# HANDBOOK OF GRID GENERATION

grid components. Since available software has not been designed with overlapping grids in mind, problem components are typically gridded independently in a sequential fashion. Given the level of geometric and physical complexity that is often required for flow simulation, this practice places a heavy burden on the analyst in terms of time and expertise required to generate needed grids. Fortunately, grid generation schemes that exploit the flexibility inherent to an overset approach are active areas of research [Petersson, 1995; Chan and Meakin, 1997]. Efficient and highly automated methods of overset grid generation and domain decomposition should be available in the near future.

The present chapter is divided into three main sections covering the topics of domain decomposition, domain connectivity, and research issues. These sections are followed by brief sections that define terms, references, and sources for more detailed information on subjects related to overset grids. Terms peculiar to overset grid nomenclature appear in italic at their first occurrence, and are defined in Section 11.5.

For the purposes of this chapter, the starting point for grid generation is assumed to be a trimmed "water-tight" definition of problem surface geometry in a suitable format (e.g., NURBS, or panel networks). Note that the subjects of surface and volume grid generation are covered in other chapters of the handbook (Chapter 9 and 4, respectively) and will be referred to only indirectly in the present chapter. Chapter 5 on hyperbolic grid generation should be of particular interest to anyone seeking more information about the overset grid approach.

## 11.2 Domain Decomposition

This section covers domain decomposition issues for composite overset structured grids. Included in the discussion are surface geometry decomposition, volume decomposition, and issues peculiar to multiple-body applications.

### 11.2.1 Surface Geometry Decomposition

All real objects can be viewed as composites of discontinuities (point and line) and simple surfaces. A finite cone, for example, has both point and line discontinuities. Surface geometry entities not associated with point or line discontinuities are simple surfaces. The task of surface geometry decomposition is to partition given problem definitions into sets of surface areas that can readily be converted into overlapping surface grid components. It is worth noting that surface geometry decomposition problems do not have unique solutions. A number of trivial shapes can be represented very well with a single surface (e.g., a sphere, a rectangular flat plate, etc.). However, even simple shapes can be decomposed into component parts and represented with an infinite variety of sets of component surface areas. The present objective is simply to define a convenient set of surface areas to form the basis for surface grid generation. In this chapter, the term *seam* is used to denote surface areas that are associated with either point or line discontinuities in a geometry definition. The term *block* is used to denote simple surface areas. Hence, the task of surface geometry decomposition can be restated as one of partitioning a given problem geometry into a *quilt*[*] of overlapping seams and blocks (see Figure 11.1).

Once a surface definition has been decomposed into seams and blocks, generation of a corresponding number of overlapping surface grids is a conceptually simple task. Most of the basic algorithms needed to develop fully automated surface grid generation software currently exist. Algebraic and elliptic surface grid generation techniques, appropriate for simple surfaces, have long been available (see Chapter 9 of this handbook). The idea for hyperbolic surface grid generation (Chapter 5) was put forward more recently [Steger, 1989], and has since been generalized [Chan and Buning, 1995].

[*] Quilt nomenclature has been adopted here to describe surface geometry decomposition issues unique to composite overset structured grids. The patches of material stitched together in "patchwork quilts" are commonly know as "blocks." Hence, in this analogy, seam and block surface components correspond to quilt stitches and square quilt patches, respectively.

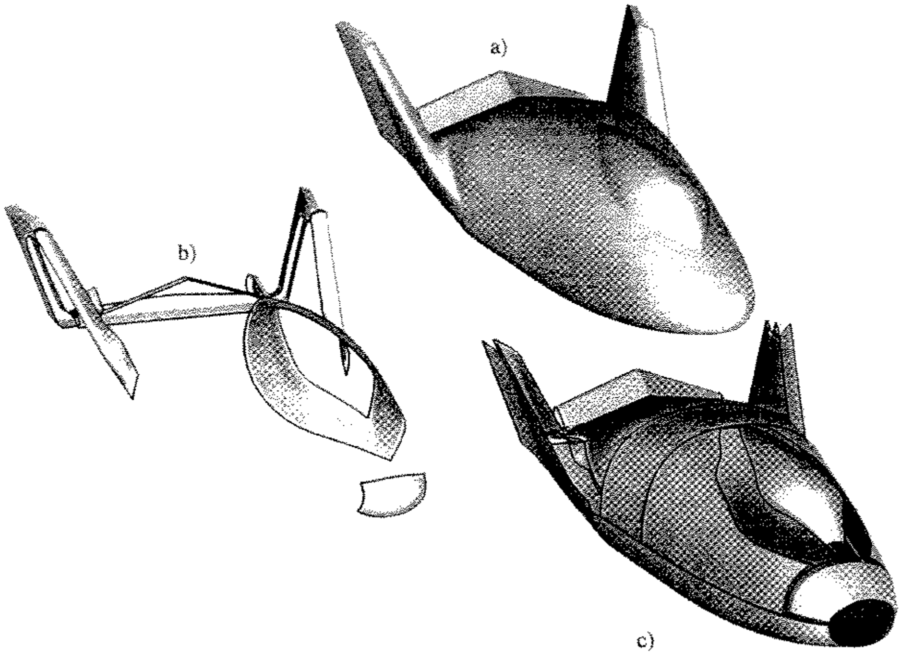


**FIGURE 11.1** Surface geometry decomposition into a quilt of seams and blocks. (a) X-38 surface geometry definition, (b) seams over control lines and line discontinuities, (c) blocks over simple surfaces.

### 11.2.1.1 Seam Topologies

Point discontinuities can exist as a natural feature of an object, such as the tip of a cone. Such situations may dictate the use of a *tip* topology for the surface area in the immediate vicinity of the discontinuity. A tip topology is defined by placing a grid point coincident with the discontinuity and marching away from the point an acceptable distance on the surface. A tip decomposition preserves the point discontinuity in the corresponding surface grid. In addition, a volume grid generated from the surface will have a polar axis that extends from the discontinuity. The existence of an axis generally implies that the flow solver will be required to implement special boundary conditions along the axis. Typically, this means that the flow solution along the axis will be derived from an averaging process involving the nearby off-axis solution. If the point discontinuity is mild, as in a wide-angle cone, it may be acceptable to ignore the discontinuity and use a block topology in the vicinity of the point.

Figure 11.2 indicates two surfaces that could be decomposed with a tip topology. The tip of a generic finned-store is shown in Figure 11.2a, and an aircraft fuselage nose tip is shown in Figure 11.2b. The figure contrasts narrow and wide-angle tips, and illustrates how a wide-angle tip can be appropriately represented as a block (i.e., a simple surface) rather than as a seam.

Surface intersections on an object result in line discontinuities, such as at the junction between an aircraft fuselage and wing. An object can also have line discontinuities as a result of "forced contouring," such as exterior mold lines on an automobile, or crease lines that result from plastic deformation of an object due to stress, or fold lines as on the edges of a box. All line discontinuities that are germane to the flow analysis problem at hand must be faithfully represented in the surface grid system. A seam topology can be defined in the vicinity of a line discontinuity by marching in both line-normal directions away from the line an acceptable distance on the surface, resulting in a quadrilateral patch. In this way, a seam topology can be used as the basis for surface grids that are aligned with the discontinuity and accurately represent the surface shape. Figure 11.3 indicates three example seams aligned with surface line discontinuities. Seam components are indicated in the figure for a fuselage crease-line, rotor-blade trailing edge line, and fin-store intersection line. In some of the Chimera literature*, seam topologies like that shown in Figure 11.3c are referred to as *collars* [Parks et al., 1991].

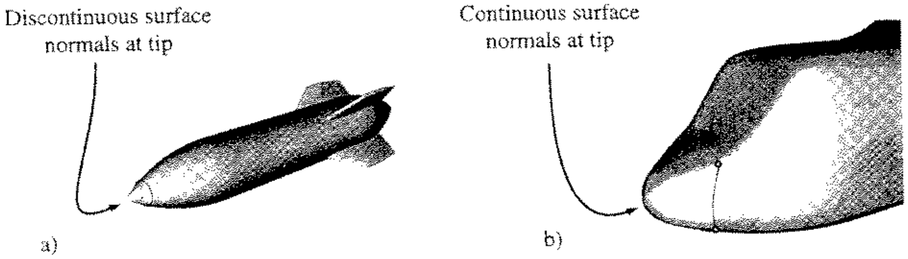

Discontinuous surface normals at tip
Continuous surface normals at tip
a)
b)

**FIGURE 11.2** Seam topologies. (a) Sharp nose of a store decomposed into a surface tip. The radial boundary of the seam is indicated by a thin black line, (b) blunt nose of a fuselage decomposed into a quadrilateral surface area. Seam boundaries are indicated by thin black line segments. Dots indicate seam boundary corners.

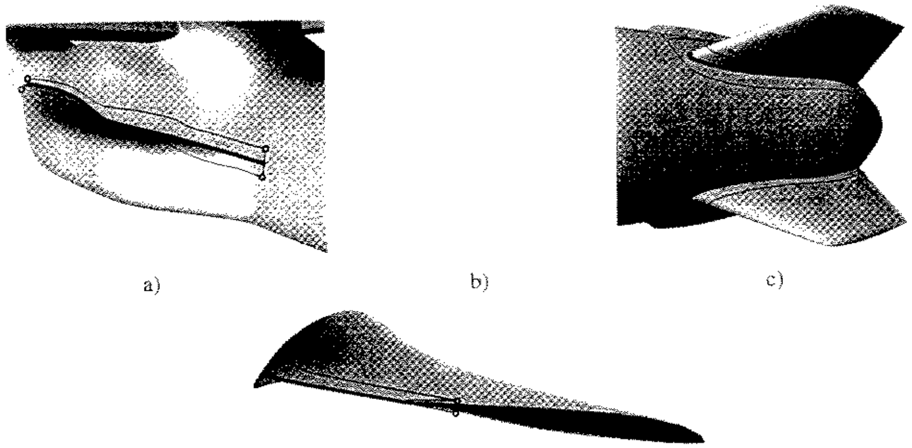


**FIGURE 11.3** Surface geometry decomposition into seams over line discontinuities. Discontinuities are indicated by thick black lines. Seam boundaries are indicated by thin black lines. Dots indicate seam boundary corners. (a) V-22 fuselage/sponson crease, (b) rotor-blade trailing edge, (c) fin-store intersections.

In addition to actual line discontinuities in an object surface, it is sometimes desirable to align grid lines on a surface for other reasons. For example, even though the leading edge of a wing generally has a smooth radius of curvature, and is not a surface discontinuity, accurate flow simulations require a high degree of geometric fidelity of this aspect of a wing surface definition. This is easily done by identifying the wing leading edge as a surface control line, and decomposing the wing surface with a seam topology in the vicinity of the leading edge (see Figure 11.4a). Other examples of seam topologies are shown in Figures 11.1 and 11.4b. Figure 11.1 shows a possible surface geometry decomposition of the X-38 (crew return vehicle). Specifically, Figure 11.1b shows seam components at the vehicle nose, around the canopy, and over the rims of the twin vertical tails. Additional seam topologies are also indicated in the figure (less visible) for various components of the aft portion of the vehicle. Figure 11.4b shows a seam component over the tip of a rotor blade, which avoids the special boundary conditions required by "slit" topologies commonly used as wing and blade tip endings. Seams like this can provide a higher degree of geometric fidelity to the grid system employed than is realizable by collapsing a wing, or blade-tip, into a slit.

[*] A "Chimera" is a mythological creature made up of incongruent parts of other beasts. Steger appropriately coined the term "Chimera overset grids" to indicate a powerful way to apply structured grid solution techniques to geometrically complex multibody configurations.

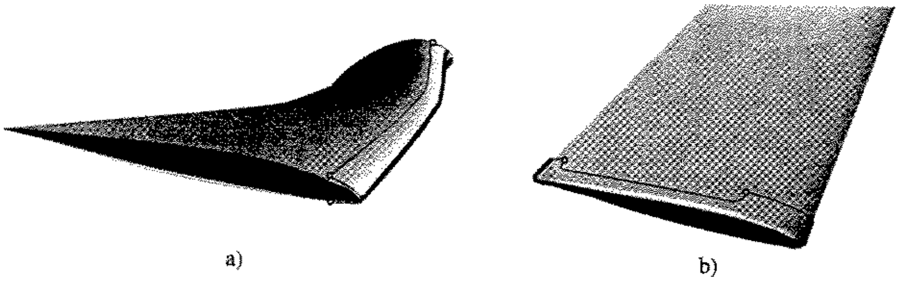


FIGURE 11.4 Surface geometry decomposition into seams over control lines. (a) Rotor-blade leading edge control line, (b) outboard blade-tip ending control line. Control lines are indicated by thick black lines. Seam topology boundaries are indicated by thin black lines. Dots indicate seam boundary corners.

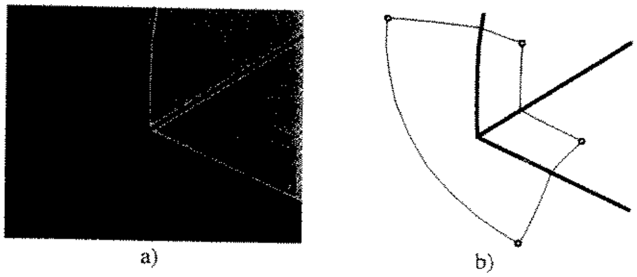


FIGURE 11.5 Surface geometry decomposition in the vicinity of a point discontinuity. (a) Intersection of three line discontinuities, (b) seam topology over the point of intersection. Line discontinuities are indicated by thick black lines. Seam topology boundaries are indicated by thin black lines. Dots indicate seam boundary corners.

A final topology that deserves mention here is one for point discontinuities that result from the intersection of three surfaces, such as exist at the corners of a box. This type of discontinuity defines the point of intersection of three line discontinuities. In most cases of this type, the appropriate decomposition is a seam topology like that which is used for simple line discontinuities. The situation is illustrated in Figure 11.5 for a component of the X-38. For this type of decomposition, two of the three intersecting lines are concatenated into one coordinate line. The seam is then defined by marching in both line-normal directions away from the concatenated line an acceptable distance on the surface, while constraining one of the line-normal seam lines to be co-linear with the third line discontinuity. If the three angles of intersection of a corner point are all narrow, then the corner will approximate a cone and a tip topology can be used instead.

### 11.2.1.2 Block Topologies

Blocks are simple surface areas, or areas that contain mild discontinuities that can effectively be ignored (as in the case shown in Figure 11.2b). Typically the surface area of a given geometry definition can be decomposed primarily into such blocks. For example, Figure 11.1c shows the blocks corresponding to one possible decomposition of the X-38. Block boundaries are always quadrilateral and represent the simplest basis from which structured surface grid components can be generated.

## 11.2.2 Volume Geometry Decomposition

A convenient way to think of volume decomposition is to categorize the physical domain of a problem into near-body and off-body regions. The near-body portion of a domain is defined to include all seams and blocks required to describe problem surface geometry and the volume of space extending a short distance away from the respective surfaces. The off-body portion of a domain is defined to be the domain

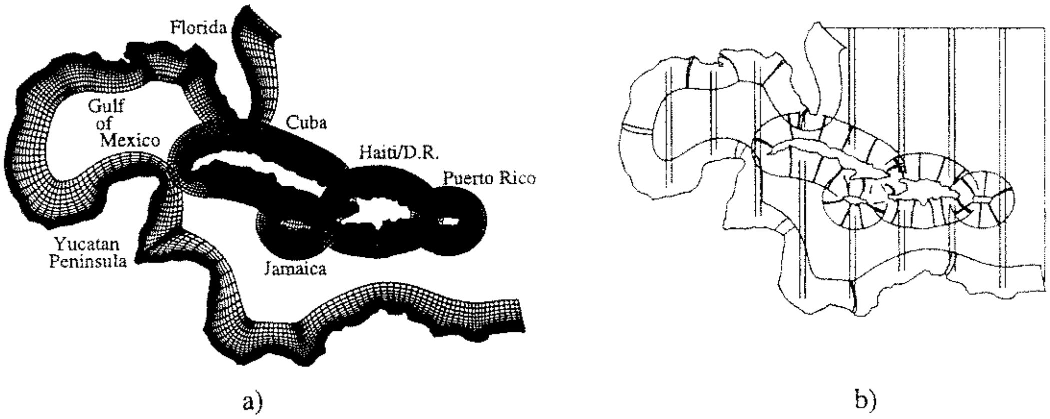


FIGURE 11.6 Overset grid components for unsteady simulation of basin-scale oceanic flows. (a) Body-fitted grid components for the Gulf of Mexico and the Greater Antilles Islands, (b) background Cartesian grid component boundaries.

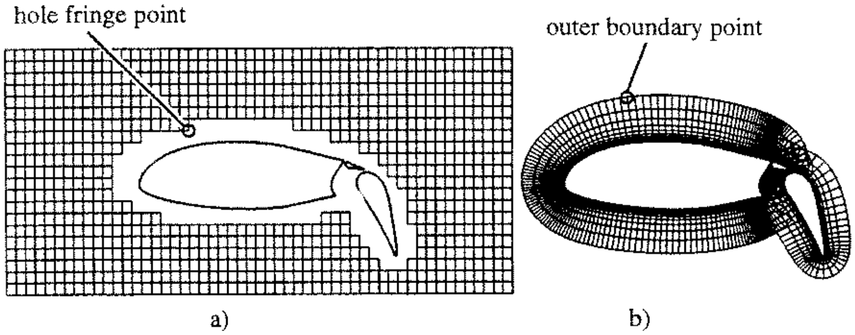


FIGURE 11. 7 Grid components for a flapped-wing configuration. (a) Background Cartesian grid with Chimera hole caused by the wing, (b) body-fitted grid components about the wing and double-slotted flap.

volume not covered (except that required for minimum overlap) by the near-body volumes. The aspect of a *Chimera* overset grid approach that trivializes off-body grid generation is the fact that off-body volume components can overlap the near-body domain by an arbitrary amount. Hence, the off-body domain can be filled using any convenient set of topologies. Typically, Cartesian systems are used for this purpose (e.g., see Figures 11.6b and 11.7a). Hyperbolic grid generation schemes can efficiently generate high quality near-body grids radiating from appropriate quilts of overlapping surface grid components. Generation of off-body Cartesian volume grids (Chapter 22) is trivial for this application.

Although the idea of solving differential equations on overlapping domains is very old [Schwarz, 1869], the idea did not blossom into a practical analysis tool until relatively recent times. Steger et al. [1983] introduced the Chimera method of domain decomposition to treat geometrically complex multiple-body configurations using composite over-set structured grids. In the approach, curvilinear body-fitted structured grids are generated about body components and overset onto systems of topologically simple background grid components. Solutions to the governing flow equations are then obtained by solving the requisite systems of difference equations according to some prescribed sequence on all grid components. Physical boundary conditions are enforced as usual (e.g., no-slip conditions at solid surfaces), while intergrid boundary conditions are supplied from solutions in the overlap regions of neighboring grid components. The solution procedure is repeated iteratively to facilitate free transfer of information between all grids, and to drive the overall solution to convergence. Intergrid boundary conditions are typically updated explicitly.

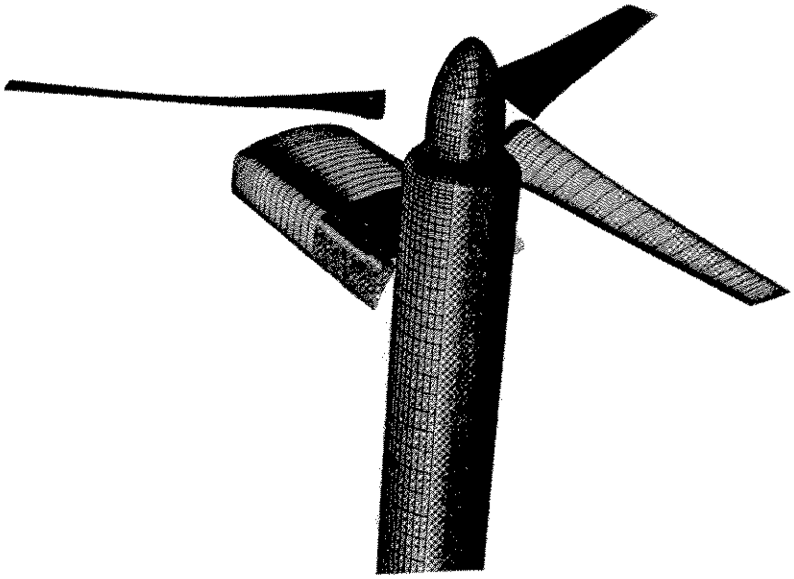


**FIGURE 11.8** Selected surface grid components for a tiltrotor and flapped-wing configuration. Rotor blade grids move relative to stationary wing, nacelle, and background grid components.

Examples of the Chimera method of domain decomposition are illustrated in Figures 11.6 through 11.9 (see Chapter 5 of this handbook for more examples). Figure 11.6 indicates a set of overlapping grids for unsteady simulation of basin-scale oceanic flows in the Gulf of Mexico [Barnette and Ober, 1994]. Body-fitted grids are used to discretize the Gulf coastline and the Greater Antilles Islands. The body-fitted grids are overset onto a system of Cartesian grids that cover the rest of the oceanic region enclosed within the Gulf Coast solution domain. In the figure, the outlines of nine Cartesian grid components are indicated. However, the number of off-body Cartesian grids used is arbitrary. The body-fitted island grids of Figure 11.6 are topologically similar to the body-fitted grids used to discretize the flapped-wing illustrated in Figures 11.7 and 11.8. Figure 11.8 is illustrative of the capacity of an overset grid approach to accommodate relative motion between problem components. The grid components shown in Figure 11.8 are for a tiltrotor and flapped-wing configuration [Meakin, 1995]. Grids associated with the rotor-blades move relative to stationary wing and background grid components. Figure 11.9 shows a detail of some of the overlapping surface grid components of the integrated space shuttle vehicle [Gomez and Ma, 1994]. The figure indicates the degree of geometric complexity and fidelity that has been realized with the approach.

The novel contribution of Chimera to the overall approach of structured grid based domain decomposition is the allowance for *holes* within grid components. For example, the rotor-blade grids shown in Figure 11.8 cut through several other grid components during the course of a simulation. Likewise, the flapped-wing grids cut holes in background Cartesian grid systems. A detail of this is shown in Figure 11.7, where a hole is cut in a background Cartesian grid by the flapped-wing. As indicated in the figure, a Chimera domain decomposition gives rise to two types of intergrid boundary points: hole *fringe points* and grid component *outer boundary points.*

It is a relatively simple matter to adapt any viable structured grid flow solver to function within the framework of Chimera overset grids. For example, the implicit approximately factored algorithm of Warming and Beam [1978] for the thin-layer Navier–Stokes equations

$$\partial_\tau \hat{Q} + \partial_\xi \hat{E} + \partial_\eta \hat{F} + \partial_\zeta \hat{G} = Re^{-1}\partial_\zeta \hat{S} \qquad (11.1)$$

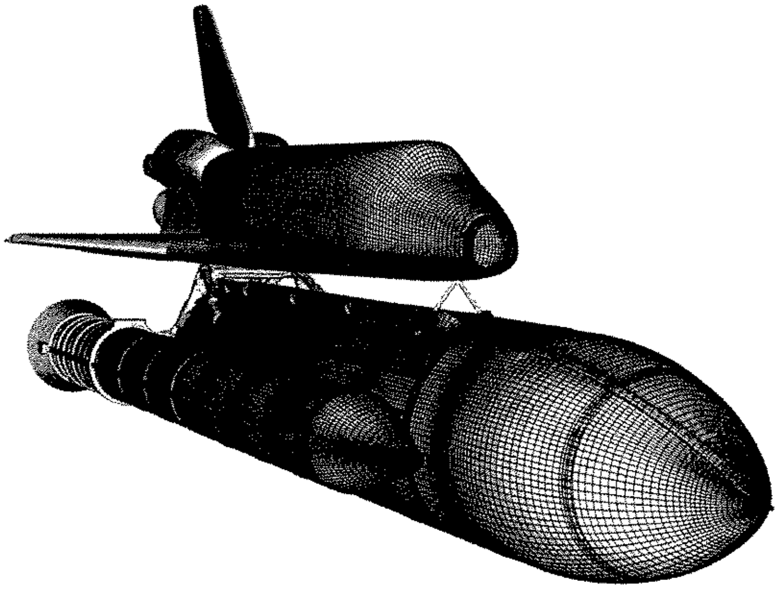


**FIGURE 11.9** Selected surface grid components from a composite overset grid discretization of the integrated space shuttle vehicle.

is easily expressed in Chimera form as

$$\begin{gathered}\left[I+i_b\Delta t\delta_\xi\hat{A}^n\right]\left[I+i_b\Delta t\delta_\eta\hat{B}^n\right]\left[I+i_b\Delta t\delta_\zeta\hat{C}^n-i_b\Delta tRe^{-1}\delta_\zeta J^{-1}\hat{M}^nJ\right]\Delta\hat{Q}^n=\\ -i_b\Delta t\left(\delta_\xi\hat{E}^n+\delta_\eta\hat{F}^n+\delta_\zeta\hat{G}^n-Re^{-1}\delta_\zeta\hat{S}^n\right)\end{gathered} \tag{11.2}$$

The single and overset grid versions of the algorithm are identical except for the variable $i_b$, which accommodates the possibility of having arbitrary holes in the grid. The array $i_b$ has values of either 0 (for hole points), or 1 (for conventional *field points*). Accordingly, points inside a hole are not updated (i.e., $\Delta Q = 0$) and the solution values on intergrid boundary points are supplied via interpolation from corresponding solutions in the overlap region of neighboring grid systems. By using the $i_b$ array, it is not necessary to provide special branching logic to avoid hole points, and all vector and parallel properties of the basic algorithm remain unchanged [Steger et al., 1983].

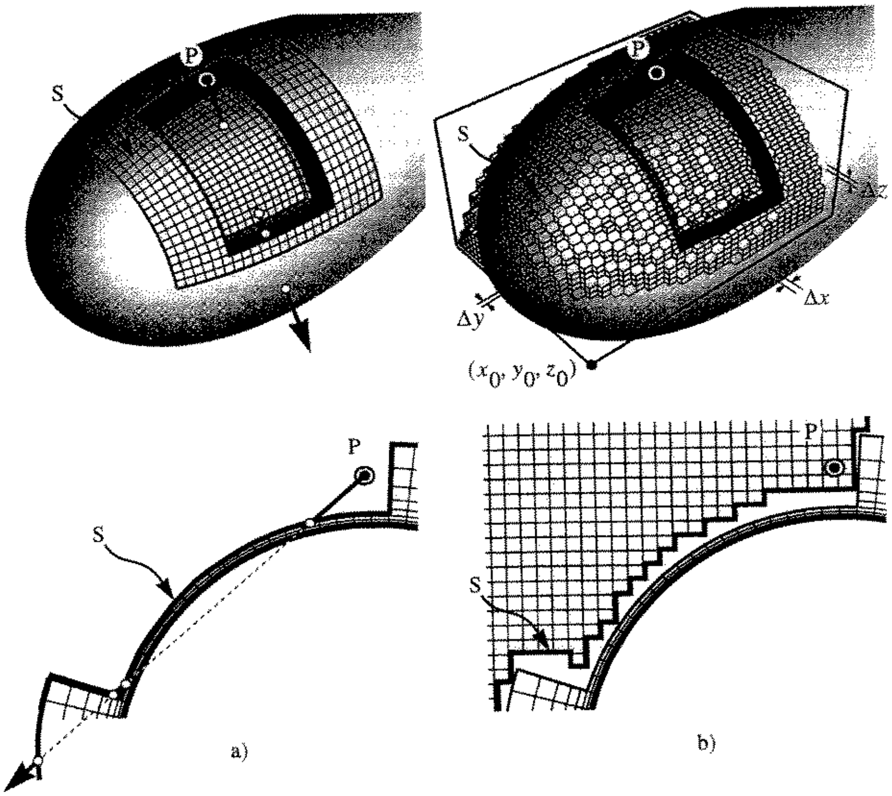
## 11.2.3 Chimera Hole-Cutting

Definition of the Chimera $i_b$ array is an important step in the realization of the several advantages of a general composite overset structured grid approach. The $i_b$ array accommodates the possibility of arbitrary holes in grid components, and thereby, allows efficient execution of the structured grid flow solver being used. The only nontrivial task associated with the definition of $i_b$ is to determine whether points in a given grid component lie inside specified hole-cutting surfaces. A number of procedures are available to make this determination. Consider point $P$ and a surface $S$ defined by a group of surface grid components taken from an existing set of volume grids (see Figure 11.10).

### 11.2.3.1 Surface Normal Vector Test

Let $\boldsymbol{r}_p$ be the position vector of point $P$, $\boldsymbol{r}_{ij}$, the position vectors* of discrete points on $S$, and $\boldsymbol{n}_{ij}$ the surface normal vectors at points $\boldsymbol{r}_{ij}$. Point $P$ is outside of surface $S$ if any of the dot products $(\boldsymbol{r}_p - \boldsymbol{r}_{ij}) \cdot \boldsymbol{n}_{ij} > 0$.

*Note that the use of "*ij*" here is to denote grid indices, rather than tensor rank.



a)

b)

**FIGURE 11.10** Chimera hole cutting. Given cutting surface S, determine the hole/field status of point P. (a) Hole cutting surface defined by collection of surfaces from existing grid components. (b) Approximate hole cutting surface represented with a uniform Cartesian "hole-map."

The cost of this operation is proportional to the number of points in the grid component being tested and the number of points used to define the hole-cutting surfaces. Typically, the cost of the test is reduced by trading dot product computations for computations to determine the Euclidean distance between point couples. Hence, the hole-cutter surface point nearest to point *P* is first identified. Then, only the dot product between *P* and the nearest hole-cutter surface point needs to be computed.

The surface normal vector test has one significant failure mode. Hole-cutting surfaces, viewed from the outside, must be convex. Even if hole-cutting surfaces are constructed from multiple surface grid components, the composite surface must be closed and convex. Hole-cutting surfaces that have concavities must be broken into multiple closed convex surfaces.

### 11.2.3.2 Vector Intersection Test

The number of intersections between an arbitrary ray extended from a point *P* and any closed hole-cutting surface can be used as the basis of a robust and unambiguous inside/outside test. If a ray intersects the closed surface an odd number of times, then the point is inside. If the ray intersects the surface zero or an even number of times, the point is outside. The test is illustrated in Figure 11.10a with an arbitrary ray drawn from a test point in the proximity of *S*. Results of the test are independent of the direction in which rays are extended from the test points, and do not require that the hole-cutting surfaces be convex.

If a ray extended from a test point intersects the hole-cutting surface at an edge of a face that is coplanar with the ray, the test will fail. However, the failure is easily overcome by redefining the ray in a random direction away from the offending face. Implementation of this test is more complicated than for the surface normal vector test. Still, the test is practical and may provide a more robust mechanism for hole determination.

#### 11.2.3.3 Uniform Cartesian Test

An idea suggested by Steger [1992] offers an efficient means of hole point determination that may prove to be the basis of future fully automatic domain connectivity algorithms. A closed surface $S$ can be enclosed within a uniform Cartesian grid, as indicated in Figure 11.10b. Points in the grid can be marked as being inside or outside of $S$ very easily [Chiu and Meakin, 1995]. A uniform Cartesian grid so marked becomes an approximate representation of $S$ and is referred to as a *hole-map*. The proximity of any point $P$ with respect to surface $S$ can be determined by consulting the hole-map of $S$.

Given the coordinates of $P$, the corresponding bounding hole-map element can be computed directly as

$$j = \frac{\left(x_P - x_0\right)}{\Delta x} + 1, \quad k = \frac{\left(y_P - y_0\right)}{\Delta y} + 1, \quad l = \frac{\left(z_P - z_0\right)}{\Delta z} \tag{11.3}$$

where $x_0$, $y_0$, $z_0$ are the coordinates of the hole-map origin, and $\Delta x$, $\Delta y$, $\Delta z$ are the hole-map spacings. If the eight vertices of hole-map element $j$, $k$, $l$ are all marked as a hole, then $P$ is inside the hole-cutting surface. If the eight vertices are all unmarked, $P$ is outside the surface. However, if the eight vertices are of mixed type (marked and unmarked), $P$ is near a hole-cutting plane and a simple radius test, or the vector intersection test can be used to determine the actual status of $P$.

### 11.2.4 Identification of Intergrid Boundary Points

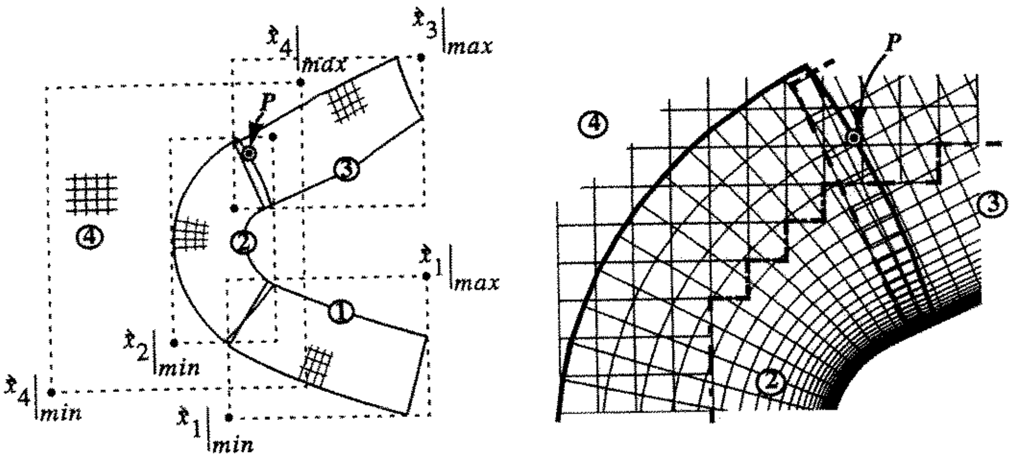
The solution of field equations on overlapping systems of grids requires numerical boundary conditions to be supplied at all intergrid boundary points. Given a definition of the $i_b$ array, it is very easy to identify the intergrid boundary points that exist in all components of an overset system of grids. Points on grid component outer boundaries that are not physical boundaries (e.g., no-slip surfaces), conventional numerical boundaries (e.g., planes of symmetry, inflow/outflow, etc.), or Chimera hole-points, are intergrid boundary points. In addition, field points bordered by one or more hole-points are also intergrid boundary points (i.e., *fringe-points*).

Accordingly, a list of all intergrid boundary points can be made by inspecting the $i_b$ array on a grid-by-grid basis. The list must include the grid component identity, and the $j$, $k$, $l$ and $x$, $y$, $z$ coordinates of each intergrid boundary point in the system of grids. Such a list completely defines the domain connectivity needs associated with a given overset system of grids and specified hole-cutter definitions.

## 11.3 Domain Connectivity

The costs of the advantages inherent to an overset grid approach are reflected in the need to establish domain connectivity. Domain connectivity is the communication of dependent variable information between grid components. Transmission of this information occurs through the intergrid boundary points of a problem. Specifically, values of the dependent variables are defined on intergrid boundary points by interpolation from the interior of overlapping neighboring grid systems. Accordingly, the domain connectivity solution for a given system of overlapping grids is the identity of a suitable *donor* element for each intergrid boundary point in the system. The present section describes basic algorithms for establishing domain connectivity among general systems of overlapping structured grids.

General implementations of the method must allow for grid components posed in curvilinear coordinate systems. This fact makes the task of establishing domain connectivity nontrivial. The position of points within all grid components is defined relative to a fixed reference frame. Data structure is realized on a component-wise basis due to the fact that grid points are distributed along curvilinear coordinate lines. However, the coordinate systems of the respective grid components are mutually independent. Hence, there is no direct correspondence between the computational space of one grid component and that of any other component in the system. The task of establishing domain connectivity can be stated for a single intergrid boundary point as follows. Given an intergrid boundary point $P$, identify a grid



a) Overlapping grid components.

b) Detail of intergrid boundaries.

**FIGURE 11.11** Donor grid identification using simple min/max bounding boxes. (a) Three overlapping grid components overset onto a background Cartesian grid. Bounding-boxes are indicated by light dashed lines. Black dots are used to indicate bounding box diagonal end points. (b) Detail of intergrid boundaries. Point $P$ is bounded by an element from component 3 and from component 4.

component that can satisfy the domain connectivity needs of $P$, and the position of $P$ within the computational space of the donating component. The following sections of this chapter describe alternative methods of establishing domain connectivity for a single intergrid boundary point. Of course, to establish domain connectivity for an entire overset system of grids, any such method would need to be applied to all of the intergrid boundary points in the system.

### 11.3.1 Donor Grid Identification

Typically, only one donor element will exist for a given intergrid boundary point. Indeed, an individual intergrid boundary point can be bounded by only one element from any one overlapping grid component. However, it is not uncommon for some intergrid boundary points to be overlapped by more than one neighboring grid component, leading to the possibility of multiple donor elements for such points. The situation is illustrated in Figure 11.11, where intergrid boundary point $P$ is overlapped by an element from two different grid components. Identification of the grid which provides interpolation information for point $P$ depends on which donor element provides the best match. A discussion of "best" donor is given shortly. However, first consider a method for identifying all grid components that might contain a donor element for point $P$.

The extreme values, $(x_{\min}, y_{\min}, z_{\min})$ and $(x_{\max}, y_{\max}, z_{\max})$, of the reference frame coordinates of any grid component define the diagonal end-points of a rectilinear box that encompasses the entire component (e.g., see Figure 11.11a). Even for an overset system of grids that contains numerous grid components, the information required to define all diagonal end-points is minimal (*viz.*, $6 \times N$, where $N$ is the number of grids). A necessary condition for donor grid identification is that $P$ be bounded by the diagonal end-points of the grid component in question, i.e.,

$$x_{\min} < x_P < x_{\max}\,, \quad y_{\min} < y_P < y_{\max}\,, \quad z_{\min} < z_P < z_{\max} \tag{11.4}$$

If the grid component is Cartesian, then Eq. 11.4 is sufficient proof that the component contains a donor element for $P$. However, in general, overset grid discretizations are comprised of (at least some) non-Cartesian grid components. Therefore, in general, Eq. 11.4 is only an indicator of donor potential. For example, Figure 11.11a illustrates three overlapping grid components overset onto a background Cartesian grid component. The bounding-box diagonals readily indicate donor potential between the four components shown. The present discussion considers identification of the donor grid and element for a single intergrid boundary point. In practice, this information is sought for groups of intergrid boundary points at a time. In this sense, the information available through Eq. 11.4 also provides a simple mechanism for identifying all intergrid boundary points that may have a donor in a given grid component.

For intergrid boundary points that are bounded by an element from more than one neighboring grid component, a choice must be made as to which element will be allowed to provide the needed donor information. Current domain connectivity algorithms employ only a rudimentary set of rules for determining the acceptability of available donor elements. The most fundamental rule is that none of the vertices which define a donor element can be hole points. Values of the dependent variables are not defined at hole points. Hence, acceptance of donor elements with any number of hole-point vertices would corrupt the transfer of dependent variable information to the receiving intergrid boundary point. Typically, the first donor element identified that satisfies the rudimentary set of donor acceptability rules is used when more than one bounding element exists for a given intergrid boundary point.

The accuracy of dependent variable information transfer is maximum when the geometric properties of donor and recipient elements are comparable. The relative volume size, orientation, and aspect ratios between donor and recipient elements govern sources of numerical error in the intergrid communication process. Of course, the magnitude of numerical error is proportional to the gradients of the dependent variables being communicated between the grids. Hence, if the dependent variables are represented smoothly in donor and recipient grids, then the error will be small. Indeed, formal solution accuracy can be maintained on overlapped systems of grids using simple interpolation [Meakin, 1994]. Hence, given a robust method of adaption to guarantee smooth variation of dependent flow variables in computational space, the existing rudimental rules of donor element acceptability should be sufficient.

There are two reasons why this is not the case in practice, and that donor acceptability rule definition constitutes a valid area of research. First, very few flow solvers that accommodate an overset grid approach also have an adaptive capability. Therefore, resolution of gradients of the dependent variables cannot be ensured in most overset grid solvers currently available. In addition, the magnitude of interpolation error for resulting applications can only be estimated after the fact. Second, donor acceptability rules based on geometric measures of goodness will accept only the best available donor element in instances where multiple donors exist. Probably much more significantly, rules based on geometric measures of goodness can form the basis of an iterative procedure to obtain the best realizable domain connectivity solution from a given system of grids. Maximization of domain connectivity solution quality will even contribute to error reduction when coupled with solution adaption.

## 11.3.2 Donor Element Identification

Once it has been determined that a given grid component may contain a suitable donor for an intergrid boundary point *P*, the task remains to identify the actual element that bounds *P* and evaluate its acceptability. Some of the acceptability issues have been discussed in the previous section. In the present section, methods of donor element identification are given. By far the simplest method of donor element identification is an exhaustive search. Such a scheme would involve testing all possible elements within a grid component to determine if point *P* is inside, or not. Although simple inside/outside tests can be devised, the cost of an exhaustive search is prohibitive for all but highly idealized problems. Fortunately, a variety of alternatives to an exhaustive search exist. Since all search procedures require an inside/outside test, a particularly useful test is described below. Then, some of the search algorithms in common use within available domain connectivity codes are discussed.

### 11.3.2.1 Inside/Outside Test

Let $\mathbf{x}(\mathbf{s}_p)$ define the reference frame coordinates of intergrid boundary point *P* as a function of the computational space coordinates **s** of a candidate interpolation donor element. Values of **x** and **s** are known for the eight vertices of the candidate donor element. The value of **x** is also known for point *P*.

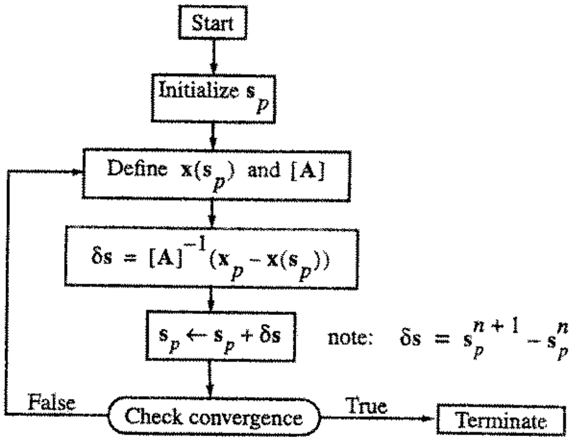

Start
Initialize $\mathbf{s}_p$
Define $\mathbf{x}(\mathbf{s}_p)$ and $[\mathbf{A}]$
$\delta\mathbf{s} = [\mathbf{A}]^{-1}(\mathbf{x}_p - \mathbf{x}(\mathbf{s}_p))$
$\mathbf{s}_p \leftarrow \mathbf{s}_p + \delta\mathbf{s}$
note: $\delta\mathbf{s} = \mathbf{s}_p^{n+1} - \mathbf{s}_p^{n}$
False
Check convergence
True
Terminate

**FIGURE 11.12** Iteration to solve Eq. 11.5 for the computational space position of a point relative to the origin of a candidate interpolation donor element.

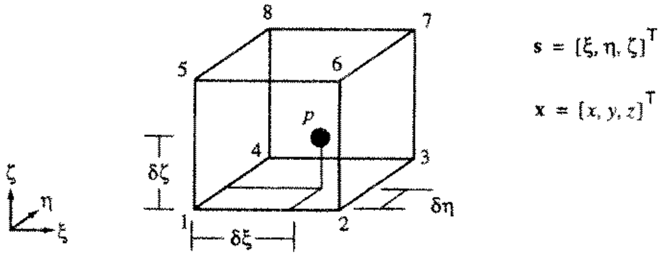


**FIGURE 11.13** The computational space of a candidate interpolation donor element for point *P*.

We seek values of **s** for point *P*, $\mathbf{s}_P$. If *P* is inside the element, values of $\mathbf{s}_P$ will be bounded between 0 and 1. A quadratically convergent iterative scheme for $\mathbf{s}_P$ can be constructed from Eq. 11.5 and is outlined in Figure 11.12.

$$\mathbf{x}_p = \mathbf{x}(\mathbf{s}_p) + [\mathbf{A}]\delta\mathbf{s} \tag{11.5}$$

If the solution to Eq. 11.5 produces values of $\mathbf{s}_P < 0$, or $\mathbf{s}_P > 1$, *P* is outside the candidate donor element. The expressions used to define $\mathbf{x}(\mathbf{s}_P)$ and $[\mathbf{A}]$ depend on the specific interpolation scheme used to define the variation **s** of within the donor element. A definition of $\mathbf{x}(\mathbf{s}_P)$ is given by Eq. 11.6 below, assuming the use of trilinear interpolation and the element notation indicated in Figure 11.13.

$$\begin{aligned}\mathbf{x}(\mathbf{s}_p) = \mathbf{x}_1 &+ (-\mathbf{x}_1 + \mathbf{x}_2)\xi + (-\mathbf{x}_1 + \mathbf{x}_4)\eta + (-\mathbf{x}_1 + \mathbf{x}_5)\zeta \\ &+ (\mathbf{x}_1 - \mathbf{x}_2 + \mathbf{x}_3 - \mathbf{x}_4)\xi\eta \\ &+ (\mathbf{x}_1 - \mathbf{x}_2 - \mathbf{x}_5 + \mathbf{x}_6)\xi\zeta \\ &+ (\mathbf{x}_1 - \mathbf{x}_4 - \mathbf{x}_5 + \mathbf{x}_8)\eta\zeta \\ &+ (-\mathbf{x}_1 + \mathbf{x}_2 - \mathbf{x}_3 + \mathbf{x}_4 + \mathbf{x}_5 - \mathbf{x}_6 + \mathbf{x}_7 - \mathbf{x}_8)\xi\eta\zeta\end{aligned} \tag{11.6}$$

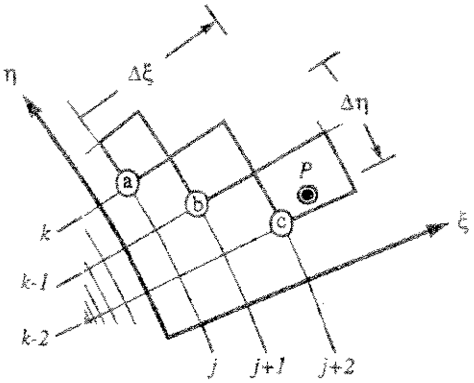


**FIGURE 11.14** Gradient search of a given grid component for the element that bounds point P. Search initiated in element "a" and terminated in bounding element "c."

The gradient matrix [A] is simply

$$[\mathbf{A}] = \begin{bmatrix} \dfrac{\partial}{\partial \xi} x(s_p) & \dfrac{\partial}{\partial \eta} x(s_p) & \dfrac{\partial}{\partial \zeta} x(s_p) \\ \dfrac{\partial}{\partial \xi} y(s_p) & \dfrac{\partial}{\partial \eta} y(s_p) & \dfrac{\partial}{\partial \zeta} y(s_p) \\ \dfrac{\partial}{\partial \xi} z(s_p) & \dfrac{\partial}{\partial \eta} z(s_p) & \dfrac{\partial}{\partial \zeta} z(s_p) \end{bmatrix} \tag{11.7}$$

where the elements of [A] are the corresponding derivatives of Eq. 11.6.

Although other methods exist to determine whether, or not, a point is inside a given element, the iteration defined by Eq. 11.5 is certainly adequate. Equation 11.5 is an expression of the method of steepest descents, and can be used to drive a gradient search procedure* for the bounding element of *P*.

### 11.3.2.2 Gradient Search

The use of a gradient search procedure to find an element within a structured grid component that bounds a given intergrid boundary point can be very effective [Benek et al., 1986]. A search for the element that bounds point *P* can be initiated from an arbitrary element in the donating grid component. However, typical implementations of the method often begin by evaluating the euclidean distance between *P* and points on the grid component outer boundary. The grid component outer boundary point nearest to *P* defines a convenient element from which to initiate the search. If this element fails, at least the actual donor is nearby and, hopefully, only a few steps of the search procedure will be required to find it.

In any case, the element identified as the search starting point is considered as a candidate donor element for point *P*, and Eq. 11.5 is solved for the local coordinate increments $\mathbf{s}_p$ from the candidate donor element origin. If the vector components of $\mathbf{s}_p$ are bounded between 0 and 1, then *P* is inside the element and the search is complete. If any of the components of $\mathbf{s}_p$ are outside these bounds, the search must be continued. However, the direction (i.e., gradient), in computational space, to the element that bounds *P* is indicated by $\mathbf{s}_p$. For example, consider the situation indicated in Figure 11.14. If Eq. 11.5 is solved for the element "a" and point *P* indicated, the vector components (2D example) of $\mathbf{s}_p$ would be $[(\xi > 1), (\eta < 0)]^T$. Accordingly, the gradient to the actual bounding element points in the $+j$, $-k$ direction in computational space. Further, the correct donor to the example indicated in Figure 11.14 would be identified on the second application of Eq. 11.5 from element "a."

*A gradient search method is commonly referred to as "stencil-walking" in the Chimera overset grid literature.

General implementations of the method must take into account certain grid topological situations that can obscure the path (in computational space) to needed donor elements. A periodic plane, for example, poses a minor difficulty. The same is true for "slit" topologies (e.g., wake cuts). The search procedure will step in computational space toward the element that bounds *P* from the side of the periodic (or slit) plane where the search is initiated. The search will terminate on either the actual bounding element, or on the element nearest *P* on the "wrong" side of the periodic (or slit) plane. A grid about a thin body may also pose a similar type of problem. If a search is initiated for the element that bounds *P* near the actual bounding element in physical space, but far from it in computational space (i.e., because it's on the opposite side of the body), the search may fail. In this case, the search procedure would step in computational space toward the actual bounding element, but terminate on the thin body surface nearest the actual bounding element, but on the "wrong" side of the body. Pathological search situations that arise because of topological issues can easily be remedied by allowing for restart locations within the grid component.

General search algorithms must also allow for degenerate elements. For example, the donor element for a given intergrid boundary point may reside in the volume grid generated from a surface seam component that has a "tip" topology. Accordingly, the volume grid will have a polar axis that emanates from the point discontinuity at the body surface, and all grid elements associated with the axis will have a collapsed edge. Axis elements are prismatic, rather than hexahedral. This type of element can be acceptable. Therefore, a robust domain connectivity algorithm must detect candidate donor element degeneracies and allow the gradient search procedure to continue when encountered.

### 11.3.2.3 Spatial Partitioning

A viable alternative to an exhaustive search approach to donor identification is spatial partitioning. There are numerous methods of this type. Applied to the domain connectivity problem, the methods involve partitioning the physical space of a grid component into rectilinear volumes of space, and establishing a correspondence between partitions and the grid points they contain. Then, the task of domain connectivity is to identify the partition that bounds a given intergrid boundary point, and apply an exhaustive search within the partition to find the actual bounding element. The methods differ primarily in the allowable levels and mechanisms of partitioning.

The simplest spatial partitioning approach is known as the "bucket" method, and allows only one level of partitioning. Applied to the domain connectivity problem, the approach partitions the domain of a grid component into a three-dimensional array of rectilinear buckets. Then, the grid points are sorted into the resulting buckets. In order to find the grid component element that bounds intergrid boundary point *P*, the bucket that contains *P* is first identified. If the data structure used to define the buckets is Cartesian, identification of the bucket that bounds *P* is trivial, otherwise this step could become computationally significant for large problems. Given the identity of the bucket that bounds *P*, an exhaustive search of the grid points contained in the bucket is conducted to find the actual bounding element.

It is possible that none of the points inside the bucket that bounds *P* define the origin of the grid element that bounds *P*. In fact, since the possibility of empty buckets exists, it is possible that the bucket that bounds *P* is empty. Accordingly, if the search of the bounding bucket fails, neighboring buckets must be searched until the actual bounding element is identified. The bucket method is an improvement over an exhaustive search, but is limited by costs associated with nonuniform distribution of grid points among existing buckets. Large numbers of empty buckets may exist, requiring cost to identify the bucket which contains the donor element. Other buckets may have a large number of points, requiring substantial computational effort to do an exhaustive search of individual buckets.

Multilevel partitioning methods exist that remedy many of the shortcomings of the simple bucket method. A "split tree" (binary alternating direction tree) is one example. In this approach, the physical space of a grid component is split into two partitions at each level. The partitioning occurs alternately along planes of constant *x*, *y*, and *z*. Ideally, positioning of the planes is done such that the grid points are divided equally between the two partitions that exist at any level. In this approach, the empty bucket problem does not exist, and grid points are more evenly distributed among partitions. However, the highest level partition (tree branch) that bounds a given intergrid boundary point must be identified. This requires a recursive search of the tree-structure of the partitioning. Once the high-level bounding partition has been identified, an exhaustive search of the points inside the partition is required to identify the actual element that bounds *P*.

As with the bucket method, it is possible that the origin of the bounding element of *P* does not lie inside the partition that bounds *P*. For such cases, neighboring partitions on the same branch that bounds *P* must be searched until the bounding element is found.

### 11.3.2.4 Combined Spatial Partitioning and Gradient Search

Many overset grid applications involve very large geometrically complex three-dimensional domains. Discretization of such problems can involve numerous grid components and millions of grid points. Problems of this magnitude demand an efficient and robust domain connectivity algorithm. A way to enhance the computational efficiency of the search aspects of domain connectivity is to combine the strengths of spatial partitioning with a gradient search procedure. Such an approach has been introduced to provide domain connectivity for problems involving relative motion between component parts [Meakin, 1991].

In the implementation just noted, a set of rectilinear partitions, or "buckets," are used to completely enclose the physical space of a curvilinear grid component, the partitions forming a stair-step enclosure around the grid component boundaries. An example of such a partitioning is illustrated in Figures 11.15a and 11.15b. Each partition encloses a unique portion of the grid component. The computational space of the grid component bounded by a given partition is mapped to a uniform Cartesian grid defined within the partition. Discretization of the several partitions into separate uniform Cartesian grids is a second level of partitioning of the domain, but one that retains data structure to facilitate search by truncation, rather than by exhaustion.

Partitioning and mapping the computational space of a grid component is a one-time expense, even for moving body problems (assuming rigid-body motion). The object of the mapping is to define values of the grid component curvilinear coordinates (*j, k, l*) at each uniform Cartesian grid point within the partitions. This can be done with a single sweep through a grid component. For example, consider the curvilinear grid indicated in Figure 11.15c. All uniform Cartesian points in a partition will be bounded at least once by the bounding box of a curvilinear element from the grid component. Figure 11.15c indicates three curvilinear element bounding boxes associated with points *j,k*, *j,k* + 1, and *j,k* + 2, respectively. The values of *j, k, l* assigned to a bounding-box are the coordinates of the curvilinear element origin bounded by the box. Then, all partition uniform Cartesian points that are also enclosed within the bounding box are defined to have the same *j, k, l* values as the box. Accordingly, in Figure 11.15c, curvilinear grid indices *j, k* (2D) are mapped to the uniform Cartesian points marked with solid squares during the *k*-sweep indicated. The curvilinear grid indices mapped to the two upper solid squares are then overwritten by *j,k* + 1. Also, *j,k* + 1 is mapped to the uniform Cartesian points marked with solid diamonds. This procedure is continued through the *k*-sweep indicated, and on through the curvilinear grid. A single pass through the curvilinear grid is sufficient to define *j, k, l* (3D) at all uniform Cartesian points in all partitions associated with the grid component. This is true even though spacing of the uniform Cartesian grids may be small or large relative to the curvilinear grid. In Figure 11.15c, the uniform Cartesian elements are slightly smaller than the curvilinear grid. In contrast, the uniform Cartesian elements are large relative to the curvilinear grid shown in Figure 11.15d. Figure 11.15e indicates the mapping of the "*k*" coordinate direction to the partitions of the curvilinear grid indicated in Figure 11.15a (shades of gray are proportional to values of *k*).

Given this type of spatial partitioning, a very good approximation to the grid component element that bounds a given intergrid boundary point *P* can be identified directly. First, the partition that contains *P* is identified by evaluating Eq. 11.4 for each partition. Then, the uniform Cartesian element (of the bounding partition) that bounds *P* is computed directly using Eq. 11.3. In many cases, these two steps correctly identify the element that bounds *P*. If the element identified does not bound *P*, Eq. 11.5 can be used to drive a gradient search for the correct bounding element. Since the actual bounding element is

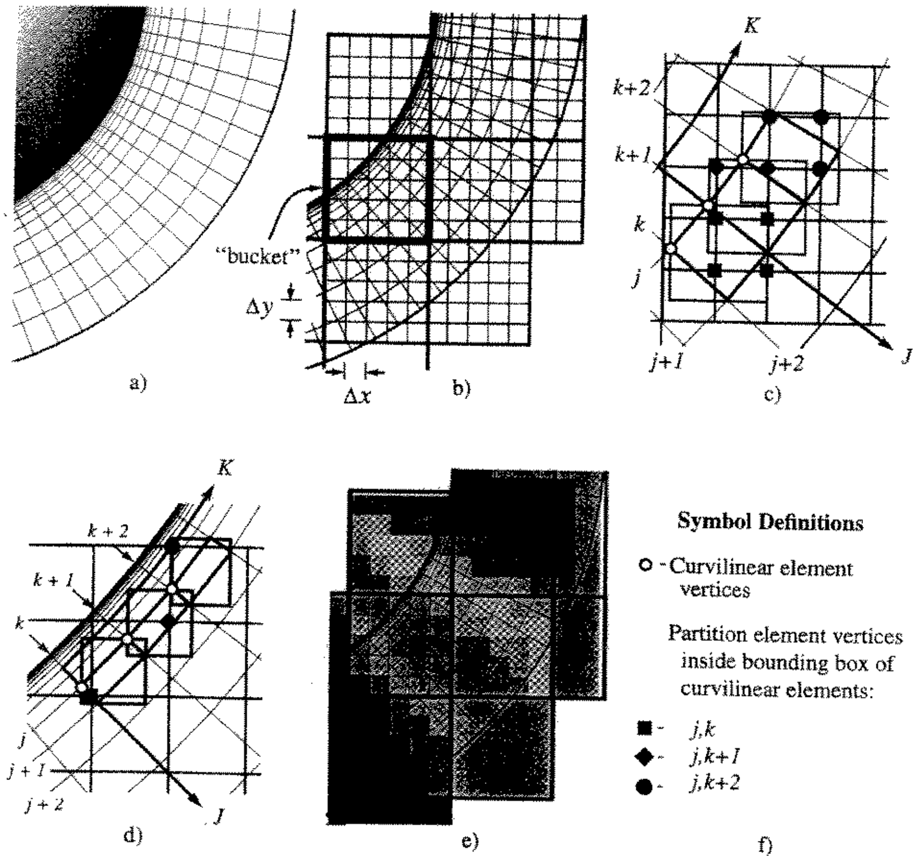


**FIGURE 11.15** Simple spatial partitioning of curvilinear grid component. (a) Curvilinear grid, (b) partition boundaries, or "buckets," (c) *j*, *k* space of curvilinear grid mapped to uniform Cartesian points within a partition, (d) partition uniform Cartesian grid that is coarse relative to the curvilinear grid component being mapped, (e) *k*-coordinate of curvilinear grid component mapped onto uniform Cartesian grids of several partitions, (f) definition of symbols used in (c) and (d).

guaranteed to be near the element identified by the spatial partitioning, only a few steps (at most) of the gradient search routine should be required.

## 11.4 Research Issues

The preceding sections have described basic concepts in domain decomposition and domain connectivity required for implementation of a composite overset structured grid approach. It has been noted that the approach has been used advantageously on a wide variety of applications of practical importance. The approach is indeed very powerful. However, it is still a maturing technology. The following paragraphs provide a brief statement of areas that require further development to realize the full advantages inherent to the approach.

### 11.4.1 Surface Geometry Decomposition

Software tools are needed to simplify the task of surface geometry decomposition. Automation of many aspects of this task are possible (see Chapter 29). This aspect of grid generation is the most fundamental part of grid generation and affects all subsequent processes of analysis from grid generation to solution of PDEs on the final system of grids. Moreover, this aspect of grid generation currently relies most heavily on user expertise, but has the least amount of software available to assist in the task. Some research is being carried out in this area by Petersson [1995], and Chan and Meakin [1997]. Continued effort in the area is needed.

### 11.4.2 Surface and Volume Grid Generation

Research in the areas of surface and volume grid generation for overset grid applications is much less pressing than for surface geometry decomposition. This is so because a number of excellent software tools already exist for performing these tasks automatically (see Chapter 5 of this handbook). The main problem here is that grid generation software designed for overset grids exist as stand-alone entities. A user must be familiar with many codes and input requirements to use the software. A software engineering effort to combine existing overset grid generation tools into a stand-alone and easy-to-use package is needed.

### 11.4.3 Adaptive Refinement

A criticism sometimes leveled against structured grid approaches is that adaptive refinement cannot be done, or is very difficult to do within a structured grid framework. This is simply not true. Adaptive refinement schemes have been developed and applied within structured grid codes for many years. The first adaptive mesh refinement (AMR) schemes began to appear in the international literature in the early 1980s [Berger and Oliger, 1984]. Many advances have been realized since then. Perhaps a more accurate criticism would be to note that structured-grid adaptive refinement applications involving geometrically complex configurations have been very limited. Adaptive refinement needs to be demonstrated for applications of practical importance using overset structured grids. In general, all methods of adaptive refinement require further research to improve generality and robustness. The area of error estimation and feature detection is independent of discretization methodology, and requires further investigation.

### 11.4.4 Domain Connectivity

Software exists to establish domain connectivity among systems of overset structured grids [Benek et al., 1986; Brown et al., 1989; Meakin, 1991; Suhs and Tramel, 1991; Maple and Belk, 1994]. Existing domain connectivity software is very close to providing the degree of automation required for this task. Software advances in the areas of surface geometry decomposition and volume grid generation will eliminate many of the overset grid related problems that currently do not become apparent until domain connectivity is attempted. Still, existing domain connectivity software is deficient in some respects. Requirements for user specified hole-cutting surfaces need to be eliminated. For problems of practical importance, hole-cutter shape specification is a tedious task that is prone to human error. Given a set of volume grids and corresponding topological and boundary condition information, fully automatic, high-quality domain connectivity solutions should be realizable. Advances in methods to create Chimera holes and the establishment of robust definitions of geometric measures of donor element goodness are basic to the realization of fully automatic domain connectivity software.

Even with fully automatic domain connectivity, improved computational efficiency in the areas of Chimera hole-cutting and donor element identification will probably always be desirable. This is especially true for unsteady moving body applications that require domain connectivity to be established at every time-step.

## Defining Terms

**Block**: simple surface area in a geometry definition that can be covered with a quadrilateral patch (see Figure 11.1c).

**Chimera**: a type of domain decomposition that allows arbitrary *holes* in overlapping grid components (see Figure 11.7).

**Collar**: grid component generated from a *seam* about the junction of two surfaces, such as the junction between an aircraft wing and fuselage (see Figure 11.3c).

**Donor element**: the element of a grid component used to supply values of the dependent variables (typically by interpolation) to an intergrid boundary point (see Figure 11.14).

**Field points**: points in a grid component where values of the dependent variables are determined by numerical solution to the governing set of equations to be solved on the grid system.

**Fringe points**: points in a grid component that define the border between conventional *field points* and Chimera *hole points* (see Figure 11.7).

**Hole-map**: an approximate representation of a Chimera hole-cutting surface (see Figure 11.15e).

**Hole points**: points in a grid component for which values of the dependent variables will not be updated or defined (see Figure 11.7).

**Outer boundary points**: points on the exterior surfaces of a grid component that are not flow boundaries or *hole points* (see Figure 11.7).

**Quilt**: surface geometry decomposition that results in a set of overlapping *seams* and *blocks* (see Figure 11.1).

**Seam**: surface areas that are associated with point or line discontinuities, or control lines, in a geometry definition (see Figure 11.1b).

**Tip**: surface topology for an area associated with a point discontinuity in the geometry definition (see Figure 11.2a).

## Acknowledgment

A chapter on composite overset structured grids, such as presented here, must include an acknowledgment of the seminal role of the late Professor Joseph L. Steger to this area of computational mechanics. Recently, the Third Symposium on Overset Composite Grid and Solution Technology was held at the Los Alamos National Laboratory. The impact of Steger's "Chimera" method of domain decomposition was clearly apparent. Applications ranging from biological issues regarding the mechanisms of food particle entrapment inside the oral cavities of vertebrate suspension feeding fish, to the aerodynamic performance of atmospheric reentry vehicles were also presented. Simulations of blast wave propagation to consider safety regulations for launch facilities located near populated regions, studies of the acoustic noise levels of high-speed trains passing through tunnels, and simulations of the aeroacoustic performance of rotary wing aircraft were also presented. Demonstrations of analysis capability that relate to many other aspects of our society were also given. Truly, Professor Steger's influence has been great.

## Further Information

Many domain connectivity issues are actually problems in computational geometry, which has a large literature of its own. The text by O'Rourke [1994] is very good. Melton's Ph.D. thesis [1996] also describes a number of algorithms that are particularly relevant to domain connectivity. A complete discussion of spatial partitioning methods is given in the book by Samet [1990]. *Computational Fluid Dynamics Review 1995* includes a review article on "The Chimera Method of Simulation for Unsteady Three-Dimensional Viscous Flow" [Meakin, 1995a] and has a substantial set of references that point to basic research being carried out in a number of areas related to composite overset structured grids. Henshaw [1996] recently published a review paper on automatic grid generation that devotes a section to overlapping grid generation.

## References

1. Barnette, D. and Ober, C., Progress report on high-performance high resolution simulations of coastal and basin-scale ocean circulation *Proceedings of the 2nd Overset Composite Grid Sol. Tech. Symp.*, Fort Walton Beach, FL, 1994.
2. Benek, J., Steger, J., Dougherty, F., and Buning, P., Chimera: A grid-embedding technique *AEDC-TR-85-64*, 1986.

3. Berger, M. and Oliger, J., Adaptive mesh refinement for hyperbolic partial differential equations *J. Comput. Phys.* 1984, 53: 484–512.
4. Brown, D., Chesshire, G., Henshaw, W., and Kreiss, O., On composite overlapping grids *7th Int. Conf. Finite Element Methods in Flow Probs.*, Huntsville, AL, 1989.
5. Chan, W. and Buning, P., Surface grid generation methods for overset grids *Computers and Fluids.* 24, (5): 509–522.
6. Chan, W. and Meakin, R., Advances towards automatic surface domain decomposition and grid generation for overset grids(submitted for publication) In *Proceedings of the 13th AIAA CFD Conf.*, Snowmass, CO, 1997.
7. Chiu, I.T. and Meakin, R., On automating domain connectivity for overset grids AIAA Paper 95-054, *33rd Aero. Sci. Mtg.* Reno, NV, 1995.
8. Gomez, R. and Ma, E., Validation of a large-scale chimera grid system for the Space Shuttle Launch Vehicle In *Proceedings of the 12th AIAA Applied Aero. Conf.* 1994, Paper 94-1859-CP., pp 445–455.
9. Henshaw, W., Automatic grid generation *Acta Numerica.* 1996, pp 121–148.
10. Maple, R. and Belk, D., Automated set up of blocked, patched, and embedded grids in the beggar flow solver *Numerical Grid Generation in Computational Fluid Dynamics and Related Fields*, Weatherill, N.P., et al., (Ed.), Pine Ridge Press, 1994, pp 305–314.
11. Meakin, R., A new method for establishing intergrid communication among systems of overset grids *Proceedings of the 10th AIAA CFD Conf.*, Paper 91-1586-CP, 1991, pp 662–671.
12. Meakin, R., On the spatial and temporal accuracy of overset grid methods for moving body problems *Proceedings of the 12th AIAA Appl. Aero. Conf.*, 1994, Paper 94-1925-CP, pp 857–871.
13. Meakin, R., The chimera method of simulation for unsteady three-dimensional viscous flow *Computational Fluid Dynamics Review.* Hafez, M. and Oshima, K., (Eds.), John Wiley & Sons, Chichester, England, 1995, pp 70–86.
14. Meakin, R., Unsteady simulation of the viscous flow about a V-22 rotor and wing in hover *Proceedings of the AIAA Atmos. Flght. Mech. Conf.*, 1995, Paper 95-3463-CP, pp 332–344.
15. Melton, J., Automated three-dimensional Cartesian grid generation and Euler flow solutions for arbitrary geometries Ph.D. thesis, University of California, Davis, 1996.
16. O'Rourke, J., *Computational Geometry in C.* Cambridge University Press, 1994.
17. Parks, S., Buning, P., Steger, J., and Chan, W., Collar grids for intersecting geometric components within the chimera Overlapped Grid Scheme *Proceedings of the 10th AIAA CFD Conf.* Paper 91-1587-CP, 1991, pp 672–682.
18. Petersson, N.A., A new algorithm for generating overlapping grids *CAM-report 95-31*, (submitted to *SIAM J. Sci. Comp.*)UCLA, 1995.
19. Samet, H., *The Design and Analysis of Spatial Data Structures.* Addison-Wesley, Reading, MA, 1990.
20. Schwarz, H.A., Ueber einige Abbildungsaufgaben *J. Reine Angew. Math.* 1869, 70, pp 105–120.
21. Steger, J., Notes on surface grid generation using hyperbolic partial differential equations. (unpublished report), Dept. Mech., Aero. & Mat. Eng., University of California, Davis, 1989.
22. Steger, J., Notes on composite overset grid schemes — chimera. (unpublished report), Dept. Mech., Aero. and Mat. Eng., University of California, Davis, 1992.
23. Steger, J., Dougherty, F.C., and Benek, J., 1983. A chimera grid scheme *Advances in Grid Generation*, Ghia K.N. and Ghia, U., (Eds.), ASME FED, 1983, Vol 5., pp 59–69.
24. Suhs, N. and Tramel, R., *PEGSUS 4.0 User's Manual.* AEDC-TR-91-8, 1991.
25. Warming, R. and Beam, R., On the construction and application of implicit factored schemes for conservation laws *SIAM-AMS Proc.* 1978, 11: 85–129.