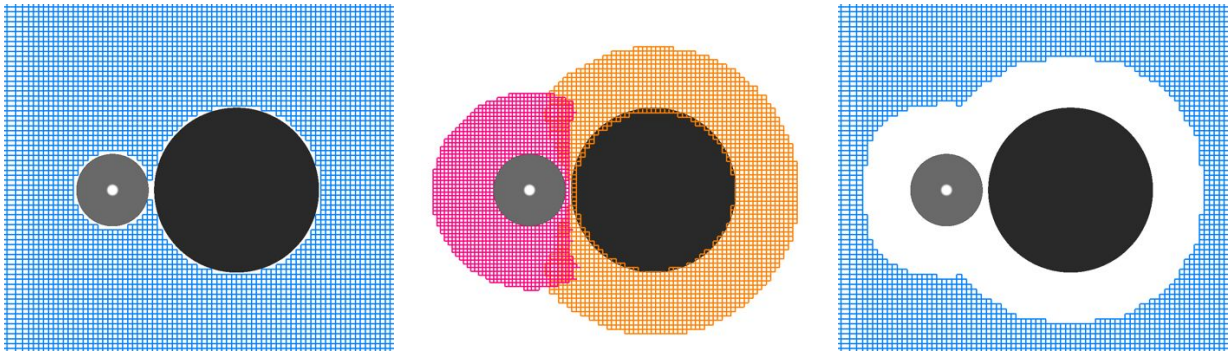
Figure 15. Grid point blanking on parent grids of collar grids. (a) Blanked points on parent component (fuselage) surface grid. (b) Surface grids in collar region after grid point blanking. (c) Parent component (fuselage) volume grid slice and surface grid after grid point blanking.

IV.D. Blanking of Off-Body Grids

At the end of the previous step, grid-point blanking between all near-body grids is complete. Since the available donor stencils in all near-body grids are now frozen, donor-stencil maps of the near-body grids can now be used to consider grid-point blanking for the off-body grids. Similar considerations on proper grid overlap exist here as in the previous section. If all near-body grids have been constructed so that proper

overlap is maintained in the J and K directions from the wall to the outer boundary opposite to the wall, then the off-body grids only communicate with the near-body grids near the $L = N_L$ boundaries of the near-body grids. In practice, this is rarely accomplished in complex configurations. The L_{up} and N_{look} parameters introduced in the previous section are also applicable here to improve grid overlap in the grid point blanking scheme. Default values chosen for these parameters are based on overset grid best practices. When large deviations from these defaults are needed to reduce the number of orphan points, the user is advised to consider improving the cell size compatibility and amount of overlap between neighboring grids, rather than over-extending the values of these parameters.

In order to maintain low memory usage in the code, the available donor stencil map of each near-body grid is constructed, used to mark up off-body grid points that can be blanked, and then the map is deallocated. For each near-body grid, grid point blanking on all off-body grids that overlap the near-body grid is performed in parallel. Fig. 16a, b, and c show the minimum hole on an off-body grid slice, the available donor stencil maps of the near-body grids, and the hole boundary estimate on the off-body grid slice after applying the grid point blanking scheme described above.



(a) (b) (c)

Figure 16. Grid point blanking in off-body grids. (a) Off-body grid slice with minimum hole blanking. (b) Slices of Cartesian donor stencil maps of near-body grids. (c) Hole boundary estimate on off-body grid slice after application of off-body grid blanking rule.

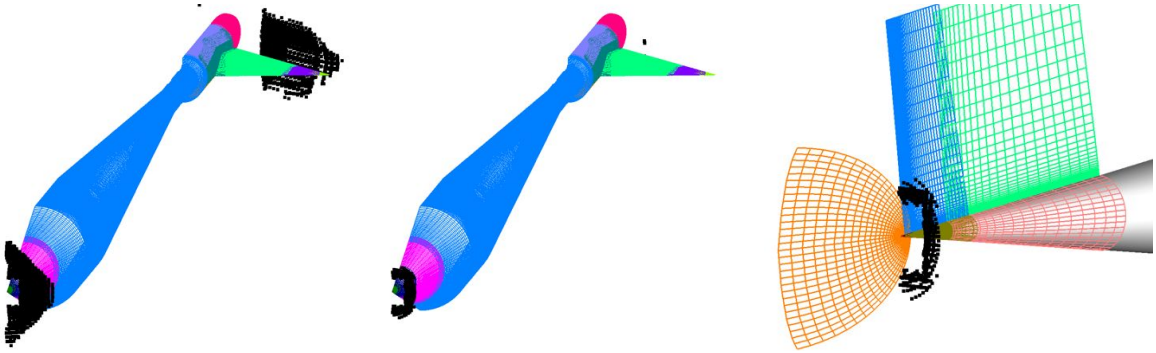
V. Test Cases

The scheme described in the previous section was applied to a number of test cases. All cases were run on a Linux workstation using 24 OpenMP threads and the Intel Fortran compiler. The wall clock times for the smaller test cases were not improved when compared to running with just 8 or 16 OpenMP threads. However, the larger cases do show good time savings when using more threads. Improvements obtained by using the more accurate minimum hole cut and more sophisticated hole boundary estimates in the current scheme are measured by comparing the number of orphan points that remain after the hole boundary estimate step with that from the previous method described in Ref. 15. Since the primary objective of the current work is to improve the hole boundary estimate (step 2 described in Section I), the orphan point removal iterations[15] (step 3 described in Section I) are not performed.

V.A. Delta-Wing-Body

The delta-wing-body test case consists of a delta wing, fuselage and sting near-body grids embedded in off-body Cartesian grids. This is the same geometry from the 1st AIAA Sonic Boom Workshop. The system contains 32.6 million points and 17 grids. After the initial hole boundary estimate, 4748 orphan points remained with the previous scheme, while 539 orphan points remained with the new scheme (see Fig. 17). The orphan points near the wing leading edge from the previous scheme are now absent in the new scheme due to the use of donor stencil maps when blanking the off-body grid points. A large cluster of orphan points remained behind the back of the sting in both cases. On closer examination, this is caused by the lack of overlap between the near-body grids in this region. Fig. 17c shows grid slices on the symmetry plane in the back region of the sting. The neighboring grids have proper overlap on the surface but rapidly deviate away from each other as the grids grow in the normal direction. Fewer orphan points could be accomplished in this case by allowing off-body grid cells to remain unblanked closer to the geometry surface, thus filling

in the space occupied by the orphan points. However, the flow solution may not be accurately transferred between grids due to the large discrepancy in cell sizes between the off-body and near-body grids. In this case, it is beneficial to have the connectivity software report these orphan points and identify the flaws in the near-body grids so that they can be fixed prior to running the flow solver.

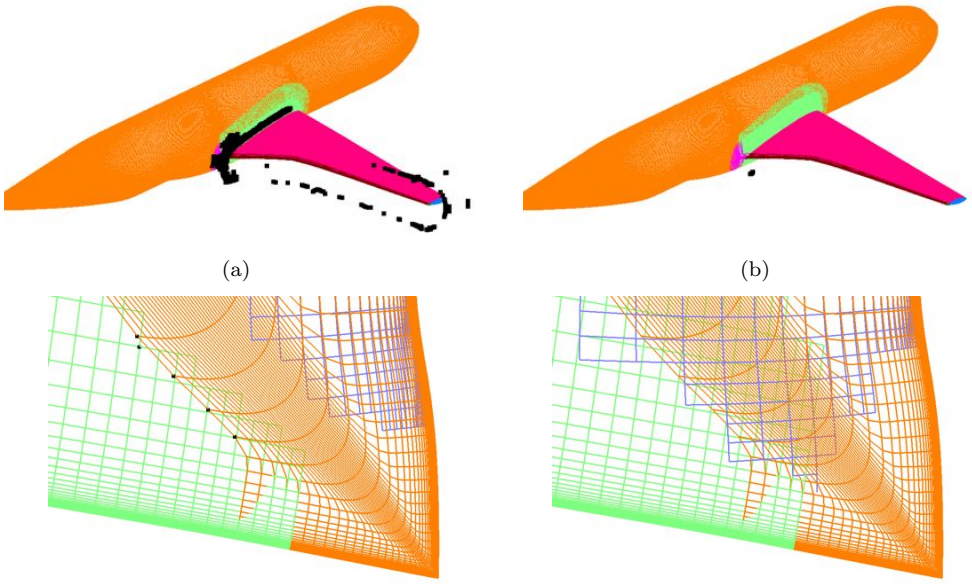


(a) (b) (c)

Figure 17. Rear view of delta-wing-body-sting test case with orphan points marked by black symbols. (a) Orphan points from previous scheme (4748). (b) Orphan points from current scheme (539). (c) Grid slices in back region of the sting showing lack of overlap between near-body grids.

V.B. Subsonic Wing-Body

The subsonic wing-body geometry consists of a swept wing and a fuselage which is also known as the Common Research Model from the AIAA Drag Prediction Workshops. The system contains 17.8 million points and 14 grids. After the initial hole boundary estimate, 2576 orphan points remained with the previous scheme, while 10 orphan points remained with the new scheme (see Fig. 18). A better estimate of the hole boundary in the wing/fuselage collar junction by the new scheme is able to eliminate a large number of orphan points in this region. The orphan points around the outer boundary of the wing are eliminated with the use of the donor stencil maps for blanking the off-body grid points.



(a) (b)

(c) (d)

Figure 18. Common Research Model test case with orphan points marked by black symbols. (a) Orphan points from previous scheme (2576). (b) Orphan points from current scheme (10). (c) Grid slices from previous scheme in wing/fuselage collar grid region. (d) Grid slices from current scheme in wing/fuselage collar grid region.

V.C. Tank-Booster

The tank-booster test case consists of a cylindrical tank in close proximity to a rocket booster with the near-body grids embedded inside an off-body Cartesian grid. The system contains 28.5 million points and 6 grids. After the initial hole boundary estimate, 102210 orphan points remained with the previous scheme, while no orphan points remained with the new scheme (see Fig. 19). A better estimate of the hole boundary in the small gap region between the two components by the new scheme is able to eliminate all orphan points in this region. The use of the near-body grid donor stencil map to blank off-body grid points has resulted in a smoother hole boundary in the off-body grid compared to the previous scheme which utilized an approximate map of the near-body outer boundary distance (see Fig. 19c,d).

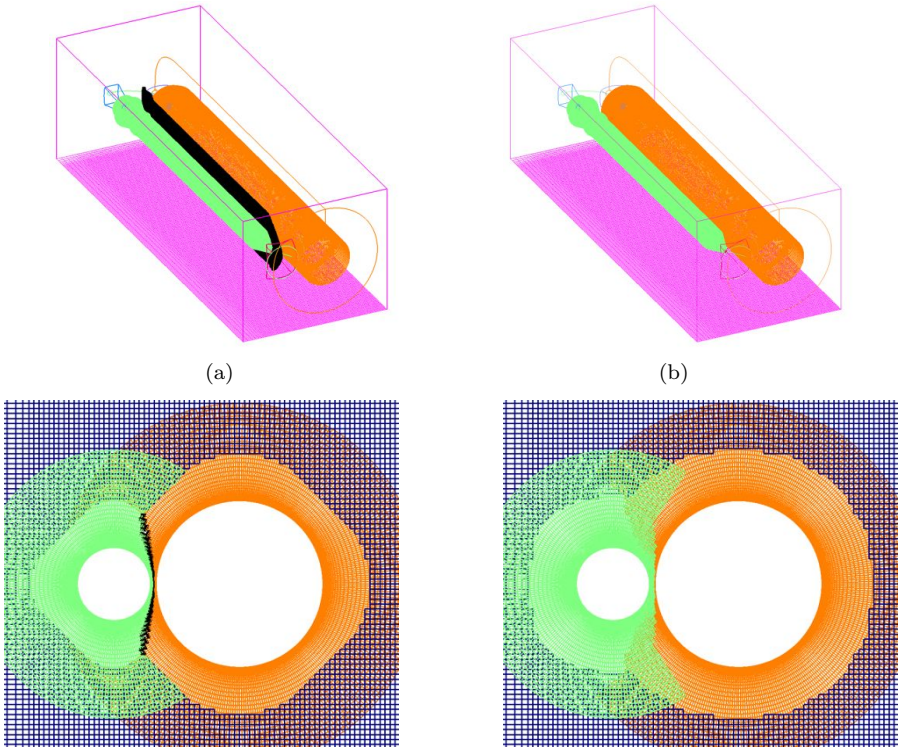


Figure 19. Tank-booster test case with orphan points marked by black symbols. (a) Orphan points from previous scheme (102210). (b) No orphan points from current scheme. (c) Grid planes from previous scheme at cut through constant streamwise coordinate. (d) Grid planes from current scheme at cut through constant streamwise coordinate.

V.D. High-Lift System

The high-lift test case, also known as the Trap Wing, consists of near-body grids for a slat, wing, flap, and fuselage. The three elements are in close proximity to each other. These are embedded inside off-body Cartesian grids. The system contains 50.6 million points and 24 grids. After the initial hole boundary estimate, 85000 orphan points remained with the previous scheme, while only 25 orphan points remained with the new scheme (see Fig. 20). A better estimate of the hole boundary in the tight gaps between the three components by the new scheme is able to eliminate most orphan points in these regions.

V.E. Feedline with Bracket

The Feedline with Bracket test case consists of a tank, a feedline that is connected directly to the tank at one end and through a fairing block to the tank at the other end. Additionally, there is a bracket that sits on the tank and cradles the feedline and is connected to the feedline at two ends (see Fig. 21). The system contains 82.7 million points and 23 grids. After the initial hole boundary estimate, 257000 orphan points remained with the previous scheme, while 381 orphan points remained with the current scheme (see Fig. 21). A better estimate of the hole boundary in the tight gaps between the various components by the current

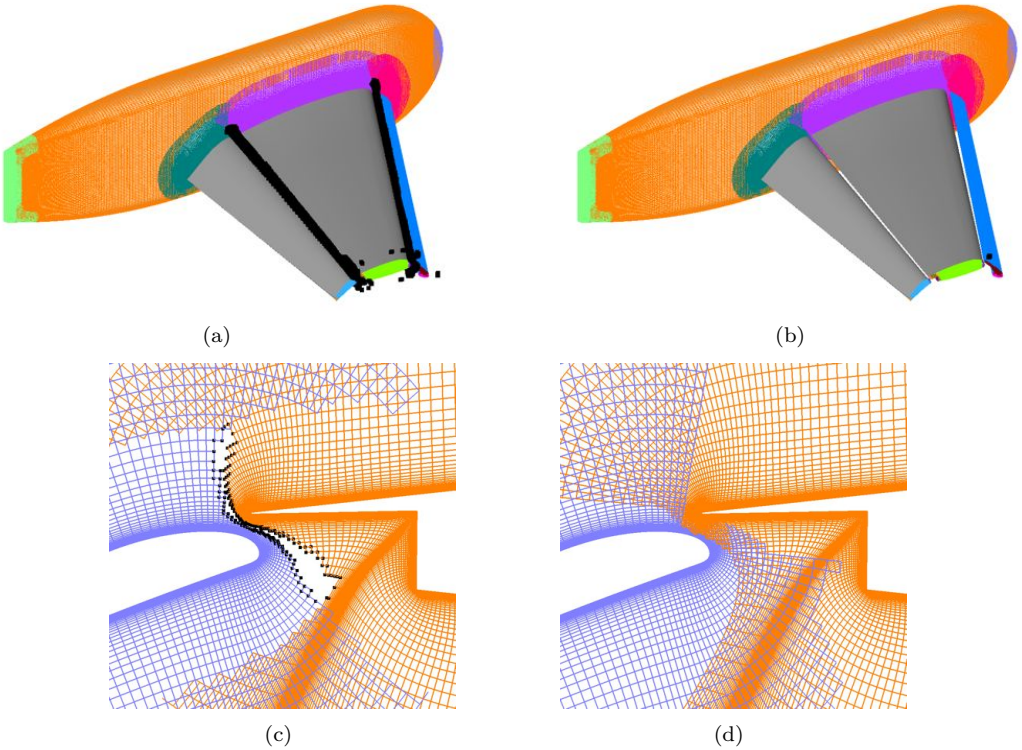


Figure 20. High-lift system test case with orphan points marked by black symbols. (a) Orphan points from previous scheme (85000). (b) Orphan points from current scheme (25). (c) Grid planes from previous scheme at span cut. (d) Grid planes from current scheme at span cut.

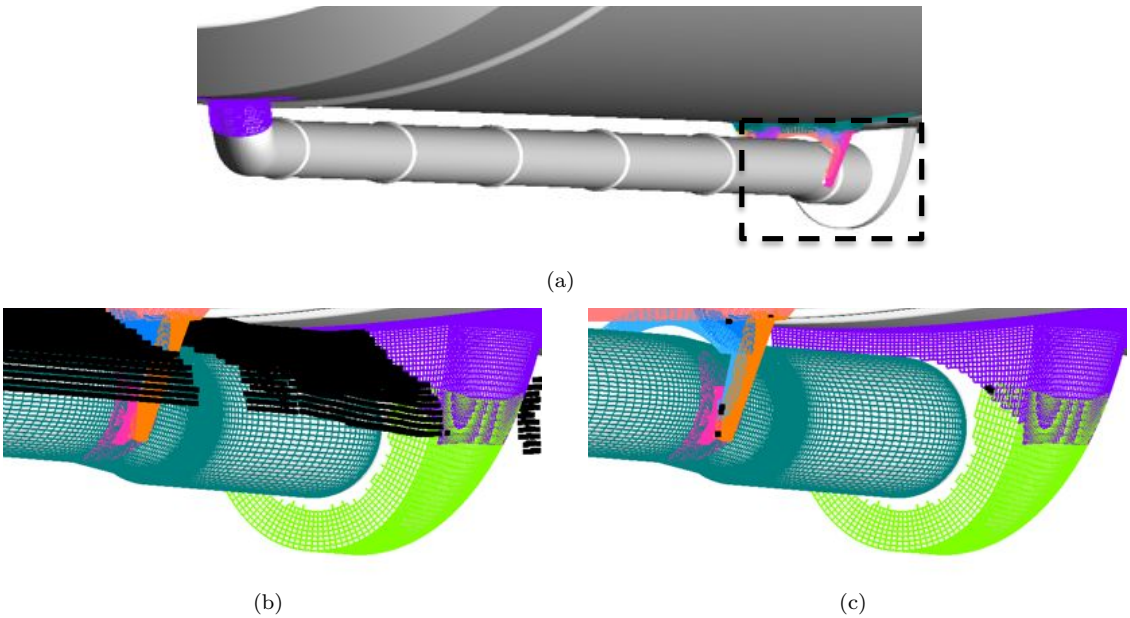
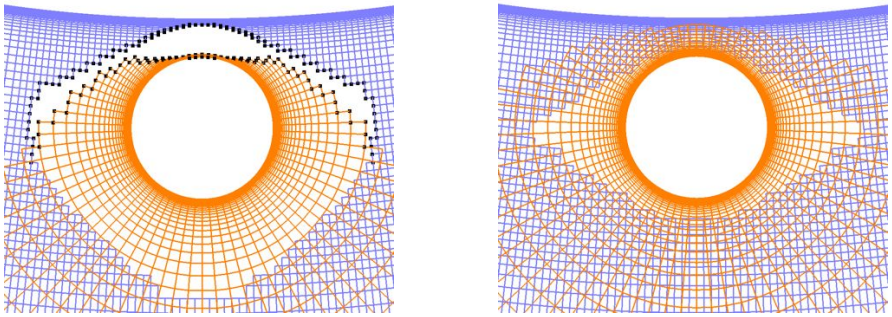


Figure 21. Feedline with bracket test case with orphan points marked by black symbols. (a) Far view with black dashed rectangle marking close-up view of bracket and fairing region. (b) Orphan points from previous scheme with close-up view of bracket and fairing (257000). (c) Orphan points from current scheme with close-up view of bracket and fairing (381).

scheme is able to eliminate most orphan points in these regions. An example of this is illustrated in grid slices through the feedline and the tank as shown in Fig. 22. The previous scheme produced a hole boundary that left insufficient overlap between many neighboring grids while the current scheme is able to create a hole boundary with proper overlap.

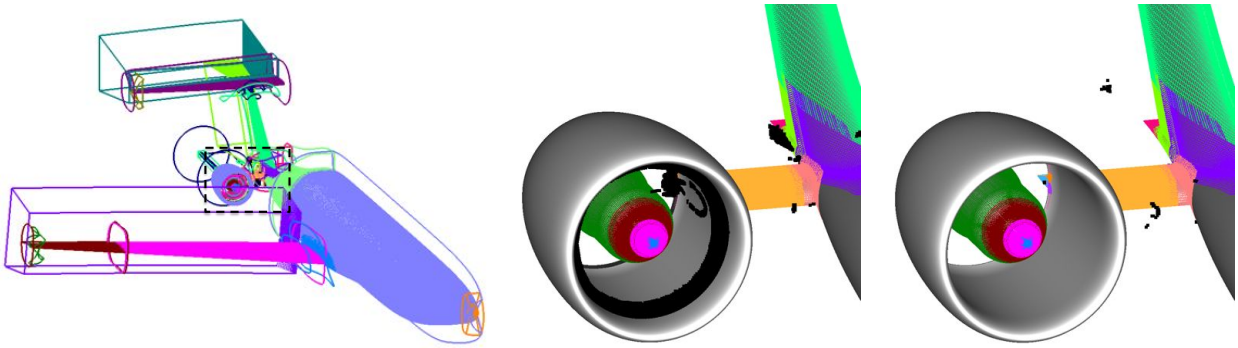


(a) (b)

Figure 22. Grid slice through feedline and bracket test case with orphan points marked by black symbol. (a) Previous scheme. (b) Current scheme.

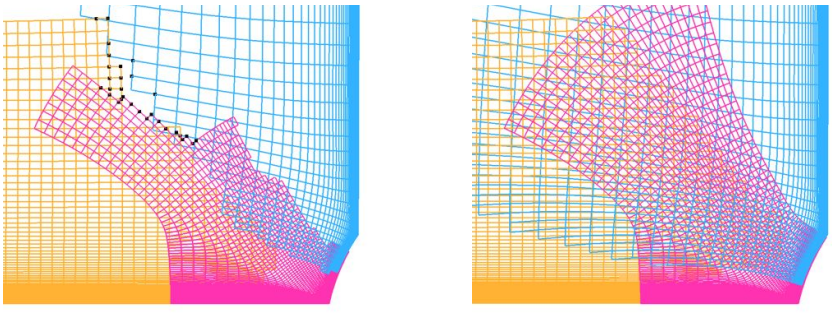
V.F. Double-Bubble Subsonic Airplane

The D-8 Double-Bubble half-body test case consists of a fuselage, subsonic wing, vertical and horizontal tails, and a rear-mounted nacelle connected to the fuselage via a pylon. The system contains 140.5 million points and 36 grids. After the initial hole boundary estimate, 90000 orphan points remained with the previous scheme, while 219 orphan points remained with the new scheme (see Fig. 23). A better estimate of the hole boundaries between the nacelle, hub, and vertical tail by the new scheme is able to significantly reduce the number of orphan points in these regions. Fig. 24 compares the hole boundary obtained using the previous and current schemes in the pylon-vertical-tail region. The new scheme is able to create a hole boundary with proper overlap between the neighboring grids while the previous scheme left gaps between grids.



(a) (b) (c)

Figure 23. D-8 Double-Bubble test case with orphan points marked by black symbols. (a) Far view with black dashed rectangle marking close-up view in vertical tail, pylon, nacelle region. (b) Close-up view of vertical tail, pylon, nacelle region with orphan points from previous scheme (90000). (c) Close-up view of vertical tail, pylon, nacelle region with orphan points from current scheme (219).



(a) (b)

Figure 24. Grid slices through pylon-vertical-tail collar region for D-8 Double-Bubble test case with orphan points marked by black symbols. (a) Previous scheme. (b) Current scheme.

V.G. Summary of Computational Times

The algorithm described in this paper has been implemented into a Fortran 95 software program called Chimera Components Connectivity Program (C3P) which has been moderately parallelized using OpenMP. Table 1 compares the number of orphan points after the hole boundary estimate step using the previous and current schemes. It also lists the wall-clock time in seconds for performing all domain connectivity steps up to the hole boundary estimate using 24 OpenMP threads on an Intel Linux workstation. This includes reading input grid files, determining the minimum hole, estimating the hole boundary offset, searching for donor interpolation stencils, and writing the output files. The table shows that the number of orphan points remaining after the hole boundary estimate for the current scheme is significantly less than that from the previous scheme. For most cases, the number is sufficiently small and the distribution of the orphan points is sufficiently sparse that it is acceptable to proceed to running the flow solver. Also, the wall-clock time required for the current scheme is similar or less than the previous scheme. This demonstrates that the current method is superior since it is less expensive, results in fewer orphan points than the previous scheme at the end of the hole boundary estimate step, and further orphan point removal iterations are not necessary in many cases.

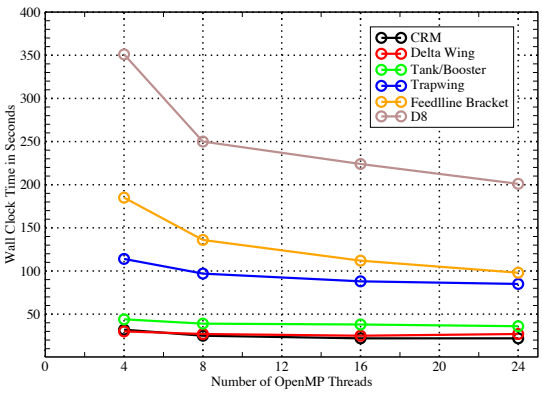


Figure 25. Total wall-clock time versus different number of OpenMP threads for performing I/O, creating minimum hole, estimating hole boundary offset from minimum hole, and finding donor stencils for resulting fringe points.

Table 1. Comparison of wall-clock time (seconds) for various test cases on Linux workstation with Intel Fortran compiler and 24 OpenMP threads for performing I/O, creating minimum hole, estimating hole boundary offset from minimum hole, and finding donor stencils for resulting fringe points.

		Previous		Current	
Test case	**# Points** ($\times 10^6$)	**# Orphans**	**Wall time**	**# Orphans**	**Wall time**
Delta-Wing-Body	32.6	4748	32	539	27
CRM	17.8	2576	22	10	22
Tank-Booster	28.5	102210	37	0	36
High-Lift System	50.6	85000	100	25	85
Feedline-Bracket	82.7	257000	110	381	98
D-8 Podded Nacelle	140.5	90000	344	219	201

Fig. 25 shows the total wall-clock time versus number of OpenMP threads used for performing a connectivity run for the various test cases presented above. The run time includes all steps in the process: file I/O, determining the minimum hole, estimating the hole boundary locations, and searching for donor interpolation stencils. For the small test cases, there is hardly any gain from 8 to 24 threads. For the larger test cases, good speed up is obtained by utilizing more threads, but the gain is far from linear. This is because no effort has been made to ensure work load balance between the threads.

VI. Summary and Conclusions

An automatic and efficient method to position hole boundaries between overlapping grids is presented. The first step involves the determination of an accurate minimum hole using each geometric component's closed surface triangulation. A new robust scheme for constructing such a closed triangulation using cut cells and a painter's algorithm is described. For grid points not close to the geometry surface, the inside/outside test is performed using standard X-ray maps. For grid points close to the surface, a direct ray casting method through the point is deployed to precisely determine the inside/outside status. The second step involves the displacement of the hole boundaries away from the minimum hole using distance rules and donor stencil maps. The current scheme improves upon previous methods by examining local distances with consideration of grid point blanking information. Efficient look-up of approximate closest wall distances and

available donor stencils is accomplished via Cartesian reference maps. Effective rules for grid point blanking have been formulated for near-body non-collar and collar grids, as well as off-body grids. For all of the test cases presented, the current scheme results in significantly fewer orphan points compared to the previous method.[15] Moreover, the computational expense for the current scheme is equal or less than the previous scheme for all the test cases. Even though the main goal of the current work is to reduce the manual effort to perform domain connectivity, it is important to maintain a low computational expense so that the scheme may be efficiently applied to moving-body problems.

At the end of the hole boundary estimate step, the number of remaining orphan points from the current scheme is sufficiently small and the distribution sufficiently sparse that it is feasible to start running the flow solver for most cases. With the previous scheme, a large number of orphan points typically remain after the hole boundary estimate step and orphan removal iterations are needed prior to running the flow solver. Orphan removal iterations can still be activated after application of the current hole boundary estimation scheme if the number of remaining orphan points is unsatisfactory.

Acknowledgements

The authors would like to thank Dr. Jeffrey Housman and Dr. Stuart Rogers from NASA Ames Research Center for insightful discussions and for reviewing this paper. This work has been partially funded by NASA's Space Launch System Program and Advanced Air Transport Technology Project.

References

[1]Gomez, R. J., Vicker, D., Rogers, S. E., Aftosmis, M. J., Chan, W. M., Meakin, R. L. and Murman, S., "STS-107 Investigation Ascent CFD Support," AIAA Paper 2004–2226, 2004.

[2]Ahmad, J. U., Pandya, S. A., Chan, W. M. and Chaderjian, N. M., "Navier-Stokes Simulation of Air-Conditioning Facility of a Large Modern Computer Room," FEDSM 2005-77225, Proceedings of the 2005 ASME Fluids Engineering Division Summer Meeting and Exhibition, Houston, Texas, 2005.

[3]Pandya, S., Onufer, J., Chan, W. and Klopfer, G., "Capsule Abort Recontact Simulation," AIAA Paper 2006–3324, 2006.

[4]Kiris, C. C., Kwak, D., Chan, W. M., Housman, J. A., "High-Fidelity Simulations of Unsteady Flow Through Turbopumps and Flowliners," *Computers & Fluids*, Vol. 37, pp. 536-546, 2008.

[5]Bhagwat, M., Dimanlig, A., Saberi H., Meadowcroft, E., Panda, B. and Strawn, R., "CFD/CSD Coupled Trim Solution of the Dual-Rotor CH-47 Helicopter Including Fuselage Modeling," Proceedings of the American Helicopter Society Aeromechanics Specialist's Conference, San Francisco, 2008.

[6]Kiris, C., Housman, J., Gusman, M., Chan, W. and Kwak, D. , "Time-Accurate Computational Analysis of Ignition Overpressure in the Flame Trench," *Computational Fluid Dynamics Review*, Eds. Hafez, Oshima, Kwak, Publisher: World Scientific, 2010.

[7]Suhs, N. E. and Tramel, R. W., "PEGSUS 4.0 User's Manual," Arnold Engineering Development Center, AEDC-TR-91-8, Arnold AFB, TN, Nov. 1991.

[8]Meakin, R. L., "Object X-rays for Cutting Holes in Composite Overset Structured Grids," AIAA Paper 2001–2537, 2001.

[9]Lee, Y. and Baeder J., "Implicit Hole Cutting - A New Approach to Overset Grid Connectivity," AIAA Paper 2003–4128, 2003.

[10]Rogers, S. E., Suhs, N. E. and Dietz, W. E., "PEGASUS5 : An Automated Pre-Processor for Overset-Grid CFD," *AIAA J.*, Vol. 41, No. 6, pp. 1037-1045, Dec. 2003.

[11]Noack, R. W., "Suggar++: An Improved General Overset Grid Assembly Capability," AIAA Paper 2009–3992, 2009.

[12]Sitaraman, J., Floros, M., Wissink, A. and Potsdam, M., "Parallel Domain Connectivity Algorithm for Unsteady Flow Computations Using Overlapping and Adaptive Grids," *J. Comp. Phys.*, Vol. 229, pp. 4703-4723, March 2010.

[13]Henshaw, W. D., "Ogen: An Overlapping Grid Generator for Overture," Research Report UCRL-MA-132237, Lawrence Livermore National Laboratory, 1998.

[14]Chan, W. M., Kim, N. and Pandya, S. A., "Advances in Domain Connectivity for Overset Grids Using the X-rays Approach," Paper ICCFD7-1201, 7th International Conference on Computational Fluid Dynamics, Big Island, Hawaii (http://www.iccfd.org/iccfd7/assets/pdf/papers/ICCFD7-1201_paper.pdf), 2012.

[15]Chan, W. M., Pandya, S. A. and Rogers, S. E., "Efficient Creation of Overset Grid Hole Boundaries and Effects of Their Locations on Aerodynamic Loads," AIAA Paper 2013-3074, 21st AIAA Computational Fluid Dynamics Conference, San Diego, California, June 24-27, 2013.

[16]Nakahashi, K., Togashi, F. and Sharov, D., "Intergrid-Boundary Definition Method for Overset Unstructured Grid Approach," *AIAA J.*, Vol. 38, No. 11, pp. 2077-2084, 2000.

[17]Parks, S. J., Buning, P. G., Steger, J. L. and Chan, W. M., "Collar Grids for Intersecting Geometric Components Within the Chimera Overlapped Grid Scheme," AIAA Paper 1991–1587, 1991.

[18]Meakin, R. L., "A New Method for Establishing Intergrid Communication Among Systems of Overset Grids," AIAA Paper 1991–1586, 1991.

[19]Kim, N. and Chan, W. M., "Automation of Hole-Cutting for Overset Grids Using the X-rays Approach," AIAA Paper 2011-3052, 20th AIAA Computational Fluid Dynamics Conference, Honolulu, Hawaii, June 27-30, 2011.

[20]Aftosmis, M. J., "Solution Adaptive Cartesian Grid Methods for Aerodynamic Flows with Complex Geometries," Lecture Notes for 28th Computational Fluid Dynamics Lecture Series, von Karman Institute for Fluid Dynamics, Belgium, 1997.