CHAPTER 52

Composite Grids for Flow Computations on Complex 3D Domains*

Ramana G. Venkata†
Joseph Oliger†
Joel Ferziger**

Abstract. Flow situations involving localized phenomena and 3D, complex geometries are very important and are often encountered in engineering applications in the aerospace, chemical and petroleum industries. Such geometries defy attempts to lay a single grid over the entire domain, for numerical solution of the problem using finite-difference methods. In a Composite Grid method, the domain is decomposed into overlapping regions which communicate at their boundaries. Each of these is individually transformed to a discrete, orthogonal parallelepiped grid. The transformed flow equations are then solved on these grids, in conjunction with the other grids which communicate with them, by using any of the wide variety of solvers including adaptive, multigrid versions. In this paper, we will describe the grid generation procedure, the data structure used to create the composite grid and some communication and other design issues.

1 Introduction. Numerical simulation of a flow problem using finite-difference methods is most convenient to implement if the flow domain is rectilinear. For domains which are not rectilinear, a *boundary-fitted* curvilinear grid, consisting of nearly orthogonal gridline families, is desired for accurate application of the boundary conditions. A rectilinear grid in the *computational space* is then mapped to this curvilinear grid in the *physical space* using a *coordinate transformation.* However, it is often not possible to overlay the entire domain with a single grid since we would like to avoid

- nonuniform density of gridlines.
- degeneracies in the transformation, when grid lines belonging to supposedly near-orthogonal families are nearly parallel.
- singularities in the Jacobian of the transformation caused for example, in 2D, by the mapping of a rectangular region in the computational space to a region with a different number of corners in the physical space.

*This work has been supported by the Office of Naval Research under grants N00014-90-J-1344 and N00014-89-J-1815 and by NASA Ames Research Center under Agreement NCA2-440.
†Department of Computer Science, Stanford, CA 94305.
**Department of Mechanical Engineering, Stanford University, Stanford, CA 94305.

These problems can be overcome by decomposing the domain in the physical space into multiple, overlapping regions exchanging information at the boundaries. *Curvilinear parallelepiped*[1] grids are then formed in the physical space by mapping an orthogonal parallelepiped grid in the computational space onto each of these regions. The result of this tessellation of the physical domain is called a *Composite Grid.* The flow equations are also correspondingly mapped using the metrics of the transformation. They are then solved to the desired accuracy on the grids in the computational space, by using any of the dedicated, fast solvers which take advantage of the grids' regular structure. Boundary information is swapped by overlapping grids in the region of overlap, during the global iteration.

This *Domain Decomposition* is therefore necessitated due to the geometrical complexity of the original domain as opposed to the other well-known reasons, such as achieving computational speed-up by use of parallel processors or dividing the domain into zones based on the physical nature of the flow. Of course, it doesn't preclude further subdivisions for computational reasons and the use of parallel processors to obtain speed-up.

Composite Grids were used by Atta and Vadyak[ATT83] to solve the full-potential equation using a composite-adaptive grid approach on a set of overlapping grids. Rai *et al.* describe a composite grid method for unsteady Euler equations using touching grids in [HES86] and for compressible Navier-Stokes equations using overlapping grids in [RAI87]. In incompressible flows, because of the nonconservative form of the continuity equation, conservative exchange of information in the region of overlap becomes difficult[MEA86]. Henshaw[HEN85] describes a Composite Grid method for a time-dependent, two dimensional oceanographic problem. Wijngaart[WIJ89] describes a composite grid method for incompressible flows in two dimensions. This work is an extension to three dimensions of Wijngaart's method.

We will first describe the various constructs that are created in our Composite Grid system to convey the geometric and the boundary description of the flow problem into the system and to enable the grid generation. We will then discuss some related issues that arise due to our use of the multiple grids. Finally, we will briefly describe some associated work in progress at Stanford aimed at tackling the visual editing, and language, needs of a Composite Grid system.

2 Data Structures for the creation of the Composite Grid. Given a 3D, complex domain in the physical space and the flow problem described therein, the *grid generation procedure* consists of the following steps:

- incorporating the geometric data, which constitutes the domain description, into the Composite Grid system
- specifying the various boundary conditions which are applied to the equations that describe the flow problem
- generating the set of discrete grids along with all the associated metrics as well as the boundary information.

2.1 Surfaces. The regions into which the domain in the physical space is divided are called *volumes.* These volumes are defined by their bounding *surfaces*, each of which is a curvilinear quadrilateral. These surfaces in turn are defined either by their bounding *curves* or by a more direct description. A curve is any segment of the boundary contour of the domain, which is parameterized by a single parameter. Thus, a curve is strictly a geometric feature. When a surface is defined by the specification of its bounding curves, a transfinite interpolation of these bounding curves results in a definition of the interior of the surface. Thus knowing the parametric definition of each of the bounding curves (which

[1] A curvilinear parallelepiped is defined as a polyhedral volume with six curvilinear quadrilateral faces, not necessarily lying in parallel planes. The faces don't need to be planar either.

establishes the relation between the curve's parameter and the physical coordinates at every point along the curve) as well as the transfinite interpolation from the boundaries into the interior of the surface, one can establish the relation between the parametric coordinates and the physical coordinates at any point on the surface. However, a surface with important interior features may also be more directly defined, e.g., as a bicubic B-Spline surface. This definition also establishes the same relation as above.

2.2 Curves. From the above description, it is evident that the types of constructs allowed for the definition of the curve determine to a large measure the range of applicability of the Composite Grid system to practical domains. We chose to allow the Bézier family of curves, such as Bézier curves, B-Spline curves and NURBS (Non Uniform Rational B-Splines), as the primary means of representation of curves in our system. We will now briefly describe the properties of these curves and the reasons for our choice[FAR90].

- A Bézier curve is defined by the following[2]
  de Casteljau algorithm:
  Given: $\mathbf{b_0}, \mathbf{b_1}, \ldots, \mathbf{b_n} \in \mathbf{E^3}$ and $t \in \Re$,
  set

$$\mathbf{b}_i^r(t) = (1-t)\mathbf{b}_i^{r-1}(t) + t\mathbf{b}_{i+1}^{r-1}(t) \begin{cases} r = 1, \ldots, n \\ i = 0, \ldots, n-r \end{cases}$$

  and $\mathbf{b}_i^0(t) = \mathbf{b}_i$. Then $\mathbf{b}_0^n(t)$ is the point with parameter value t on the *Bézier curve* $\mathbf{b^n}$.
  The polygon $\mathbf{P}$ formed by $\mathbf{b}_0, \ldots, \mathbf{b}_n$ is called the *Bézier polygon* or the *control polygon* of the curve $\mathbf{b}_n$ and the polygon vertices $\mathbf{b}_i$ are called the *Bézier points.* By virtue of the convex barycentric combinations (all weights are non-negative and sum to one) that the de Casteljau algorithm is solely composed of, a Bézier curve possesses the following properties:
  1. convex hull property - the curve lies within the convex hull of the control points.
  2. affine invariance - the curve is invariant under an affine transformation (involving just translation, rotation, stretching and shear). This feature allows us to avoid redundant specification of curves in the system, since we can use affine transformations of existing curves instead.
  3. symmetry - replacing t by $(1-t)$ has no effect on the curve.
  4. pseudo-local control - when one of the control vertices is moved, though the whole curve changes, but it is mostly affected locally.
- modeling a curve of a complex shape by use of a single Bézier curve requires a representation of high degree, which is undesirable from a computational standpoint. In such cases, we use *spline* curves which are *piecewise polynomial* curves. In particular, a spline curve composed of Bézier curve segments which is specified by using a minimal information set and incorporates the continuity conditions at the junction points is called a *B-Spline.* The B-Spline possesses all of the advantages of the Bézier curve and, in addition, has more localized control along with the ability to represent more complex shapes.
- however, a B-Spline still cannot represent conics, which are a very popular design tool in industry. To represent these and other rational curves, we use NURBS which are built upon the fact that a conic section in $\mathbf{E^2}$ can be defined as the projection

[2] $\mathbf{E^n}$ is the n-dimensional Euclidean (or point) space and $\Re^m$ is the m-dimensional linear (or vector) space

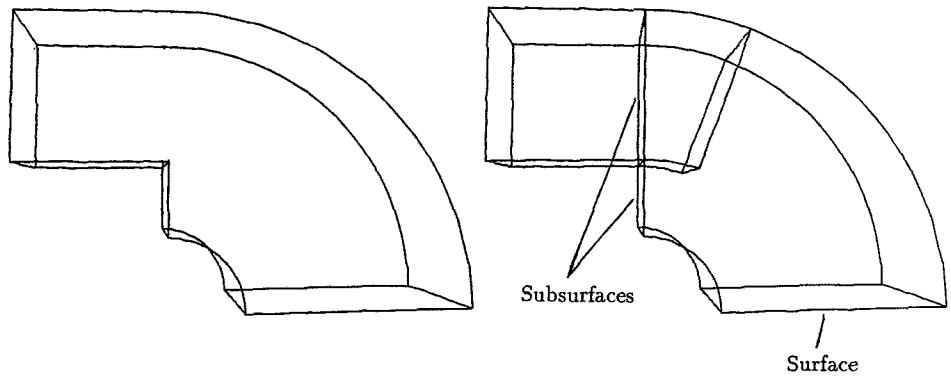

Figure 1: A simple, hypothetical 3D back-step flow domain for concept illustration. Physical Problem - Geometry and Boundary Data

of a parabola in $\mathbf{E}^3$ into a plane. Since a NURB is defined in 3D as the projection through the origin of a 4D nonrational B-spline curve into the hyperplane $w = 1$, it has the ability to represent conics. It also has all the aforementioned properties of Bézier curves in addition to greater localized control.

- The Bézier constructs appear to be a natural choice since these are the representations that are most often used in the CAD industry to design the domains on which we solve our flow problems. However, in addition to the above constructs, we also provide for the non-Bézier parametric polynomial representations of curves, to be used where convenient.

2.3 Subsurfaces. So far, we have only described the means by which geometric domain data is input into the system. The connection to the flow problem is established via the concept of *subsurfaces*. A surface is composed of a tessellation of one or more subsurfaces, each of which has exactly one *physical role* to play in the flow problem. For example, one part of a surface might be a *physical boundary*, while another part could be a *periodic boundary*. Those surfaces, or parts thereof, which are artificially introduced during the division of the domain into volumes, and are not part of the domain boundaries, are given the role of an *auxiliary boundary*.

Thus, a subsurface is defined in terms of its parent surface, the parametric intervals it occupies within the parent surface and its physical role. Associated with each of these roles is a means of describing and enforcing the problem boundary conditions in terms of the variables of the flow problem. The specification of the surfaces and subsurfaces completes the description of the flow problem in the physical space. We now move on to generate the grids replete with the requisite information.

2.4 Faces. The domain in the discrete space is described in terms of *grids*. The grids are defined in terms of their bounding *faces*, each of which is a quadrilateral region, curvilinear in the physical space and rectilinear in the computational space. A grid face is defined as a tessellation of subsurfaces with one continuous parameterization in terms of the parameters of the face. This parameterization is defined by a linear rescaling of the parameterization of the component subsurfaces in terms of the corresponding parent surfaces. The boundaries of the grid face are first parameterized using the parameterization

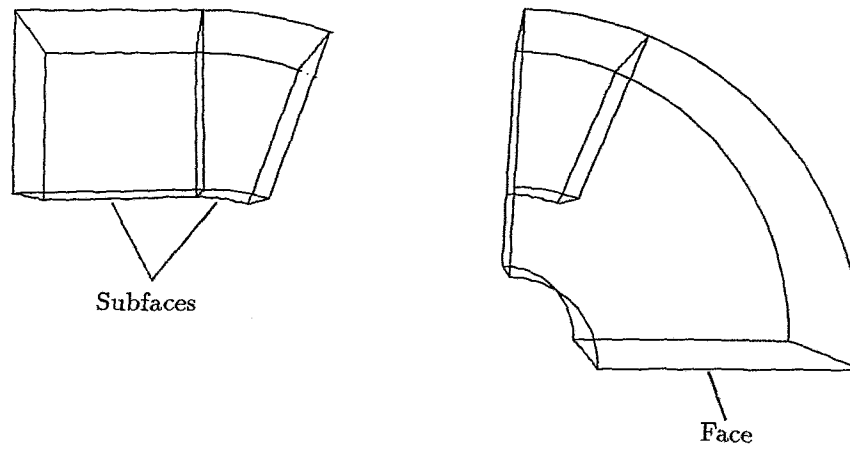

Figure 2: Numerical Problem - Geometry and Boundary Data

of the surfaces on which these boundaries lie. Transfinite interpolation is then employed to first obtain a parameterization of the interior of each face from its boundaries and then of the interior of the grid from its bounding faces. This establishes the coordinate transformation from an orthogonal parallelepiped grid in the computational coordinates (u, v, w) to the curvilinear parallelepiped grid in the physical coordinates (x, y, z). The metrics of the the transformation are also easily computed.

The subsurfaces that comprise a face *need not necessarily* be from the same surface; they just need to be contiguous in the physical space. Also, a subsurface can be a component

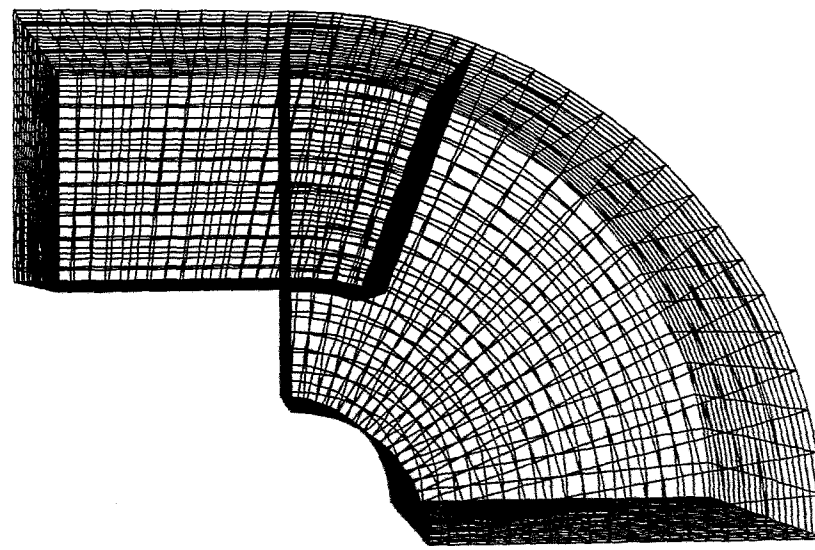

Figure 3: Composite Grid - only gridlines on faces shown

of more than one face, as in the overlap region between two grids. Thus, a subsurface, which imparts the flow boundary information to that geometric entity called the surface, also provides the geometric description of the face. *This duality acts as the conduit of information from the physical problem to the numerical problem in the computational space.*

2.5 Subfaces. So far, we have only provided a means to transfer the geometrical description from the physical space to the computational space. We now introduce the concept of *subfaces* to facilitate the transfer of the flow boundary information to the computational space. The subfaces are contiguous parts of a face, each with exactly one role: *physical boundary, periodic boundary, interpolation boundary etc.*

Thus, if contiguous subsurfaces comprising a face (but from different surfaces) have the same boundary role, they would together form a single subface with that role and the associated means of enforcing that boundary condition, be it applying a physical boundary condition or getting the necessary information from a donor grid. Analogous to a subsurface, a subface is defined in terms of its parent face, the parametric interval it occupies in the parent face and the role it plays. For every grid, we also specify the number of grid cells in each coordinate direction.

2.6 Summary of grid generation constructs. To summarize:

- curves and surfaces are purely geometric features employed to input into the system the geometric description of the domain.
- subsurfaces serve to provide the flow boundary information of the physical problem and the geometrical information to the numerical problem.
- subfaces provide the flow boundary information to the numerical problem.

	physical problem	numerical problem
geometrical info	surfaces	subsurfaces
boundary info	subsurfaces	subfaces

Table 1: Functions of the constructs used in grid generation

3 Other issues related to a Composite Grid. We will now briefly discuss some related aspects that arise from the use of Composite Grids such as specification of the boundary conditions, computing the solution on Composite Grids with communication between grids, specification of the order of traversal of grids, etc.

3.1 Boundary Conditions. As mentioned before, the subsurfaces serve as the vehicle for the description of the boundary conditions. We use *strongly functionally consistent* boundary conditions[WIJ89] to ensure convergence both in the classical and the iterative sense. This implies that we apply the physical boundary conditions at all grid points that lie on the physical boundary of the domain. Thus, for every point on a grid face that lies on a subsurface whose role is that of a physical boundary, we use the relationship between the parameterizations of the face and the surface (to which that subsurface belongs) and obtain the proper boundary condition.

When the boundary condition is periodic, we will need to specify where in space the linear boundary condition operator is to be evaluated. This involves the definition of an affine transformation linking the target and donor points. We can then obtain the boundary conditions at the target point by evaluating the boundary condition operator at the donor point. If the target point is contained in more than one component grid, then additional information is required to choose the appropriate donor grid. Interpolation boundary conditions are enforced in a manner similar to periodic boundary conditions, except that the communication is always between different grids.

3.2 Communication between Grids. To compute the solution on the Composite Grids, we use the traditional technique of *Schwartz Alternating Procedure*[SCH69]. The Schwartz algorithm applied to two overlapping domains Ω_1 and Ω_2 consists of

- assuming the boundary values on the edge of Ω_1 in the overlap region (thus decoupling the problem),

$$
\textit{Composite grid} = \left\{ \begin{array}{l}
\textit{grid}_1 \left\{ \begin{array}{l}
\left[\begin{array}{l}
\textit{face}_{11} = \left\{ \begin{array}{l} \left[\begin{array}{c} \textit{subface}_{111} \\ \cdots \\ \textit{subface}_{11j} \end{array} \right] \\ \left[\begin{array}{c} \textit{subsurface}_{111} \\ \cdots \\ \textit{subsurface}_{11k} \end{array} \right] \end{array} \right. \\
\textit{face}_{12} = \ldots\ldots\ldots\ldots \\
\cdots \\
\textit{face}_{1m} = \ldots\ldots\ldots\ldots
\end{array} \right] \\
(\textit{number of grid cells}(u,v,w))
\end{array} \right. \\
\textit{grid}_2 \left\{ \begin{array}{l}
\left[\begin{array}{l}
\textit{face}_{21} = \left\{ \begin{array}{l} \left[\begin{array}{c} \textit{subface}_{211} \\ \cdots \\ \textit{subface}_{21l} \end{array} \right] \\ \left[\begin{array}{c} \textit{subsurface}_{211} \\ \cdots \\ \textit{subsurface}_{21n} \end{array} \right] \end{array} \right. \\
\textit{face}_{22} = \ldots\ldots\ldots\ldots \\
\cdots \\
\textit{face}_{2m} = \ldots\ldots\ldots\ldots
\end{array} \right] \\
(\textit{number of grid cells}(u,v,w))
\end{array} \right. \\
\textit{grid}_i \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
\textit{surfaces} = \left[\begin{array}{c} \textit{surface}_1 \\ \textit{surface}_2 \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ \textit{surface}_p \end{array} \right]
\end{array} \right.
$$

Figure 4: Hierarchical description of a Composite Grid

- computing the solution in the interior of Ω_1,
- obtaining boundary values on the edge of Ω_2 in the overlap region from the interior of Ω_1,
- computing the solution in the interior of Ω_2,
- using this interior solution to determine the boundary values on the edge of Ω_1,
- repeating the cycle till convergence.

We note that no special routines are needed to solve the interface equations, since boundary values are interpolated directly between grids. The convergence of this algorithm has been analyzed by many researchers. Oliger *et al.*[OLI86] found that the strong dependence of the convergence of the algorithm on the amount of overlap can be reduced for elliptic problems using an overrelaxation technique for the boundary values.

3.3 Global Iteration. The order in which the various component grids are traversed in a global iteration depends upon the direction of information exchange between the grids and thus on the physics of the problem. If the specification of this ordering is manual, an explicit listing of the order is provided. Otherwise, a traversal algorithm is supplied. The ordering also has an effect on the suitability of the whole solution procedure for parallel processing. [OLI86] Oliger *et al.* also introduce a Black-Red scheme that is well-suited for multi-processor machines.

4 Associated Work under progress. We will now briefly describe two of the tools, associated with an Adaptive Composite Grid System, that are currently under development at Stanford.

4.1 VOUS. VOUS[OLI89] is an interactive, object-oriented, graphical editing system that is being developed to provide a tool for the user to input, edit, visualize and manipulate the domain data. The user will be able to input the curves and surfaces necessary for the geometric description, perform any necessary curvefitting to geometric data available, zoom in/out of regions of interest and perform various other manipulations. The hierarchical nature of the composite grids as described above, lends itself to an easy and natural translation to LISP *s-expressions*[PIC90]. The domain can be specified as a hierarchical list where the first atom in each list specifies the object to be described in its sublist. Thus, we have a powerful means of representation of the entire domain for future manipulations. These are the data structures used by VOUS to transfer the domain description that were input by the user during a session to the grid generation package $M * E * S * H$, which will then generate the various grids.

4.2 VORPAL. VORPAL[SUH91], a programming language designed primarily for scientific applications which require interaction and high-level data structures, is being implemented as a precompiler to C. It provides standard data types and operations including essentially all FORTRAN data types as well as several other predefined types, some of which are related to communication and program structure. VORPAL programmers can also define their own data types. Storage management for most types is automatic. Most data structures can be printed or read as a unit either in a textual form which can be incorporated directly into a program source file, in the form of LISP s-expressions, or in a binary form. Since VORPAL produces C code as an intermediate language, compatibly written C procedures as well as FORTRAN programs can be used with little redesign.

5 Conclusions. We have motivated the use of Composite Grids in situations where a single boundary-fitted grid cannot overlay the entire flow domain. We then described the various constructs devised to structure the geometric and boundary data that together describe the domain and the flow - the physical problem - and also the constructs used to transfer this information to the computational space, where the transformed flow equations are solved on regular grids - the numerical problem. Along the way, we motivated the choice of the Bézier family of curves as our representation of choice in the grid generation procedure. We then discussed some of the related issues raised by the use of multiple grids and the communication between them. Finally, we described some of the associated tools under development at Stanford to tackle the visual editing and language needs of the Composite Grid system.

References

[ATT83] ATTA, E. H., VADYAK, J. *A Grid Overlapping Scheme for Flowfield Computations About Multicomponent Configurations*, AIAA Journal, Vol. 21, No. 9, pp. 1271-1277, 1983.

[FAR90] FARIN, G. *Curves and Surfaces for Computer Aided Geometric Design - A Practical Guide*, Academic Press, Inc., San Diego, CA 92101.

[HEN85] HENSHAW, W. D. *Part I: The Numerical Solution of Hyperbolic Systems of Conservation Laws; Part II: Composite Overlapping Grid Techniques*, Ph. D. Thesis, California Institute of Technology, 1985

[HES86] HESSENIUS, K. A., RAI, M. M. *Applications of a Conservative Zonal Scheme to Transient and Geometrically Complex Problems*, Computers and Fluids, Vol. 14, No. 1, pp. 43-58, 1986.

[MEA86] MEAKIN, R. L. *Application of Boundary Conforming Coordinate and Domain Decomposition Principles to Environmental Flows*, Ph. D. Thesis, Stanford University, 1986.

[OLI86] OLIGER, J., SKAMAROCK, W., TANG, W. *Convergence Analysis and Acceleration of the Schwartz Alternating Method*, CLaSSiC Project Manuscript CLaSSiC-86-12, Stanford University, 1986

[OLI89] OLIGER, J., PICHUMANI, R., PONCELEÓN, D. *A Visual Object-Oriented Unification System*, CLaSSiC Project Manuscript CLaSSiC-89-23, Stanford University, 1989

[PIC90] PICHUMANI, R., OLIGER, J., VENKATA, R. *Symbolic Expressions of Composite Grid Structures*, CLaSSiC Project Manuscript CLaSSiC-90-25, Stanford University, 1990.

[RAI87] RAI, M. M. *Navier-Stokes Simulations of Rotor/Stator Interaction Using Patched and Overlaid Grids*, Journal of Propulsion, Vol. 3, No. 5, pp. 387-396, 1987.

[SCH69] SCHWARTZ, H. A. *Über einige Abbildungsaufgaben*, Journal für die Reine und Angewandte Mathematik, Vol. 70, pp. 105-120, 1869.

[SUH91] SUHR, S. *VORPAL, a language for scientific computing* Ph. D. Thesis notes (available in summer), Stanford University, 1991.

[WIJ89] WIJNGAART, R. F. *Composite-Grid Techniques and Adaptive Mesh Refinement in Computational Fluid Dynamics*, Ph. D. Thesis, Stanford University, 1989.