AIAA 2002-3188

THE OVERGRID INTERFACE FOR COMPUTATIONAL SIMULATIONS ON OVERSET GRIDS

William M. Chan
NASA Ames Research Center
Moffett Field, California 94035

32nd AIAA Fluid Dynamics Conference
24-27 June, 2002 / St. Louis, Missouri

AIAA 2002-3188

THE OVERGRID INTERFACE FOR COMPUTATIONAL SIMULATIONS ON OVERSET GRIDS

William M. Chan *
NASA Ames Research Center
Moffett Field, California 94035

Abstract

Computational simulations using overset grids typically involve multiple steps and a variety of software modules. A graphical interface called OVERGRID has been specially designed for such purposes. Data required and created by the different steps include geometry, grids, domain connectivity information and flow solver input parameters. The interface provides a unified environment for the visualization, processing, generation and diagnosis of such data. General modules are available for the manipulation of structured grids and unstructured surface triangulations. Modules more specific for the overset approach include surface curve generators, hyperbolic and algebraic surface grid generators, a hyperbolic volume grid generator, Cartesian box grid generators, and domain connectivity pre-processing tools. An interface provides automatic selection and viewing of flow solver boundary conditions, and various other flow solver inputs. For problems involving multiple components in relative motion, a module is available to build the component/grid relationships and to prescribe and animate the dynamics of the different components.

1. Introduction

In recent years, overset grid methods[1] have been successfully used to compute both steady and unsteady flows for many complex configurations. These computations have contributed to the design and analysis of a variety of aerospace and marine vehicles at government laboratories[2–11] as well as in industry.[12–18] One of the critical elements in such work is the time required to perform a complete configuration analysis. Clearly, trimming the time needed for the entire process will result in significant cost savings. The complete simulation process typically consists of virtual geometry construction (CAD work), geometry processing, surface and volume grid generation, domain connectivity, flow calculation, and solution post-processing. Up until recently, there has been little effort spent on trying to streamline the various steps of the process. Each step requires a number of tools and there is no one place from which all the tools can be accessed.

Two main approaches are being pursued to reduce the overall process time. The first is to develop automated algorithms and software tools that reduce or eliminate the user's input at each step of the process. Examples of recent efforts in grid generation and domain connectivity are given in Refs. 19-21. The second approach to reduce process time is to incorporate all essential and robust tools into a single graphical interface environment (GUI). Clearly, if every step in the process is completely automated, a GUI is not necessary to produce a final result. Since the current state-of-the-art still requires user's inputs at various stages,[22] it is most convenient to work through the different steps in a graphical interface. This paper describes OVERGRID which is one of the very few interfaces available today that has been specially designed for applying the overset approach to complex geometries. A parallel effort has also been developed in the OVERTURE[23] suite of tools.

2. Overview of Interface

2.1. Software Design

The OVERGRID interface belongs to a larger software package called Chimera Grid Tools (CGT). The CGT package consists of about 40 independent grid generation and solution analysis modules that run in batch mode, the OVERGRID graphical interface, a suite of Tcl[24] scripts that can be used for automating overset-grid computations on complex configurations, and several libraries of common routines shared by the various tools (Fig. 1).

OVERGRID serves as a central portal to many of the modules which are called as batch processes, as well as a visualization tool for the working data (geometry and grids). By keeping each module separate from the graphical interface, a sequence of grid operations can be recorded in a script and reproduced easily without manual intervention. OVERGRID offers a script generation capability which automatically

*Computer Scientist, Senior Member AIAA

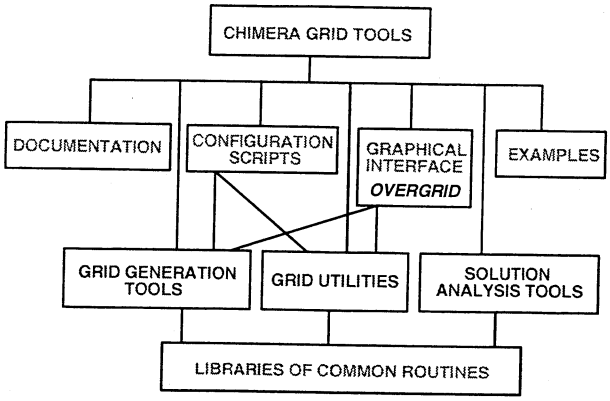

Fig. 1 Structure of Chimera Grid Tools.

records all of the user's actions, excluding those related to visualization transforms. Comment lines are automatically inserted to identify the action types and batch modules used. The script is written in the Tool Command Language Tcl[24] which provides high level functions and modularity. If a simple modification in one of the steps is desired (e.g., alter the number of points in a grid), the script can be edited and re-run in batch mode to reproduce all the steps rapidly.

Another advantage of calling batch mode modules from OVERGRID is the handling of single and double precision data. The input data type is automatically determined by OVERGRID and appropriate single or double precision arrays are dynamically allocated for data storage. Preference settings in OVERGRID are used to specify directories where single and double precision executables of the batch mode modules are located. When the user initiates an action requiring a batch mode module, executables of the appropriate precision are automatically selected to operate on the data in batch mode. This procedure allows OVERGRID to operate on both single and double precision data without the need for code re-compilation.

OVERGRID is primarily written in C and Tcl/Tk.[24] The data input and output routines are written in Fortran for efficient execution speed. Rendering of objects is accomplished with OpenGL calls from the C program, while the interface widgets (e.g., buttons and entries) are built with Tcl/Tk. OpenGL rendering from a Tcl/Tk window is made possible by the Togl widget. The software has been ported to various UNIX workstations and personal computers running Linux, Macintosh OS-X or Windows NT. With its ease of use and conciseness, Tcl/Tk has been found to be enormously valuable in rapidly creating the desired professional look and feel of the interface. Moreover, the quick learning time allows the developers of grid generation codes to easily build the graphical interface themselves. The resulting software tends to be more practical and user-friendly than one built by GUI experts with limited grid generation and computational fluid dynamics experience.

2.2. Data Formats and Main Windows

Both geometry and grid data can be manipulated and created in OVERGRID. For the rest of this paper, the words 'entity' and 'grid' will be used interchangeably to denote any object stored in OVERGRID as part of a surface geometry description (surfaces) or a computational grid (volumes, surfaces, or curves). Surface geometry can be read into OVERGRID as multiple structured panels where each panel is a rectangular array of points in PLOT3D[25] format, or as an unstructured surface triangulation in CART3D (*http://www.nas.nasa.gov/~aftosmis/cart3d/cart3dTriangulations.html*) or FAST[26] formats. For surface triangulations, a tag is associated with each triangle which could be used to store component information. The CART3D format further supports a file containing both grid and scalar function data at the vertices of the triangulation (see description on DIAGNOS module in Section 3). Computational grids are always read and written in PLOT3D format - unformatted or formatted, single or multiple zone, with or without iblanks, single or double precision. The above attributes for an input file are automatically determined by OVERGRID.

Structured entities in OVERGRID are stored as discrete points and a J/K/L convention is employed to denote the index directions in a volume grid. Surface geometry definitions derived from CAD are frequently represented by Non-Uniform Rational B-Spline (NURBS) surfaces or solids. With OVERGRID currently limited to reading structured panels or unstructured triangles, another software package such as GRIDGEN[27] is needed to convert other geometry formats into panel networks or triangles prior to using OVERGRID. Future plans for OVERGRID to read other data formats are given in Section 9.

On execution, OVERGRID will bring up the following four main windows shown in Fig. 2.

(1) The Display window contains a graphical display of the entities currently in memory.

(2) The Controls window contains widgets for setting various display options, resetting views, showing information on the number of volume, surface and curve entities currently in memory.

(3) The Main window contains widgets for input and output of entities, script creation, access to the various modules, and general on-line help. More detailed on-line help is also available for the individual modules.

(4) The Selection window contains widgets for performing selection of entities, blanking/unblanking of entities from view, and deletion of entities. A more direct entity selection mechanism is also available by clicking on the desired entities in the Display window.

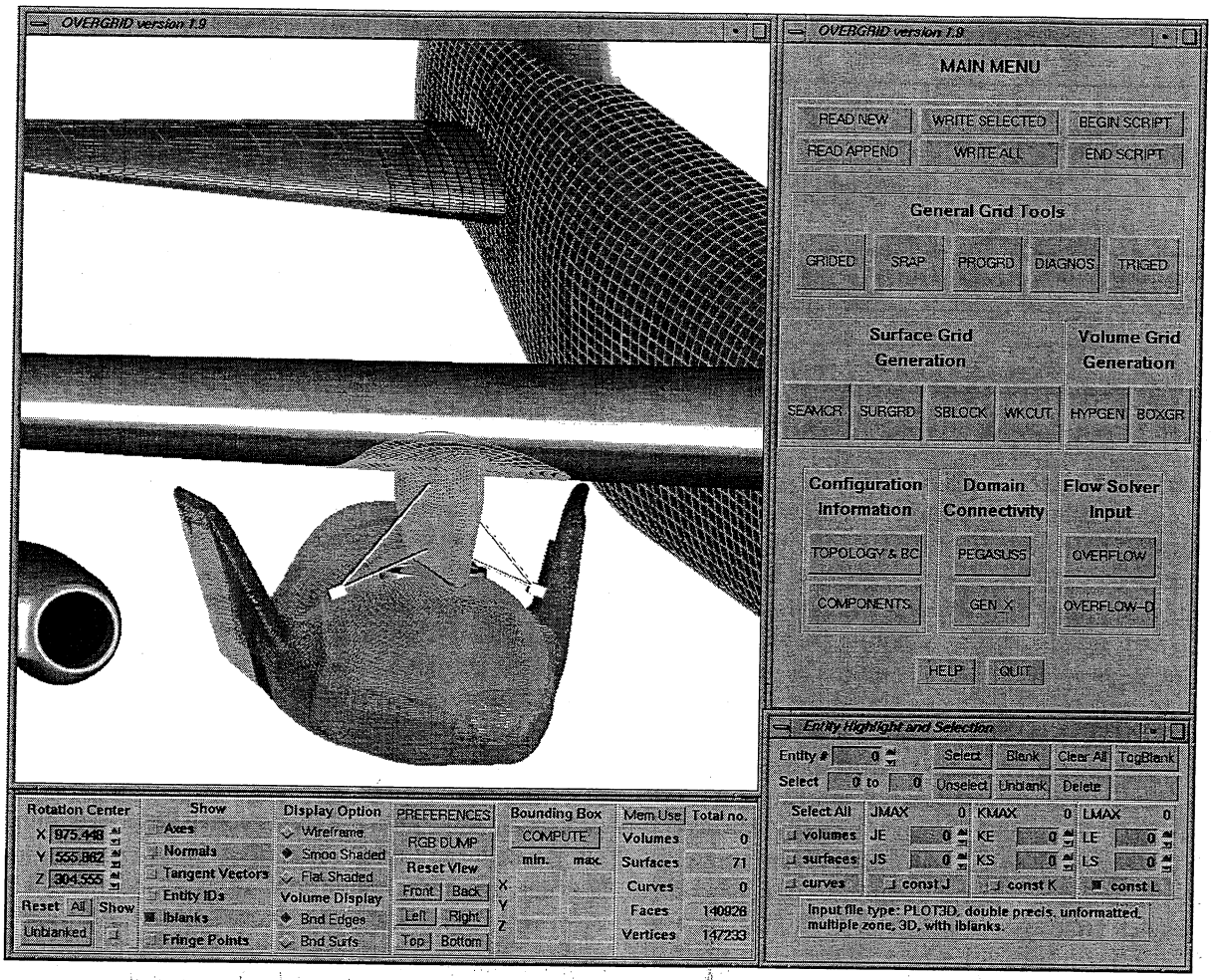

Fig. 2 OVERGRID main windows: upper left - Display, lower left - Controls, upper right - Main, lower right - Selection.

3. General Grid Tools

General grid tools in OVERGRID fall into two classes: diagnostic tools for analyzing entity attributes, and manipulation tools for modifying entities. These are described in more details below.

3.1. Diagnostic Tools

The Controls window offers widgets for toggling the display of entity attributes such as tangent and normal vectors for surfaces and entity identification numbers as illustrated in Fig. 3.

An important attribute of overset grids not typically found in other gridding methods is an 'iblank' value associated with each grid point. The iblank value is used to denote whether the point is a field point, a hole point, or a fringe point. Field points are where the flow equations are solved and the dependent flow variables are computed. Hole points are points that lie outside the flow domain, e.g., points inside the solid surface of an object. The flow equations are not solved at these points. Fringe points are points where the dependent variables are interpolated from stencils in neighboring grids. Such points arise on boundaries of holes and on the outer boundaries of a grid. Points on grid outer boundaries are fringe points only if no flow solver boundary conditions are applied. A fringe point without a valid stencil is called an orphan point.

It is clear that visualization of the iblank value at the grid points is immensely helpful in checking and debugging the results of the domain connectivity process. OVERGRID can be used to display grid planes, where field points are connected by wireframes, hole points are not drawn, fringe points are colored by the grid number of the grid containing the interpolation stencil, and orphan points are highlighted in black against a white display background (Fig. 4). The x, y, z coordinates, grid number, J, K, L indices and iblank value of a vertex can also be interrogated by picking the vertex via a hot key. For surface triangulations, information on specific vertices and faces can also be similarly obtained.

The DIAGNOS module accessible from the Main Menu window allows the user to check various grid quality functions. Wireframe representations of the grid surfaces are colored by the value of the grid quality function. Locations and values of the minimum and maximum are also reported to the user.

For structured surface entities, grid quality func-

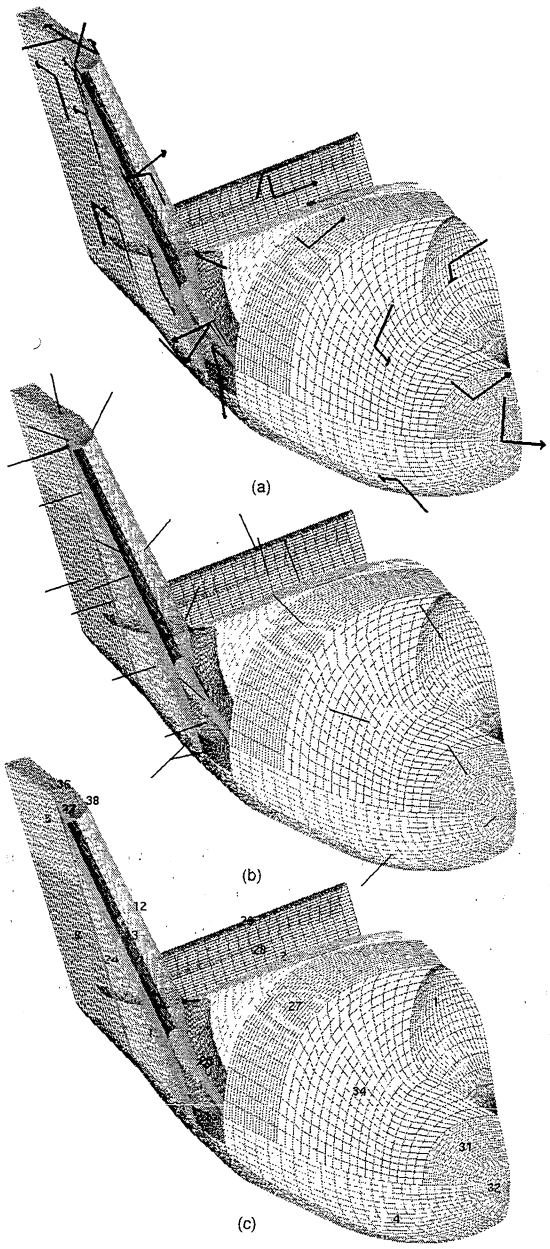

Fig. 3 Display of entity attributes in OVERGRID. (a) Surface tangent vectors (arrow: J direction, line: K direction). (b) Surface normal vectors. (c) Entity identification numbers.

tions include stretching ratios and turning angles in the J and K directions, cell areas for surface grids, and grid induced truncation error estimates (Fig. 5a). The stretching ratios and truncation error estimates can be used to identify where more grid clustering is needed. The cell area diagnostic is useful in checking grid resolution compatibility in regions of grid overlap which is a critical requirement for good inter-grid communication.

For surface triangulations, available grid quality functions include vertex valence (number of triangles connected to a vertex), the minimum and maximum angle at a vertex, and a surface curvature estimate. High quality triangulations should have a vertex valence of around 6 everywhere, and the minimum and maximum angles at a vertex should not have large extrema. The surface curvature estimate could be used to identify surface features but a more robust formula that is independent of grid resolution needs to be determined. Further grid function display is provided for an annotated surface triangulation which contains one or more scalar functions at the vertices, e.g., pressure coefficient, density, and others (Fig. 5b).

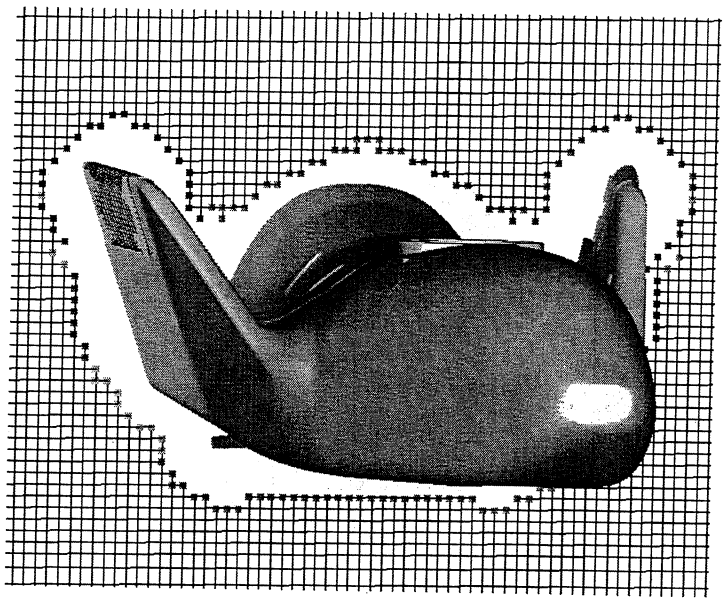

Fig. 4 Iblank display in OVERGRID. Unblanked points are connected by wireframe. Fringe points are rendered with symbols colored by the interpolation stencil donor grid number. Black symbols denote orphan points.

Other features in the DIAGNOS module include a report on the number and percentage of blanked points, the number of orphan points from domain connectivity and the number of negative Jacobians in volume grids. A utility is provided to check the topology (periodic, axis, constant plane) of a surface grid and allow manual resetting of the topology if necessary. For example, a periodic grid with non-coincident start and end planes in the periodic direction can have these planes reset to being coincident.

3.2. Manipulation Tools

Four modules are available in OVERGRID for general grid manipulation.

GRIDED - Structured Grid Editing Tool

The GRIDED module provides the following list of commonly used functions for operations on one or more structured grids.

(1) Swap J and K, K and L, or J and L grid indices.

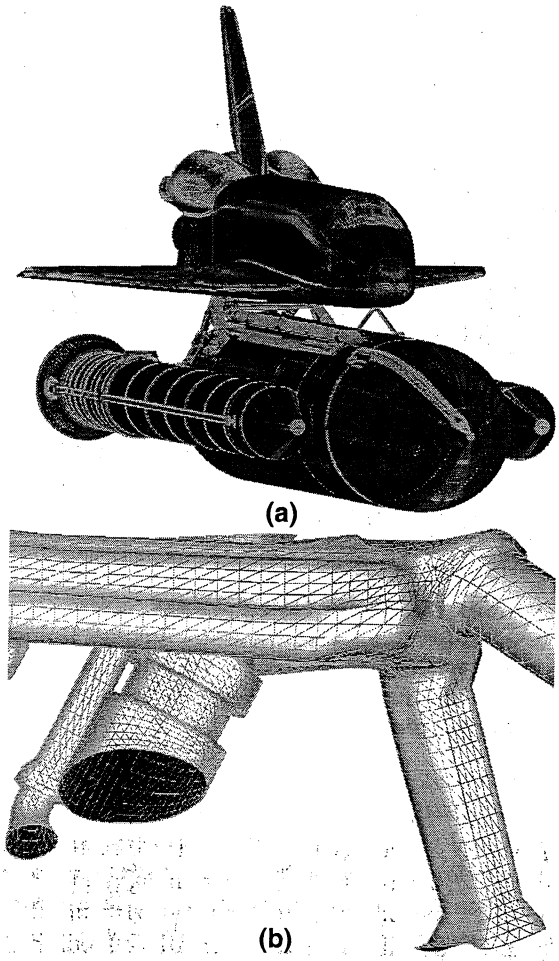

Fig. 5 (a) Grid induced truncation error estimate on the Space Shuttle Launch Vehicle structured overset grid system. (b) Pressure coefficient on a surface triangulation of the aft-attach hardware.

(2) Reverse one or more of J, K and L indices.

(3) Mirror about $X = 0$, $Y = 0$, or $Z = 0$ plane.

(4) Scale or translate.

(5) Rotate about X, Y, or Z Cartesian axes.

(6) Resequence the identification numbers of a collection of grids.

(7) Extract a subset.

(8) Add extra layers of points at $\pm J$, $\pm K$, $\pm L$ boundaries by extrapolating in the tangential, x, y, or z direction using a specified stretching ratio.

(9) Concatenate any two volume, surface or curve grids in the specified direction (J, K, or L).

(10) Split an entity into two along a constant J, K, or L direction at a specified index where the split entities may overlap in one or more points.

(11) Automatically concatenate any number of surface or curve grids in arbitrary relative orientations. A tolerance parameter allows adjacent grids separated by small gaps to be concatenated.

(12) Smooth any subset of a grid in one or more directions in J, K, and L.

(13) Generate surface or volume of revolution from a curve or surface, respectively.

(14) Find intersection curve(s) between an intersector surface and one or more intersectee surface grids.

TRIGED - Surface Triangulation Editing Tool

The TRIGED module provides the following list of commonly used functions for operations on an unstructured surface triangulation.

(1) Swap common edge between two adjacent triangles.

(2) Reverse direction of normal on all triangles.

(3) Mirror about $X = 0$, $Y = 0$, or $Z = 0$ plane.

(4) Scale or translate.

(5) Rotate about X, Y, or Z Cartesian axes.

(6) Remove un-used vertices.

(7) Extract all triangles on one side of a Cartesian cutting plane.

(8) Extract all triangles belonging to a given list of components.

SRAP - Grid Redistribution Tool

The SRAP module is used to redistribute grid points on a structured curve, surface or volume entity. In the case of a surface or volume entity, it is treated as a collection of curves in the J, K or L index direction. Points along each curve are fitted to a cubic spline, and then redistributed based on user input specifications. There is an option to project the new points back onto the original piece-wise linear definition of each curve. Redistribution can occur in one or more segments in each direction where each segment is defined by a start and end index. The user has four input specification options for redistributing points in a segment.

(1) Specify the new number of points and grid spacings at end points (OVERGRID reports the maximum stretching ratio).

(2) Specify the maximum stretching ratio and grid spacings at end points (OVERGRID reports the new number of points needed).

(3) Same as (2) but a maximum grid spacing is also specified.

(4) Specify a uniform spacing (OVERGRID reports the new number of points needed so that the grid spacing in a uniform mesh will not exceed the specified spacing).

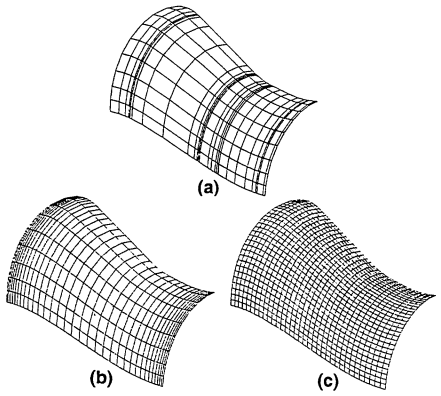

Fig. 6 SRAP functions in OVERGRID. (a) Original surface. (b) Redistribution in one direction with clustering at end points. (c) Redistribution to uniform spacing in both directions.

The input grid spacings can be in absolute units or relative to the total arc length of the segment. The current absolute end grid spacings of the segment are displayed as information for the user. Results of redistribution options (2) and (4) are shown in Fig. 6.

PROGRD - Grid Projection Tool

The PROGRD module is used to project a set of active entities onto a set of reference entities. The reference entities may consist of structured surfaces or unstructured surface triangulations, but not a mixture of the two. Active entities may be structured curves or surfaces. Members of the reference and active entity sets can be graphically selected in OVERGRID. Grid points on the active entities are projected to a bilinear representation of the reference entities. The projection can be performed in the surface normal direction of the reference entities or in the X, Y, or Z directions. After the projection, the maximum distance moved by a point in the active entity set is reported to the user. The user can choose from one of the following three options if an active point falls outside the reference surfaces.

(1) Do not move point.

(2) Project point to closest cell on reference surfaces.

(3) Project point to tangentially extrapolated reference surfaces.

Fig. 7 illustrates the use of PROGRD to model a bump on a blade. A surface grid was originally generated on a clean blade. Subsequent design changes introduced a small bump on the blade. The user decided that it is not critical to model the bump/blade intersection line exactly and that keeping the final configuration in a single grid is more important. PROGRD is used for this task to project the original blade grid (active entity) onto the bump grid (reference entity). Points on the blade grid outside of the bump are undisturbed by selecting option 1 above.

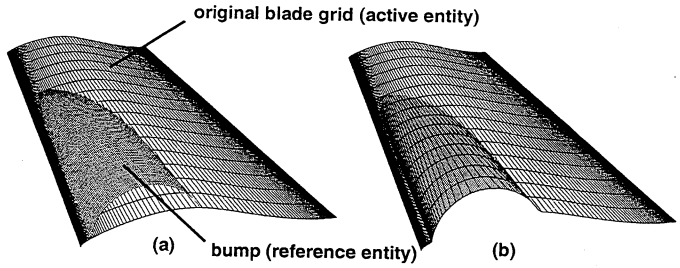

Fig. 7 PROGRD function in OVERGRID. (a) Original reference entity (bump grid) and active entity (blade grid). (b) Active entity (blade grid) after projection.

4. Overset Grid Generation Tools

A current strategy for creating overset grids around a complex configuration consists of the following steps.[22]

(1) A high fidelity definition of the surface geometry is obtained in the form of multiple panel networks (structured patches) or an unstructured surface triangulation. Conversion from other data types such as IGES files or solid models may be necessary.

(2) Surface feature curves and other surface curves are constructed from the surface geometry, e.g., intersection curve between components, sharp surface discontinuities, high surface curvature contours, and open boundaries.

(3) The geometry surface is decomposed into four sided domains. Some are bounded by one or more surface feature curves while others are not bounded by any. Concatenation and splitting of feature curves may be necessary.

(4) Surface grids are generated on the decomposed domains by hyperbolic or algebraic methods.

(5) Body-conforming volume grids in the near field are created from the surface grids by hyperbolic marching.

(6) Off-body Cartesian box grids are generated to enclose the near-field volume grids and to extend the computational domain to the far field.

(7) Hole cutting and interpolation stencil search between the volume grids are performed using domain connectivity software.[21,28–30]

To the frustration of overset grid users for many years, most common grid generation packages do not contain tools that are convenient for accomplishing the above steps. OVERGRID was specifically designed to connect steps 2 to 7, taking full advantage of the freedom allowed by the overset approach to grid generation. Implementation of the various overset related tools in OVERGRID are discussed in the subsections

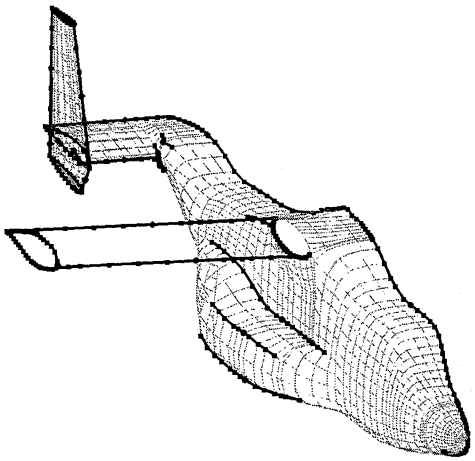

Fig. 8 Automatically generated feature curves for the V-22 fuselage and wing. Surface panels for the wing are not shown.

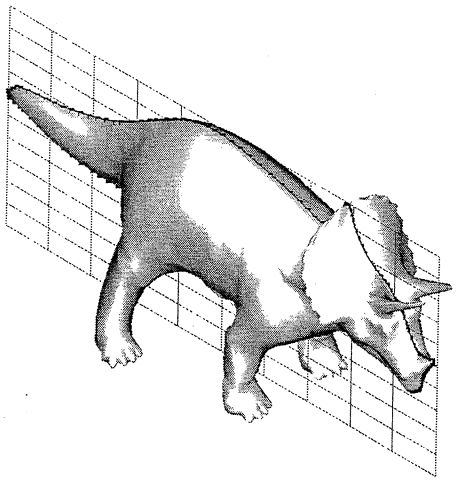

Fig. 9 Surface curve created by intersection with specified Cartesian cutting plane on a triangulation.

below. In each case, a code with the same name that runs in batch mode also exists in the Chimera Grid Tools package.

4.1. Surface Curve Creation Tools

The SEAMCR module is used to create surface curves from a surface definition consisting of multiple panel networks or a surface triangulation. For both types of geometry definitions, two methods are available for making surface curves: automatic extraction of surface feature curves, and intersection with a Cartesian cutting plane. In regions of low surface fidelity, e.g., a curved region represented by just a few panels or triangles, many small curves may be automatically extracted. A simple filter is available in such cases to reduce the number of curves generated. Persisting extraneous curves can then be deleted interactively. Fig. 8 shows feature curves automatically extracted by SEAMCR for the panel network geometry of the V-22 tiltrotor fuselage, wing and tail. Some extraneous seam curves are produced near the junction between the horizontal tail and the fuselage due to poor local surface geometry resolution. Fig. 9 shows a surface curve created by intersecting a given Cartesian plane with a surface triangulation. In general, multiple curves may be generated depending on the location of the cutting plane.

Two additional methods are available for creating surface curves on surface triangulations: connecting two specified vertices along existing triangle edges, and connecting two specified vertices via a direct surface path (Fig. 10). The first method is particularly useful for constructing curves along high curvature contours such as wing leading edges, while the second can be utilized to create a surface curve between two points visible to each other along the surface.

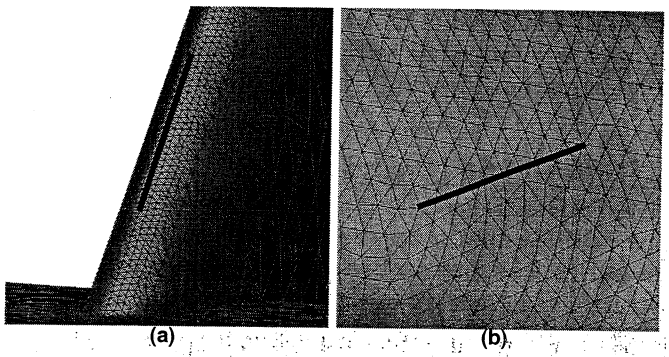

Fig. 10 Surface curve creation between two given vertices on triangulation. (a) By connecting existing edges. (b) By constructing direct path.

4.2. Surface Grid Generation Tools

After appropriate surface curves have been created, the user must determine a decomposition of the surface geometry into domains suitable for surface grid generation. The decomposition process frequently results in domains bounded by only one feature curve. Since neighboring grids are allowed to overlap arbitrarily, the other three boundaries of these domains can be freely floated. Such flexible requirements are ideally suited for hyperbolic surface grid generation. In cases where two or more feature curves bound a domain, algebraic methods are more appropriate. The SURGRD module described in Section 4.2.1 below has been designed for the above gridding strategy.

After creating surface grids in domains bounded by one or more feature curves, there may be regions of the surface that have not yet been covered. The user can define more surface curves and then use SURGRD to create more surface grids to fill the gaps; or use the SBLOCK module described in Section 4.2.2 to fill the gaps with automatically generated overlapping algebraic grids. In certain applications, it is desirable to create a wake cut behind an airfoil shape geometry

to form a C-grid. The WKCUT module discussed in Section 4.2.3 has been designed for this purpose.

4.2.1. *SURGRD - Surface Grid Generator*

The SURGRD[31] module is used to create surface grids from 1, 2, 3, or 4 initial curves, using hyperbolic marching, algebraic marching, or transfinite interpolation (TFI) methods. Surface grids are created to conform to a bilinear representation of the surface geometry defined by a collection of panel networks or a surface triangulation.

For domains bounded by one initial curve, hyperbolic or algebraic marching is used. In most situations, hyperbolic marching is selected to provide orthogonality for the grid lines emanating from the initial curve. However, algebraic marching is sometimes more suitable to create skewed grid lines due to geometric constraints, e.g., marching from a wing/body intersection curve onto a swept wing by following a family of isoparametric lines on the wing surface definition.

For both hyperbolic and algebraic marching, OVERGRID provides widgets to specify the marching distance, initial and/or end spacings, and either the number of points to use or the maximum stretching ratio. Also, for hyperbolic marching, limited control of grid lines emanating from the end points of the initial curve is given via boundary condition specifications, e.g., constant plane, periodic, free floating, and floating along a specified curve. The marching distance can be made to vary for different points along the initial curve. Smoothing parameters are available for adjustment but are rarely needed except in very difficult cases. Fig. 11 shows the SURGRD window with widgets for setting input parameters and the DISPLAY window with several V-22 surface grids created by hyperbolic marching.

For domains bounded by two opposite, two adjacent, three, or four initial curves, transfinite interpolation is used. Additional straight lines are automatically constructed by SURGRD for two-curve and three-curve cases to fill in the missing bounding curves. Fig. 12 shows the automatically simplified SURGRD window for two opposite initial curves and the DISPLAY window with a TFI grid created between the two curves. The SURGRD window simplifies even further for two adjacent, three or four curves since no stretching function needs to be specified.

4.2.2. *SBLOCK - Surface Gap Grid Generator*

Given the surface geometry and a set of surface grids created around the feature curves, the SBLOCK module can be used to automatically generate algebraic surface grids to fill in regions on the surface not already covered. Details of the SBLOCK algorithm and code are found in Ref. 19. The OVERGRID interface is very simple, and provides widgets for the input of the uniform global grid spacing to be used for the algebraic grids. In practice, this module is rarely utilized since it tends to generate a large number of small grids that results in poor flow solver efficiency.

4.2.3. *WKCUT - Wake Cut Surface Grid Generator*

The WKCUT module is used to generate and add a wake cut to the surface grid of an airfoil shape such as a wing, flap, slat, fin, or pylon to form a C-grid. Default parameters are automatically set for the streamwise extent of the wake, the number of points used and the grid spacing. For more difficult cases such as a high/low wing and fuselage, the user can select parameters to modify the deflection angle of the wake cut such that the cut intersects the fuselage. This intersection requirement is needed for the construction of a collar grid[32] in the wing/fuselage junction.

4.3. Volume Grid Generation Tools

After creating a set of overlapping surface grids, body-conforming volume grids have to be generated. Again, the overset approach only requires neighboring volume grids to overlap. This allows the specification of just the surface grid while the other five faces of the volume domain are free to float. A hyperbolic marching scheme is particularly suited for this type of grid. Significant user and computer time savings over iterative elliptic methods are possible using a marching scheme. Only one instead of six faces needs to be defined. Moreover, hyperbolic methods naturally provide the tight clustering needed near the surface for viscous computations, as well as high quality nearly orthogonal grids everywhere. The HYPGEN[33,34] module described in Section 4.3.1 has been designed to perform hyperbolic volume grid generation.

Body-conforming volume grids are usually grown a constant distance from the body, typically a fraction of the body length so that the outer boundaries of the volume grids are well clear of wall-bounded viscous effects. The BOXGR module described in Section 4.3.2 can then be used to automatically create stretched Cartesian box grids around the near field volume grids and extend the computational domain to the far field. An alternative to the BOXGR approach is to use multiple layers of adaptive off-body Cartesian grids that are automatically generated by the OVERFLOW-D[35,36] module as discussed in Section 4.3.3.

4.3.1. *HYPGEN - Hyperbolic Field Grid Generator*

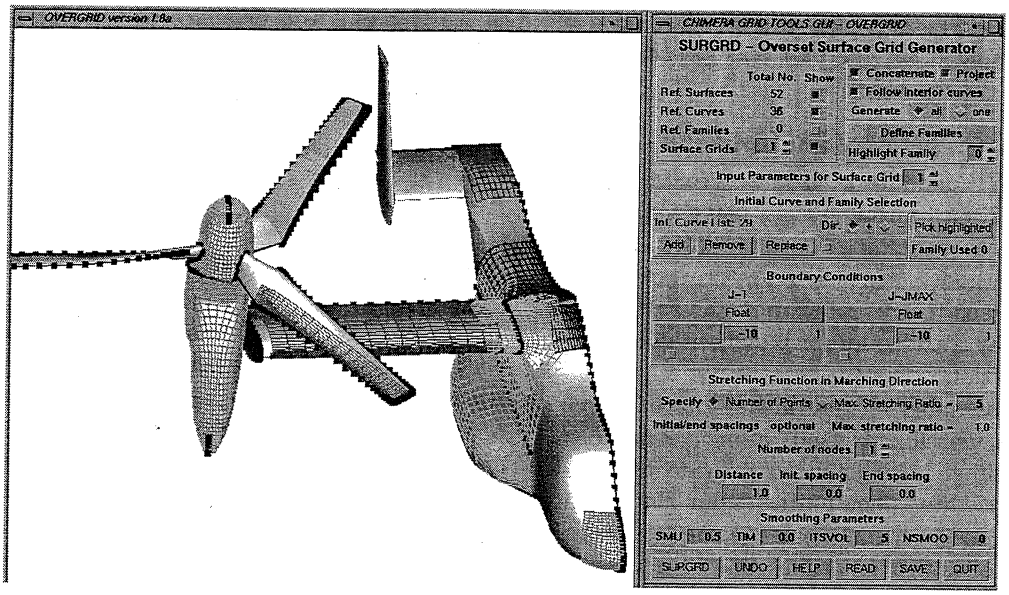

Fig. 11 Hyperbolic surface grids created by SURGRD for part of the V-22. Points on initial curves are indicated by symbols.

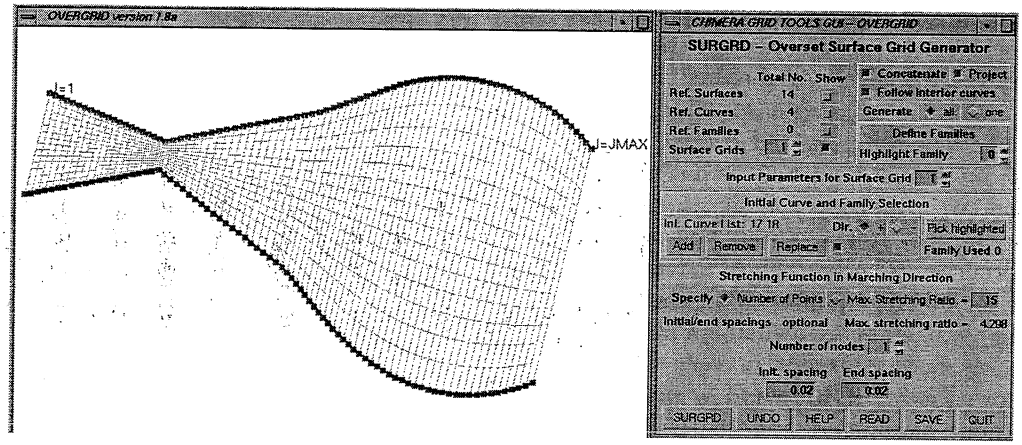

Fig. 12 A surface grid created by transfinite interpolation between two opposite curves in SURGRD.

The HYPGEN[33,34] module is utilized to create a three-dimensional volume grid by marching from a surface grid. For two-dimensional cases, a field grid on a plane is generated by marching from a curve. The marching distance, initial and/or end grid spacings and the number of points to use in the marching direction are specified by the user. Boundary conditions are automatically selected by OVERGRID based on the topology of the given surface grid, e.g., periodicity, singular axis point, constant plane, and others. A free floating boundary is selected if no special topology is detected. The user also has the choice to enforce different boundary conditions and to adjust smoothing parameters if needed for difficult cases such as highly acute concave corners. Since all near-body volume grids employ the same stretching function in the normal direction, OVERGRID allows the user to generate the volume grids for a group of surface grids using the same parameters with a single click of a button. As each volume grid is created, a table is displayed which shows if any negative Jacobians are found in each grid.

Fig. 13 shows the HYPGEN window with widgets for setting input parameters, and the DISPLAY window with several volume grids created by hyperbolic marching. For all volume grids shown here, a geometric stretching is used in the normal direction with the same marching distance, initial spacing, number of points and smoothing parameters. Default boundary conditions are employed for all grids (free floating for the cases shown). The eight volume grids shown contain a total of about 460000 points. It takes less than 15 seconds of wall clock time (user's labor time plus CPU time) to generate all eight grids on a Silicon Graphics R12000 workstation.

4.3.2. *BOXGR - Stretched Cartesian Grid Generator*

The BOXGR module is used to create a Cartesian box grid consisting of an interior core with uniform

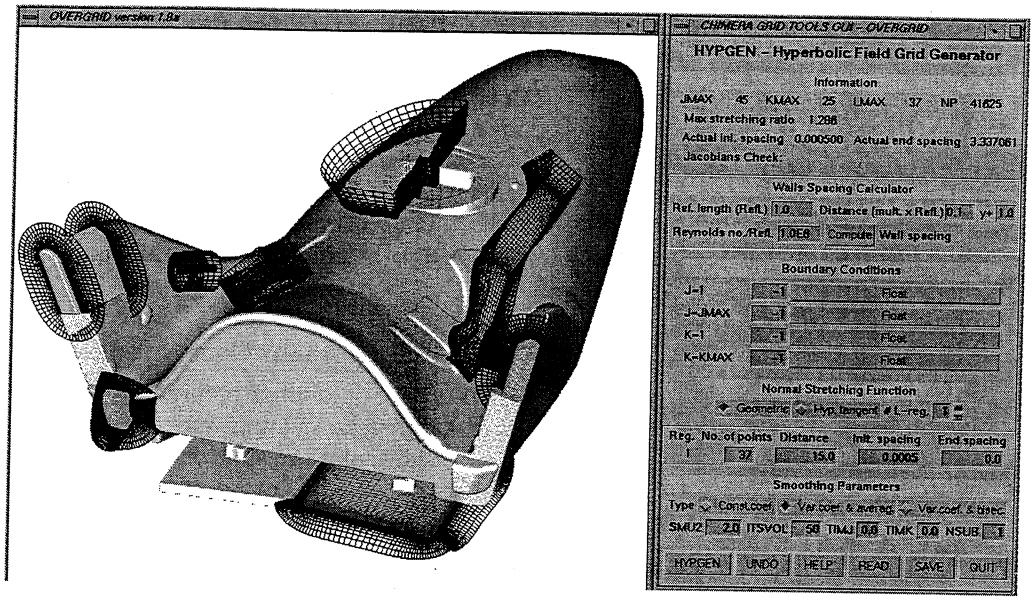

Fig. 13 Samples of hyperbolic volume grids for the X-38 Model-G created under the HYPGEN interface.

spacing, and with optional stretched outer layers in the plus and minus X, Y, and Z directions. In BOXGR's automatic mode, the user selects one or more near-field volume grids, and BOXGR automatically creates a Cartesian box grid with uniform spacing that completely encloses all the selected volume grids. The uniform spacing is automatically chosen to match the average grid spacing at the outer boundaries of the volume grids, thus providing good inter-grid communication. In BOXGR's manual mode, coordinates of the corners of the interior core can be explicitly prescribed. In both automatic and manual mode, extra stretched layers in all directions can easily be added by specifying a distance and a stretching ratio (Fig. 14a).

The computational domain can be extended to the far field via the stretched layers or ellipsoidal shell option in BOXGR. In the latter, BOXGR automatically generates an ellipsoidal surface grid that fits one or more cells inside the outer boundaries of a given stretched Cartesian box grid, thus providing proper overlap; and the grid spacings in the tangential direction are made to match those of the Cartesian grid. A hyperbolic volume grid can then be grown from this surface to extend the computational domain to the far field. The ellipsoid topology allows a uniform expansion of the grid, resulting in grid point savings over the stretched Cartesian box option which keeps the tight uniform core spacing out to the far field (Fig. 14b). However, a drawback of the ellipsoid topology is the presence of the singular polar axes which could reduce flow solver stability for time accurate computations.

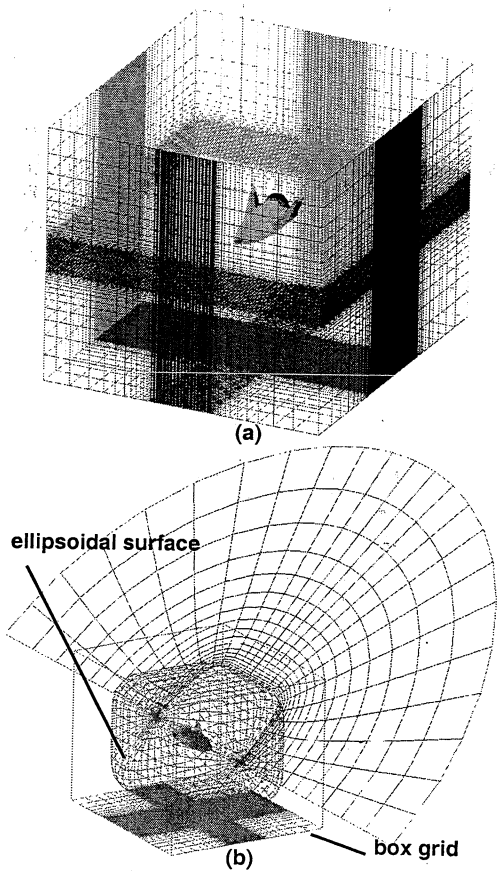

Fig. 14 Automatically generated grids by the BOXGR module. (a) Stretched Cartesian box grid with uniform core. (b) Far field ellipsoidal surface and volume grid.

4.3.3. *OVERFLOW-D Cartesian Grid Generator*

The OVERFLOW-D[35,36] program provides an option to automatically create off-body Cartesian grids.

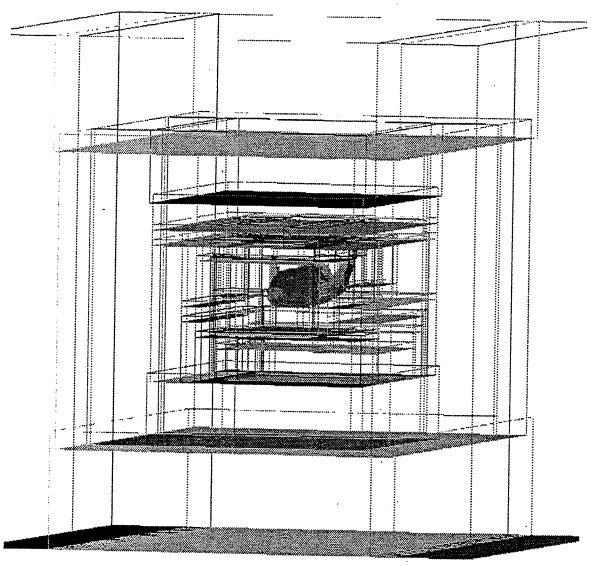

Fig. 15 Off-body Cartesian grids for the X-38 created by the OVERFLOW-D module.

Starting from a collection of near-body curvilinear volume grids, off-body Cartesian grids of successive levels of refinements are created. The finest level (level 1) is closest to the body and encloses all near-body grids with a grid spacing that matches the outer boundary spacings of the near-body grids. Progressively coarser levels of grids are generated based on proximity to the body. OVERGRID provides an interface for the specification of the level 1 grid spacing and special control planes at the outer boundaries of the Cartesian domain, e.g., a symmetry plane or a ground plane. OVERFLOW-D can be called from OVERGRID in the off-body Cartesian grid generation mode and the results displayed in the graphics window (Fig. 15).

5. Boundary Conditions Selection

After creating the volume grids, it is advantageous to determine the boundary conditions to be applied to each grid prior to performing domain connectivity and flow solution computation. This is because boundary condition information constitutes a significant part of the input required for both of these procedures. OVERGRID offers a module to automatically select boundary conditions for each grid based on grid topology (periodic, singular axis, constant plane, symmetry plane, wake cut). Furthermore, an intelligent estimate on the locations of viscous and inviscid walls is made based on heuristic rules. A menu is also available in the interface for manual override of any automatically selected inputs (Fig. 16). Additionally, a fast visual check on the automatically selected values is available in the DISPLAY window where grid surfaces can be colored by boundary condition type. The resulting set of boundary conditions for each grid can be written to input files for domain connectivity tools and flow solvers such as OVERFLOW.[37] Moreover, existing input files for such tools where the boundary conditions were selected manually can be read into OVERGRID and their validity checked on the graphical display. Overall, significant time savings can be achieved with this interface in the laborious and error prone process of boundary conditions specifications for grid systems involving a large number of grids.

6. Domain Connectivity Tools

The task of domain connectivity involves surface projection, hole cutting and fringe points interpolation. Interpolation stencils are adjusted in surface projection to remove the effect of slight mismatches between overlapping surface grids due to discretization. Hole cutting is the identification and tagging of field grid points from one grid that lie inside the solid boundary of a body. Fringe points interpolation is needed to provide communication between neighboring grids. Such points arise at the boundaries of a grid that overlaps with another grid (outer boundary points), and at the boundaries of holes left by the hole cutting process (hole boundary points). Interpolation stencils from neighboring grids are sought for these points which require inter-grid communication.

OVERGRID provides connection to two domain connectivity programs: PEGASUS5[21] and OVERFLOW-D.[28,36] Both programs utilize the complete list of boundary conditions for each grid to determine the locations of fringe points. Such information is supplied by OVERGRID's boundary conditions selection module (Section 5).

PEGASUS5's approach to hole cutting relies on an automatic capability plus additional manual hole cuts which are sometimes needed for difficult cases. The automatic hole cutting surfaces can be derived directly from the list of solid wall boundary conditions that are set for each grid. After writing an OVERFLOW[37] input file from OVERGRID, one of the PEGASUS5 tools will take this file and generate a PEGASUS5 input file and also create the directory structure necessary to start the run.

OVERFLOW-D's approach to hole cutting is based on object X-rays.[28] Hole cutters have to be user-defined, where a hole cutter is a collection of surfaces that forms a closed volume. Object X-rays are then built from the hole cutters. Each object X-ray consists of a set of pierce points on the hole cutter's surface. The list of grids that are cut by each hole cutter also has to be specified. OVERGRID provides a graphical utility for specifying the hole cutters and generating the object X-rays under the GEN_X utility. Resulting object X-rays are visualized by sweeping through constant X or Y planes with the surface pierce points displayed in symbols (Fig. 17). A menu is also available to create the input file and execute the domain connec-

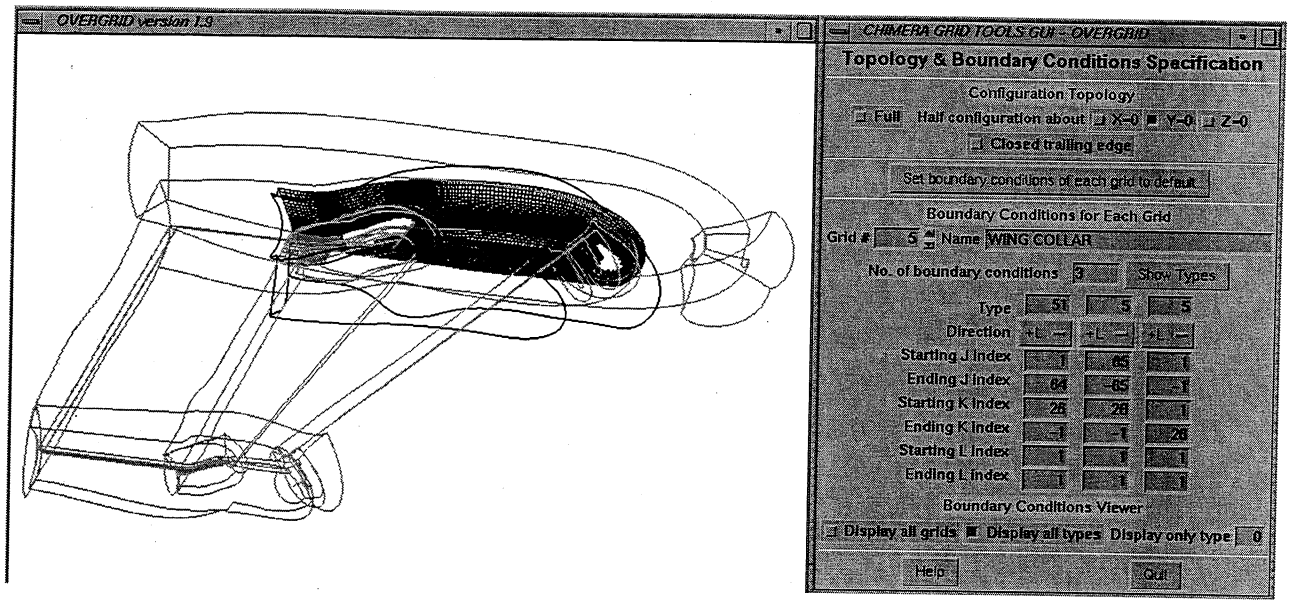

Fig. 16 Left panel: boundary conditions display with outline of volume grids. Surfaces with prescribed boundary conditions are colored by type. Boundary conditions for only one grid are shown here. Right panel: boundary conditions specification widgets.

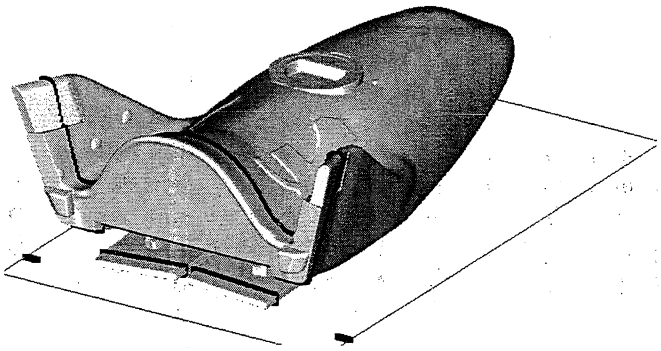

Fig. 17 Object X-rays for the X-38 Model-G created by the GEN_X module. Pierce points on geometry using constant-X cutting plane are shown by black symbols.

tivity module in OVERFLOW-D. After execution, the resulting grid file with connectivity iblanks is automatically read into OVERGRID. Fringe points that failed to find interpolation stencils (orphan points) are automatically displayed. The number of orphan points for each grid and the overall total are reported. With this interactive tool, the user can then make appropriate modifications to the volume grids and domain connectivity inputs and re-run OVERFLOW-D to eliminate the orphan points.

7. Flow Solver Input Preparation

OVERFLOW[37] and OVERFLOW-D[36] are compressible Navier-Stokes flow solvers for overset grids that are widely used for both research and production purposes. Many options have been implemented to allow for a variety of flow physics, numerical methods, geometrical complexities, and computer architectures. The general nature of the code allows a potentially complicated input file. For example, about 30 parameters can be specified globally and about 70 parameters can be specified locally for each grid in OVERFLOW. A namelist input format is utilized where defaults are automatically set for all parameters and only the parameters that apply to a particular problem need to be prescribed. However, it is still a formidable task for a novice user to understand exactly which parameters, out of the long specifiable list, are required for a particular problem.

OVERGRID offers a simplified interface for the selection and entry of the most commonly used options and inputs such as flow conditions, numerical methods, time steps, and turbulence models (Fig. 18). One important feature is the automatic selection of boundary conditions with optional manual override under the boundary conditions module described in Section 5. With the OVERGRID interface, the namelist input preparation time for the OVERFLOW and OVERFLOW-D flow solvers can be reduced by a factor of at least 2 to 5 for most complex problems.

8. Multiple-Component Dynamics

Steady flow computations on complex geometries are now routinely performed in both research and production environments. Although more expensive, unsteady flow computations are also becoming more common as a result of advances in computer technology. However, computations involving multiple components in relative dynamic motion remain scarce. This is due to three main reasons. First, this class of problems can only be computed in time-accurate mode, and thus are inherently costly. Second, domain connectivities be-

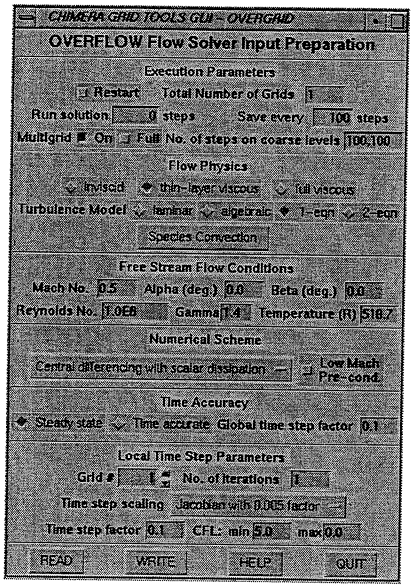

Fig. 18 OVERFLOW flow solver input options interface.

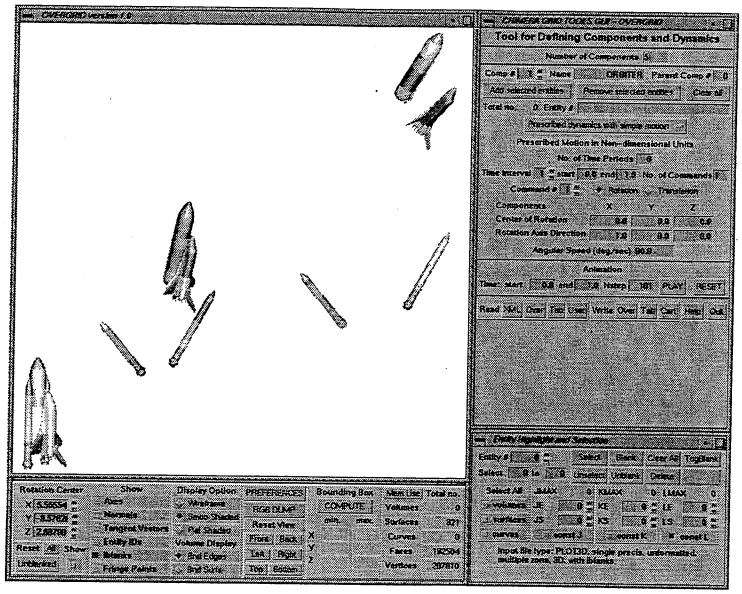

Fig. 19 Component dynamics animation module in OVERGRID showing a fictitious Space Shuttle Launch Vehicle separation scenario.

tween grids need to re-computed at every time step. Third, the dynamics involved can be extremely complex, and there is no simple method for the user to specify the motion. An interface is designed in OVERGRID to help alleviate the third problem.

Components can be defined in OVERGRID where each component may be stationary or move relative to an inertial frame. The interface allows the user to define the grids that belong to each component. Also, each component is allowed to move in one of two ways: (1) six-degrees-of-freedom motion under the influence of aerodynamic forces and moments, gravity, and any externally applied loads, or (2) prescribed motion.

In case (1), the flow solver computes the resultant aerodynamic forces and moments acting on each component and the dynamics is determined by solving the equations of motion. The input parameters required are component properties such as mass, weight, moments of inertia, and externally applied loads. Widgets are provided in OVERGRID for the entry of these parameters. This information is then communicated to the flow solver via a namelist input file.

In case (2), the motion of each component needs to be prescribed by the user and the information transmitted to the flow solver. OVERGRID provides an interface for specifying a simple class of motions consisting of a sequence of constant translational velocities and angular speeds for each component. The programmed motion can then be viewed inside OVERGRID via an animation of the defined components (Fig. 19). Work is in progress to represent this information in a data file that can be read by different flow solvers (also see Section 9). Currently, the OVERFLOW-D solver requires a user to supply a subroutine containing the prescribed motion information. While this allows the most general types of motion possible, it is difficult for a novice user to get started even for very simple cases. The ability to prescribe a wide class of motions via a data file will significantly reduce the user work required for problem setup and allow simulations involving multiple components in relative motion to be more common place.

9. Conclusions and Future Plans

A unified and practical graphical user interface for efficient and streamlined handling of the overset grid computational simulation process is presented. The goal is to reduce the manual effort needed in the steps prior to running the flow solver. Procedures covered include geometry processing, surface and volume grid generation, boundary conditions specification, domain connectivity, diagnostics, flow solver input preparation, and treatment of multiple component dynamics.

The OVERGRID project started a few years ago when there was no satisfactory interface that contained all the tools necessary for efficient overset grid generation. Today, it's capabilities have expanded beyond grid generation to the above list, and remains one of the few options available to overset grid users with complex geometry applications. Typically, a three to four-fold reduction in manual process time is achieved using the software. Further significant time savings are obtained using the scripting capability. The software is being utilized by numerous U.S. organizations in various government laboratories, industry groups, and universities under a non-disclosure agreement.

The future development plan for OVERGRID consists of a long list. One of the major items is to implement the capability to create surface grids based on CAD geometry definitions such as trimmed NURBS

surfaces and solid models. Another area that requires work is the automatic coverage of the surface domain from multiple hyperbolically generated surface grids. As domain connectivity tools evolve, OVERGRID will keep hooks to the most current and robust modules. Input generation for flow solvers other than OVERFLOW can be easily added. Work is in progress to design a general framework for multiple component hierarchy and relative motion dynamics expanding the simplified approach described in this paper. Communications to the flow solver will most likely be made via an XML file. A flexible format will be designed to allow interfacing with flow solvers from both structured and unstructured grid approaches.

Acknowledgements

The development of OVERGRID has vastly benefitted from the expertise of the author's colleagues at NASA. Mouse transforms and direct screen picking of entities were borrowed from routines implemented by Steven Nash. Insightful ideas on graphics and GUIs were shared by Dr. Shishir Pandya. Also, the author is grateful to the many brave testers throughout OVERGRID's evolution, especially Drs. Robert Meakin, Earl Duque, and Pieter Buning for their patience, encouragement and constructive comments. Finally, the software would not be as practical today without the feedback from numerous users. The X-38 geometry and grids shown in this paper have been kindly provided by James Greathouse and Reynaldo Gomez of NASA Johnson Space Center. This work was funded by an element of the DOD High Performance Computing Modernization Program known as CHSSI (Common High Performance Scalable Software Initiative) from 5/1996 to 9/2000, the NASA High Performance Computing and Communications Program from 9/2000 to 9/2001, and the NASA Computer, Information and Communication Technology Program from 10/2001 to the present.

References

[1] Steger, J. L., Dougherty, F. C. and Benek, J. A., "A Chimera Grid Scheme," *Advances in Grid Generation*, K. N. Ghia and U. Ghia, eds., ASME FED-Vol. 5, June, 1983.

[2] Smith, M., Chawla K., and Van Dalsem, W. R., "Numerical Simulation of a Complete STOVL Aircraft in Ground Effect," AIAA Paper 91-3293, 9th AIAA Applied Aerodynamics Conference, September, 1991.

[3] Atwood, C. A. and Van Dalsem, W. R., "Flowfield Simulation about the SOFIA Airborne Observatory," AIAA Paper 92-0656, January, 1992.

[4] Meakin, R., "Moving Body Overset Grid Methods for Complete Aircraft Tiltrotor Simulations," AIAA Paper 93-3350, *Proceedings of the 11th AIAA Computational Fluid Dynamics Conference*, Orlando, Florida, July, 1993.

[5] Gomez, R. J. and Ma, E. C., "Validation of a Large Scale Chimera Grid System for the Space Shuttle Launch Vehicle," AIAA Paper 94-1859, *Proceedings of the 12th AIAA Applied Aerodynamics Conference,* Colorado Springs, Colorado, June, 1994.

[6] Slotnick, J. P., Kandula, M. and Buning, P. G., "Navier-Stokes Simulation of the Space Shuttle Launch Vehicle Flight Transonic Flowfield Using a Large Scale Chimera Grid System," AIAA Paper 94-1860, *Proceedings of the 12th AIAA Applied Aerodynamics Conference,* Colorado Springs, Colorado, June, 1994.

[7] Atwood, C. A., "Computation of a Controlled Store Separation from a Cavity," *J. of Aircraft*, Vol. 32, No. 4, 1995, pp. 846–852.

[8] Duque, E. P. N, Berry J. D., Budge A. M. and Dimanlig A. C. B., "A Comparison of Computed and Experimental Flowfields of the RAH-66 Helicopter," Proceedings of the 1995 American Helicopter Society Aeromechanics Specialist Meeting, Fairfield County, Connecticut, 1995.

[9] Gee, K., Murman, S. M. and Schiff, L. B., "Computation of F/A-18 Tail Buffet," *J. of Aircraft*, Vol. 33, No. 6, 1996, pp. 1181–1189.

[10] Srinivasan, G. R. and Klotz, S. P., "Features of Cavity Flow and Acoustics of the Stratospheric Observatory For Infrared Astronomy," *Proceedings of the ASME Fluids Engineering Conference,* Vancouver, British Columbia, Canada, June, 1997.

[11] Meakin, R. L., "Unsteady Aerodynamic Simulation of Static and Moving Bodies Using Scalable Computers," AIAA Paper 99-3302, *Proceedings of the 14th AIAA Computational Fluid Dynamics Conference*, Norfolk, Virginia, June, 1999.

[12] Gea, L. M., Halsey, N. D., Intemann, G. A. and Buning, P. G., "Applications of the 3-D Navier-Stokes Code OVERFLOW for Analyzing Propulsion-Airframe Integration Related Issues on Subsonic Transports," ICAS Paper 94-3.7.4, 19th Congress of the International Council of the Aeronautical Sciences, September, 1994.

[13] Wai, J., Herling, W. W. and Muilenburg, D. A., "Analysis of a Joined-Wing Configuration," AIAA Paper 94-0657, January, 1994.

[14] Rogers, S. E., Cao, H. V. and Su, T. Y., "Grid Generation for Complex High-Lift Configurations," AIAA Paper 98-3011, 29th AIAA Fluid Dynamics Conference, Albuquerque, New Mexico, June, 1998.

[15] Slotnick, J. P., An, M. Y., Mysko, S. J., Yeh, D. T., Rogers, S. E., Roth, K., Nash, S. M. and Baker, M. D., "Navier-Stokes Analysis of a High Wing Transport High-Lift Configuration with Externally Blown

Flaps," AIAA Paper 2000-4219, 18th AIAA Applied Aerodynamics Conference, Denver, Colorado, August, 2000.

[16] Rogers, S. E., Roth, K., Cao, H. V., Slotnick, J. P., Whitlock, M., Nash, S. M. and Baker, M. D., "Computation of Viscous Flow for a Boeing 777 Aircraft in Landing Configuration," AIAA Paper 2000-4221, 18th AIAA Applied Aerodynamics Conference, Denver, Colorado, August, 2000.

[17] Brand, A., Peryea, M., Wood, T., and Meakin, R., "Flowfield and Download Measurements and Computation of a Tiltrotor in Hover," American Helicopter Society - International 57th Annual Forum, Washington, D.C., May, 2001.

[18] Naik, D. A., and Om, D., "Assessment of the OVERFLOW Navier-Stokes Code for Various Airplane Components," SAE Paper Number 2001-01-2976, World Aviation Conference, Seattle , Washington, September, 2001.

[19] Chan, W. M. and Meakin, R. L., "Advances Towards Automatic Surface Domain Decomposition and Grid Generation for Overset Grids," AIAA Paper 97-1979, *Proceedings of the AIAA 13th Computational Fluid Dynamics Conference,* Snowmass, Colorado, June, 1997.

[20] Chan, W. M. and Gomez, R. J., "Advances in Automatic Overset Grid Generation Around Surface Discontinuities," AIAA Paper 99-3303, *Proceedings of the AIAA 14th Computational Fluid Dynamics Conference,* Norfolk, Virginia, June, 1999.

[21] Suhs, N. E., Rogers, S. E., and Dietz, W. E., "PEGASUS 5: An Automated Pre-processor for Overset-Grid CFD," AIAA Paper 2002-3186, 32nd AIAA Fluid Dynamics Conference, St. Louis, Missouri, June, 2002.

[22] Chan, W. M., Gomez, R. J., Rogers, S. E., and Buning, P. G., "Best Practices in Overset Grid Generation," AIAA Paper 2002-3191, 32nd AIAA Fluid Dynamics Conference, St. Louis, Missouri, June, 2002.

[23] Brown, D. L., Henshaw, W. D. and Quinlan, D. J.,"Overture: Object-Oriented Tools for Overset Grid Applications," AIAA Paper 1999-3130, 17th AIAA Applied Aerodynamics Conference, Norfolk, Virginia, 1999.

[24] Welch, B., *Practical Programming in Tcl and Tk,* 3rd ed., Prentice Hall, 1999.

[25] Walatka, P. P., Buning, P. G., Pierce, L. and Elson, P. A., "PLOT3D User's Manual," NASA TM 101067, 1990.

[26] Walatka, P. P., Clucas, J., McCabe, R. K., Plessel, T. and Potter, R, "FAST User Guide," RND-93-010, NASA Ames Research Center, 1994.

[27] Chawner, J. R. and Steinbrenner, J. P., "Automatic Structured Grid Generation Using GRIDGEN (Some Restrictions Apply)," *Proceedings of NASA Workshop on Surface Modeling, Grid Generation, and Related Issues in Computational Fluid Dynamics (CFD) Solutions,* NASA CP 3291, May, 1995.

[28] Meakin, R. L., "Object X-rays for Cutting Holes in Composite Overset Structured Grids," AIAA Paper 2001-2537, 15th AIAA Computational Fluid Dynamics Conference, Anaheim, California, June, 2001.

[29] Maple, R. C. and Belk, D. M., "Automated Set Up of Blocked, Patched and Embedded Grids in the Beggar Flow Solver," *Numerical Grid Generation in Computational Fluid Dynamics and Related Fields,* Ed. N. P. Weatherill et al., Pine Ridge Press, pp. 305–314, 1994.

[30] Henshaw, W. D., Chesshire, G. and Henderson, M. E., "On Constructing Three Dimensional Overlapping Grids with CMPGRD," *Software Systems for Surface Modeling and Grid Generation,* Ed. Robert E. Smith, Hampton, VA, Langley Research Center, NASA CP 3143, 1992, pp. 415–434.

[31] Chan, W. M. and Buning, P. G., "Surface Grid Generation Methods for Overset Grids," *Computers and Fluids,* Vol. 24, No. 5, 1995, pp. 509–522.

[32] 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 91-1587, *Proceedings of the 10th AIAA Computational Fluid Dynamics Conference,* Honolulu, Hawaii, 1991.

[33] Chan, W. M. and Steger, J. L., "Enhancements of a Three-Dimensional Hyperbolic Grid Generation Scheme," *Appl. Math. & Comput.* Vol. 51, 1992, pp. 181–205.

[34] Chan, W. M., Chiu, I. T. and Buning, P. G., "User's Manual for the HYPGEN Hyperbolic Grid Generator and the HGUI Graphical User Interface," NASA TM 108791, October, 1993.

[35] Meakin, R., "Adaptive Spatial Partitioning and Refinement for Overset Structured Grids," *Comput. Methods Appl. Mech. Engrg.,* Vol. 189, 2000, pp. 1077–1117.

[36] Chan, W. M., Meakin, R. L. and Potsdam, M. A., "CHSSI Software for Geometrically Complex Unsteady Aerodynamic Applications," AIAA Paper 2001-0593, Reno, Nevada, January, 2001.

[37] Buning, P. G., Jespersen, D. C., Pulliam, T. H., Klopfer, G. H., Chan, W. M., Slotnick, J. P., Krist, S. E. and Renze, K. J., "OVERFLOW User's Manual," Version 1.8q, August, 2000.