NASA Technical Memorandum 106633

A Method for Flow Simulation About Complex Geometries Using Both Structured and Unstructured Grids

James R. DeBonis
Lewis Research Center
Cleveland, Ohio

July 1994

**National Aeronautics and
Space Administration**

ABSTRACT

A computational fluid dynamics code which utilizes both structured and unstructured grids was developed. The objective of this study was to develop and demonstrate the ability of such a code to achieve solutions about complex geometries in two dimensions.

An unstructured grid generator and flow solver were incorporated in to the PARC2D structured flow solver. This new unstructured grid capability allows for easier generation and manipulation of complex grids.

Several examples of the grid generation capabilities are provided. The coupling of different grid topologies and the manipulation of individual grids is shown. Also, grids for realistic geometries, a NACA 0012 airfoil and a wing/nacelle installation, were created.

The flow over a NACA 0012 airfoil was used as a test case for the flow solver. Eight separate cases were run. They were both the inviscid and viscous solutions for two freestream Mach numbers and airfoil angle of attacks of 0 and 3.86 degrees. The Mach numbers chosen were for a subsonic case, Mach 0.6, and a case where supersonic regions and a shock wave exists Mach 0.8. These test case conditions were selected to match experimentally obtained data for code comparison. The results show that the code accurately predicts the flow field for all cases.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF SYMBOLS

$A+$	van Driest damping constant $= 26$
C_p	pressure coefficient
E	total energy per unit mass
F	inviscid flux vector
G	inviscid flux vector
H	viscous flux vector
I	viscous flux vector
K	Von Karman constant $= 0.41$
Q	state variable vector
R	specific gas constant
S	surface area of control volume
T	temperature
a	speed of sound
c	airfoil chord
e	internal energy per unit mass
p	pressure
q	heat flux
u, v	velocities
x, y	distances
$y+$	distance from wall normalized by shear length scale
ℓ	turbulent length scale
ℓ_o	Thomas turbulence model constant $= 0.09$

γ	ratio of specific heats
ρ	density
τ	shear stress
μ_t	turbulent viscosity
ω	vorticity
ω_c	Thomas model maximum vorticity

Subscripts

i	control volume
k	face of control volume
U	unstructured
S	structured
ref	reference condition
∞	freestream condition

Superscripts

$*$	nondimensional quantity

CHAPTER I.

INTRODUCTION

Computational fluid dynamics (CFD) has been used extensively to predict flowfields within and about numerous geometries. As the technologies for both flow solver and grid generation codes increases, the complexity of the geometries analyzed has also increased. Examples of these complex geometries include complete aircraft, engine nacelles, and mixer/ejector nozzles[1-3]. For configurations such as these generating a satisfactory grid can be a considerable challenge. The time spent in grid generation is in some cases approaching and even exceeding the time needed to achieve a flow solution. Faster and more efficient methods must be developed for CFD analysis of complex geometries.

The purpose of this study is to develop and demonstrate the capabilities of a CFD code which utilizes both structured (quadrilateral) and unstructured (triangular) grids (figure 1) to achieve solutions for complex geometries. This code was developed from existing structured and unstructured flow solvers and an unstructured grid generator. Each grid type has distinct advantages and disadvantages associated with it. By using both grid types this new code will take advantage of the positive features and minimize the deficiencies of each type of grid. The code developed here is for two dimensional problems. This is to demonstrate the concept and show that the method is feasible. Extension of this method to three dimensions, while not trivial should be fairly straightforward.

Other current efforts addressing the problem of complex grid generation and flow solution have taken several different approaches. Structured grid generation

1

algorithms have improved greatly in the past two decades. These improvements usually involve bettergraphical interfaces and geometry definition through the use of computer aided design (CAD) databases. Examples of these codes include Gridgen 3D[4] and RAMBO-4G[5]. The Chimera scheme developed by Steger[6] is a method in which individual structured grids can be coupled within a flow solver to form complex geometries. Weatherill[7] has used a method of composite grids similar to the current study to improve mesh quality through the use of locally unstructured grids in a globally structured grid.

1.1 Structured Grids

Structured grids still are the most widely used type of grid for CFD applications. The structured grid nodes are connected sequentially in both computational directions (figure 2). For any given node, the surrounding nodes are known. Flow solvers for these types of grids can be very efficient because the neighboring nodes needed for differencing are known and do not need to be determined in additional computational steps. Because structured meshes were developed before unstructured meshes, they are more widely used, and the corresponding flow solvers are more technically advanced. The connectivity between nodes also allows for easy and efficient calculation of such quantities as turbulent length scales, cross sectional areas etc. However, the ordered nature of the grid points causes problems in creating and using these meshes. The grid structure must avoid excessive skewness and collapsed cells. For unusual shapes this is not always possible. Grid skewness can affect solution accuracy and convergence[8]. And, if the geometry is multiply connected

or topologically complex, it may be difficult for a single grid type to properly resolve .
Examples of some simple grid types are shown in figure 3. An "O" type grid is
normally used for cylindrical geometries such as airfoils or ducts of circular cross-
section. "H" grids are more suited for rectangular geometries such as wind tunnel test
sections or rectangular channels. An example where simultaneous use of different grid
types would be beneficial is an airfoil in a wind tunnel test section (figure 4). The
region around the airfoil would be best modeled using an "O" or "C" grid. However,
the rest of the test section would be better modeled with an "H" grid.

To help alleviate problems with such geometries, some current flow solvers
have incorporated a method called "grid blocking". Grid blocking breaks up the
domain into smaller easier to generate sections. A mesh for these individual blocks can
usually be created using a single simple grid. One problem associated with grid
blocking is that the flow solver must pass solution information across the interface
between grid blocks. Also, matching two different types of grid together at an interface
may be difficult.

1.2 Unstructured Grids

Unstructured grids are a more recent development designed to make grid
generation feasible for arbitrary geometries. An unstructured grid has no rigid
connectivity enforced upon it. For a given node the neighboring nodes have no known
relationship to it (figure 5). Because an unstructured node is not required to be related
to its adjacent nodes, connectivity must be stored explicitly. In general only the
boundary node locations must be specified in order to create a triangular celled mesh.

Because no fixed type of connectivity is required, problems of grid skewness and conforming to complex geometries are reduced significantly. A problem of unstructured grid generation is the inability to precisely control the grid points in the interior of the grid. It is difficult to cluster grid points in regions where large gradients exist such as boundary layers and shock waves. To address this problem adaptive mesh schemes are sometimes used[9]. However, this adds to the computer resources required. An unstructured grid cell cannot be addressed directly and its neighbors cannot be inferred implicitly. The connectivity information must be stored in an array in the flow solver and explicitly accessed in order to access the data, such as the values of flow variables associated with the particular cell. This makes the unstructured flow solver less efficient and more difficult to write. Calculation of geometric quantities such as normal distance to a wall is also difficult. Also, because this technology is relatively new unstructured code technology is not as mature and readily available as that for structured meshes.

1.3 Composite Grid

The code developed for this study combines features of both structured and unstructured grid types resulting in a "composite grid structure". Because of their efficiency, wide range of capabilities, and ability to control grid clustering, structured grids are used to model the regions near walls and other boundaries. The meshes for these regions can be generated as separate grid blocks without a priori knowledge of how the blocks will interface . A change made to one grid block will not affect the other blocks. An unstructured mesh and flow solver can be used to couple the

structured meshes together (figure 6). This unstructured mesh can be easily and automatically generated given the structured mesh boundaries. This composite grid will allow for flexible grid generation and separate manipulation of the individual grids.

The composite grid code consists of three modules; a structured grid flow solver, an unstructured grid generator and an unstructured flow solver. The three modules needed to construct the composite code were adapted from existing stand alone codes. Because a large part of the flow is still solved on structured grids, the composite grid code was built around the structured flow solver PARC2D[10]. The unstructured grid generator and flow solver are incorporated into the existing architecture of the PARC code. Inputs to the code including iteration control, boundary conditions, time steps, and grid generator options are made through the PARC interface. An effort was made to maintain the PARC interface as much as possible. A user of the current version of PARC could easily begin to use this code.

a. Structured grid   b. Unstructured grid
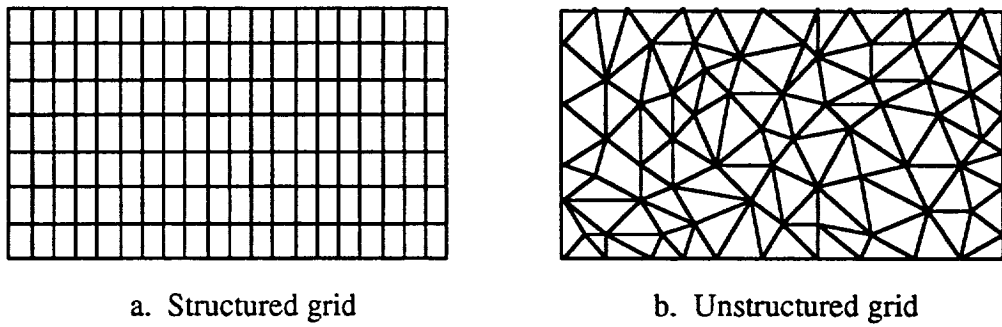
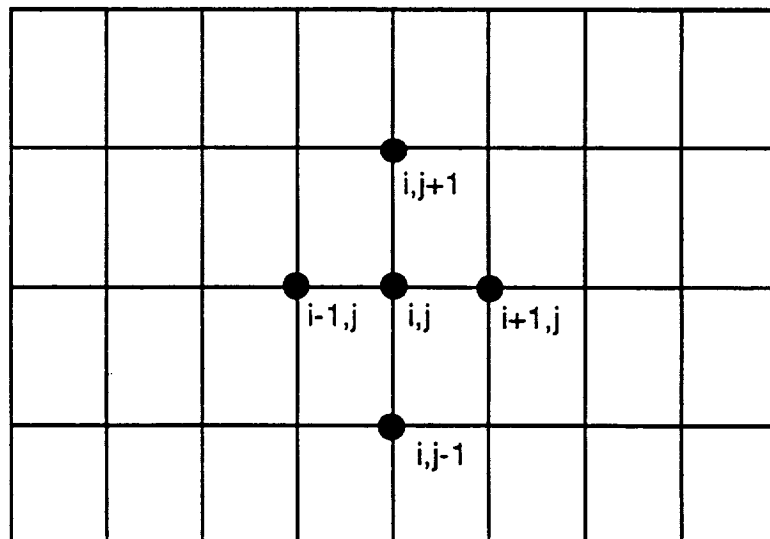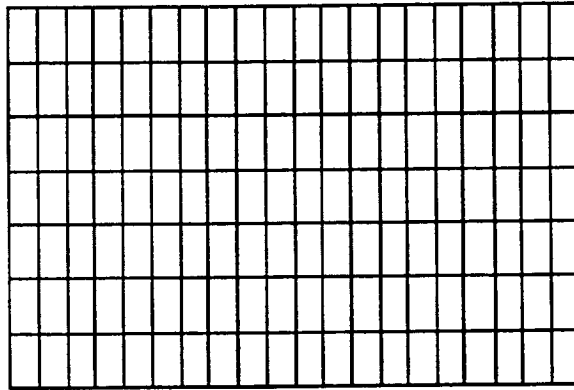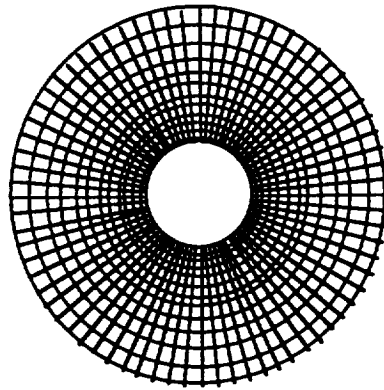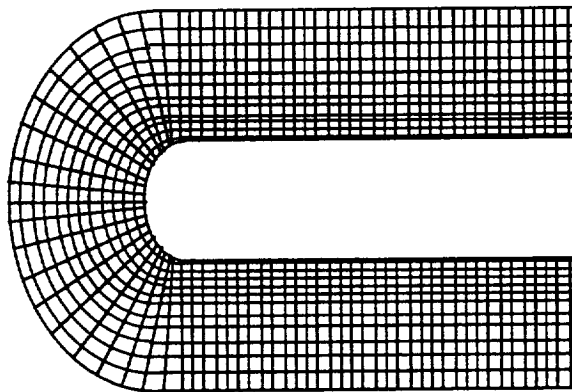Figure 1. Grid examples.

Figure 2. Structured connectivity.

a. "H" grid

b. "O" Grid

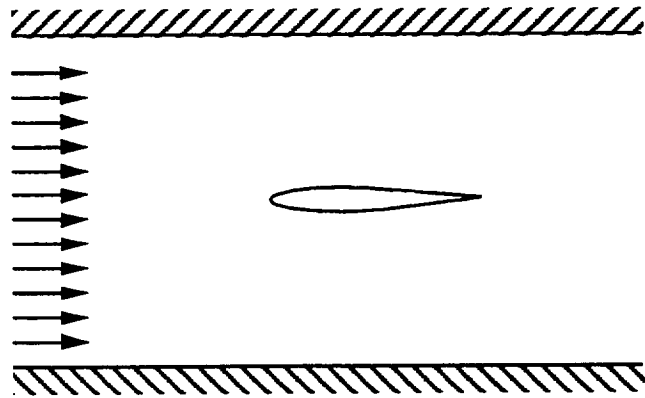c. "C" Grid

Figure 3. Structured grid types.

Figure 4. Airfoil in wind tunnel.
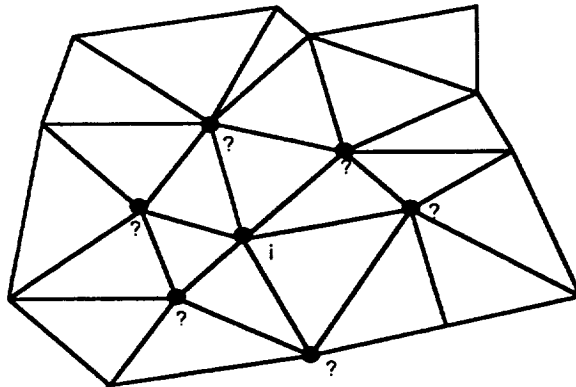
Figure 5. Unstructured connectivity.
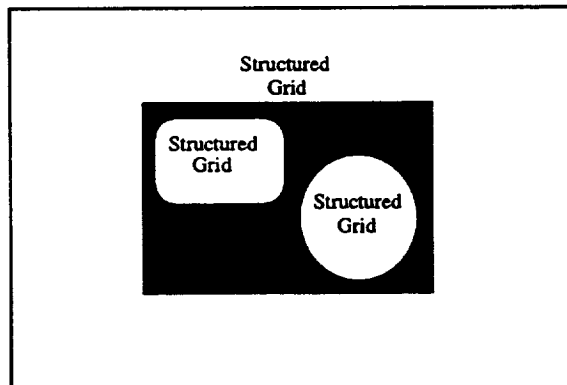
Figure 6. Composite grid concept.

CHAPTER II.

GRID GENERATION

The generation of the composite grid can be divided into structured and unstructured grid generation .

2.1 Structured Grid Generation

The structured grids are generated independently of the composite grid code. This can be done by any of several means available to the user[4,5,11,12]. A separate grid is generated about each object. The connectivity to the other grid blocks can be ignored. This can significantly reduce grid generation time because building grid block interfaces to accommodate the flow solver can be tedious. Most flow solvers require the grid points to be contiguous across a block interface. Others allow for noncontiguous interfaces but grid blocks must overlap into each other such that each point on the interface is inside the adjacent block. The grids are then input into the composite code where the unstructured grid is created.

2.2 Unstructured Grid Generation

The composite grid code constructs the unstructured grid from the structured grid information and the boundary conditions input to the code. The unstructured grid generator module used was adapted from a code developed by Anderson[13]. This code is based on the Delaunay triangulation method[14].

In this method an initial grid is created by specifying a rectangle whose boundaries are larger than the grid to be generated. This rectangle is divided into two triangular cells. Then the known boundary nodes are inserted one at a time into the mesh. After each point is inserted the mesh is retriangulated (figure 7.a). When a point is inserted into the mesh (figure 7.b) all cells are checked to see if the new node is contained within the circle which passes through the cell's three nodes(figure 7.c and d). If the new node is contained within this circumcircle, the cell's nodes becomes part of the Delaunay cavity (figure 7.e). After the all nodes of the Delaunay cavity are determined, the cavity's nodes are reattached to include the new node (figure 7.f). This process of triangulation is repeated for each inserted node. Following the insertion of all the boundary nodes, the aspect ratio of each cell is checked. The aspect ratio is defined as the ratio of the radius of the cell's circumcircle to twice the radius of the largest circle contained entirely inside the cell. If this aspect ratio is larger than a specified tolerance a new node is placed at the center of the cell's circumcircle and the mesh is retriangulated. When all cells meet the aspect ratio criterion, the cells inside all internal boundaries and outside all external boundaries are removed.

In the composite code, the structured boundary conditions specify the nodes that will make up the interface boundary and overlapping region. These nodes are also the boundary nodes for the unstructured grid. The unstructured grid generation module inserts these nodes into the Delaunay cavity as outlined above.
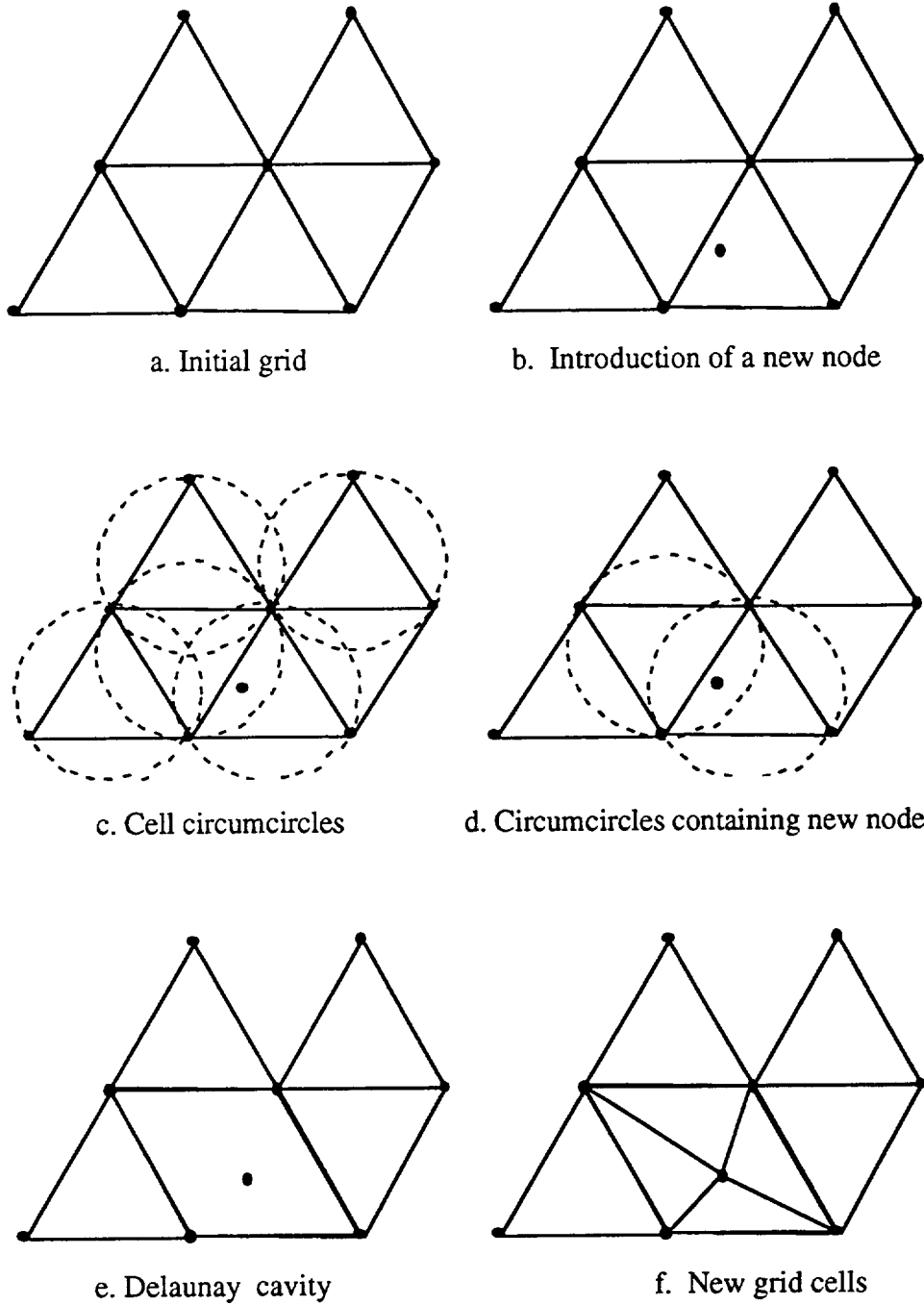
a. Initial grid

b. Introduction of a new node

c. Cell circumcircles

d. Circumcircles containing new node

e. Delaunay cavity

f. New grid cells

Figure 7. Delaunay triangulation.

CHAPTER III.

FLOW SOLUTION

3.1 Structured Flow Solver

PARC2D is a general purpose full Navier-Stokes solver. PARC was developed at the Air Force's Arnold Engineering Development Center for propulsion flows. The code was created from the basic algorithms of the ARC code[15,16]. ARC was developed at the NASA Ames Research Center for external flows. The PARC code is widely used in both government and industry for a wide variety of applications.

PARC solves the Reynolds averaged full Navier-Stokes equations. The equations are solved in strong conservation law form using the Beam-Warming algorithm[17]. Body forces are neglected. The equations are

$$\frac{\partial Q}{\partial t} + \frac{\partial F}{\partial x} + \frac{\partial G}{\partial y} = \frac{\partial H}{\partial x} + \frac{\partial I}{\partial y}$$

where Q is the vector of state variables

$$Q = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{bmatrix}$$

F and G are the following flux vectors

$$F = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ (\rho E + p)u \end{bmatrix} \qquad G = \begin{bmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ (\rho E + p)v \end{bmatrix}$$

12

H and *I* are the viscous flux vectors

$$H = \begin{bmatrix} 0 \\ \tau_{xx} \\ \tau_{xy} \\ u\tau_{xx} + v\tau_{xy} - q_x \end{bmatrix} \qquad I = \begin{bmatrix} 0 \\ \tau_{xy} \\ \tau_{yy} \\ u\tau_{xy} + v\tau_{yy} - q_y \end{bmatrix}$$

where τ is the viscous stress tensor and q is the heat flux vector. Spatial discretization

is done using second order central differencing. Artificial dissipation is added to the

right hand side of the equations for stability. The code has several options for

modeling turbulence. The default turbulence model, the Thomas[18] model is a very

simplistic algebraic mixing length model which is valid for both wall boundary layers

and free shear layers. The turbulent viscosity is defined as

$$\mu_t = \rho \ell^2 |\omega|$$

where $|\omega|$ is the magnitude of the vorticity. In wall bounded regions the length scale is

defined as

$$\ell = Ky\left(1 - e^{\frac{-y^+}{A^+}}\right)$$

where K is the Von Karman constant, y is the normal distance from the wall and the

term in parentheses is the van Driest damping factor. In the free shear layer the length

scale is

$$\ell = \frac{\ell_o\left[Max(|u|) - Min(|u|)\right]}{\omega_c}$$

where ℓ_o is an adjustable constant and ω_c is the maximum vorticity at a given axial

position. The Thomas model was the model used in this study. The Baldwin-Lomax[19]

algebraic model is also available for wall bounded flows. Both algebraic models provide for reasonable results without sacrificing large amounts of computing time. A two equation k-ε model based on Chien's formulation[20] is also available. This model generally provides more accurate results but at the expense of computing time[21].

The code can solve for the flow on domains made up of multiple grid blocks. The interfaces between blocks can be either contiguous (one to one point correspondence) or non contiguous. In the latter case a trilinear interpolation scheme is used to transfer data from one grid to another. The code's most distinguishing feature is the ability to specify any portion of any grid line as a boundary. This gives added flexibility in grid generation, since most flow solvers only allow boundary conditions to be specified on the actual grid boundaries.

3.2 Unstructured Flow Solver

Because the unstructured grid will be used only as an interface, an Euler solver was chosen for the unstructured flow solver. This will minimize the CPU time used in this step of the solution. Use of the Euler equations means vorticity cannot be diffused in the unstructured regions. However, vorticity can be convected through the unstructured regions. This approach was chosen because it is suitable to the purpose of demonstrating the code's capabilities and it simplified the flow solution process. The Euler solver could be replaced by a full Navier-Stokes solver if viscous effects are expected to be important in the unstructured mesh regions of the solution domain.

The unstructured flow solver used, FLO72, was developed by Mavriplis[22].

The code uses a finite volume formulation of the Euler equations. In differential form the equations are

$$\frac{\partial Q}{\partial t} + \frac{\partial F}{\partial x} + \frac{\partial G}{\partial y} = 0$$

where Q is again the vector of state variables

$$Q = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{bmatrix}$$

F and G are the inviscid flux vectors

$$F = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho u v \\ (\rho E + p)u \end{bmatrix} \qquad G = \begin{bmatrix} \rho v \\ \rho u v \\ \rho v^2 + p \\ (\rho E + p)v \end{bmatrix}$$

Integrating over the control volume Ω bounded by the surface $\delta\Omega$ we get the continuous integral form of the equations

$$\frac{\partial}{\partial t} \iint_\Omega Q \, dx \, dy + \int_{\partial\Omega} (F \, dy - G \, dx) = 0$$

For unstructured meshes the control volume can be taken in either of two ways. The first method uses the triangular cell as the control volume. Fluxes are computed across the three faces of the cell. In this method the variables are stored at the cell center (figure 8.a). The second method stores the variables at each node. The control volume is taken as the union of all triangles that have a vertex at a specified node (figure 8.b). Fluxes for each cell face are calculated at both nodes of the face and then averaged. The nodal method was chosen for this study because variables can be

directly transferred from one grid to another without having to interpolate the unstructured variables from the cell centers onto the node points.

The continuous form of the equations are discretized using Roe's flux difference splitting scheme resulting in the following equations

$$\frac{\partial}{\partial t}(S_i Q_i) = -\sum (F_k \Delta y_k - G_k \Delta x_k)$$

where S_i is the surface area of the control volume i and Δx_k and Δy_k are the increments of x and y on face k of the control volume. The finite volume algorithm is a quasi-one dimensional Riemann solver. It treats the interactions between cells as a local Riemann problem with mass, momentum and energy fluxes across a control volume interface determined by the states on either side of the interface. Flux difference splitting calculates the flux across the interface as the average of the flux on both sides minus a wave-based correction. The correction has a stabilizing effect much like conventional artificial viscosity, but which incorporates more information about the actual physics of the flow. Artificial dissipation is necessary to reduce odd-even and shock oscillations.

Because only the basic flow solving routines are necessary for the unstructured portion of the code, a large amount of coding in FLO72 which was extraneous to this study has been removed.

3.3 Grid Interface (Boundary Conditions)

The interface between grid types is done by overlapping the unstructured grid at least one cell deep into the structured grids. A boundary node on one grid corresponds to a cell in the interior of the other grid (figure 9). For both flow solvers the flowfield

variables on the boundary nodes are taken directly from the corresponding nodes on the other grid type. In the composite code, the nodes on the structured mesh needed for the unstructured grid generation are specified in the boundary conditions. Both the boundary and overlapping region are input. The code uses this information to designate the nodes to be inserted into the unstructured mesh generation module. The unstructured nodes are numbered according to the order they are inserted into the grid generator. This allows the code to create a group of arrays which equate the unstructured node's number to its corresponding structured node's indices and grid block.

The boundary condition routines for both the structured and unstructured flow solvers simply use the arrays containing the connectivity between grid types to transfer flow field values from one grid to another. Also, the variables used in the flow solvers are nondimensionalized differently. The boundary condition routines convert the variables from one nondimensionalization scheme to the other. The PARC code uses a dimensional reference pressure, temperature and length (p_{ref}, t_{ref} and x_{ref}) input to the code to define nondimensionalize variables.

$$\rho^* = \frac{\rho}{\rho_{ref}} \quad u^* = \frac{u}{a_{ref}} \quad e^* = \frac{e}{\gamma p_{ref}} \quad p^* = \frac{p}{\gamma p_{ref}} \quad x^* = \frac{x}{x_{ref}}$$

where the reference density and speed of sound are

$$\rho_{ref} = \frac{p_{ref}}{RT_{ref}} \qquad a_{ref} = \sqrt{\gamma R T_{ref}}$$

These reference conditions chosen by the user are arbitrary but should reflect conditions realized somewhere in the flowfield. FLO72, on the other hand, uses the freestream conditions as a reference (p_∞ and ρ_∞).

$$\rho^* = \frac{\rho}{\rho_\infty} \qquad u^* = \frac{u}{\sqrt{\frac{p_\infty}{\rho_\infty}}} \qquad e^* = \frac{e}{p_\infty} \qquad p^* = \frac{p}{p_\infty} \qquad x^* = x$$

The nondimensionalization by freestream conditions in FLO72 is more limiting than nondimensionalization by arbitrary conditions in PARC. In order to maintain a consistent nondimensionalization scheme and be able to convert from one code's scheme to the other's it is necessary to choose the reference conditions in PARC as the freestream conditions.
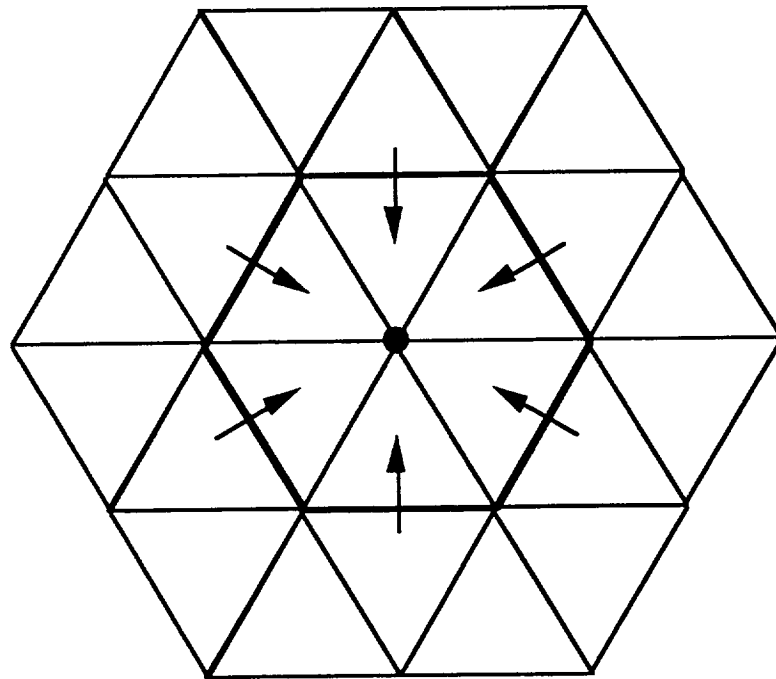
$$p_{ref} = p_\infty \qquad t_{ref} = t_\infty \qquad x_{ref} = 1$$

Then the variables can be converted as follows.

$$\rho_U^* = \rho_S^* \qquad p_U^* = \gamma p_S^* \qquad u_U^* = \sqrt{\gamma} u_S^* \qquad e_U^* = \gamma e_S^*$$
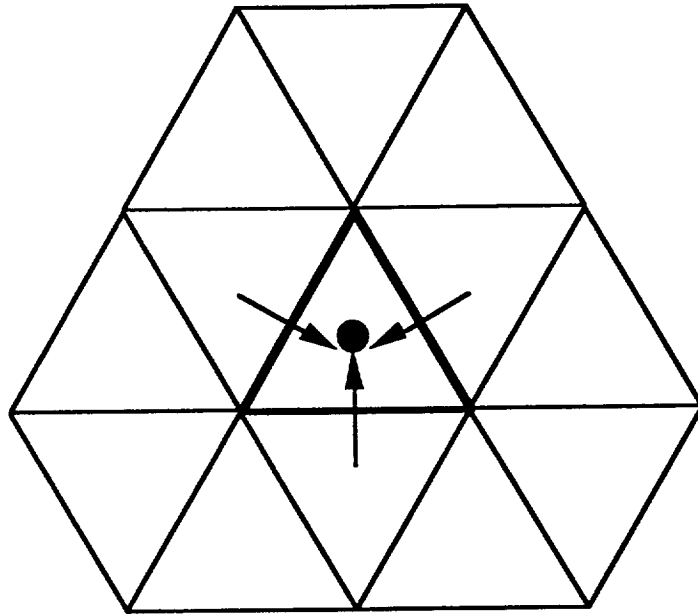
a. Cell centered scheme

b. Node centered scheme
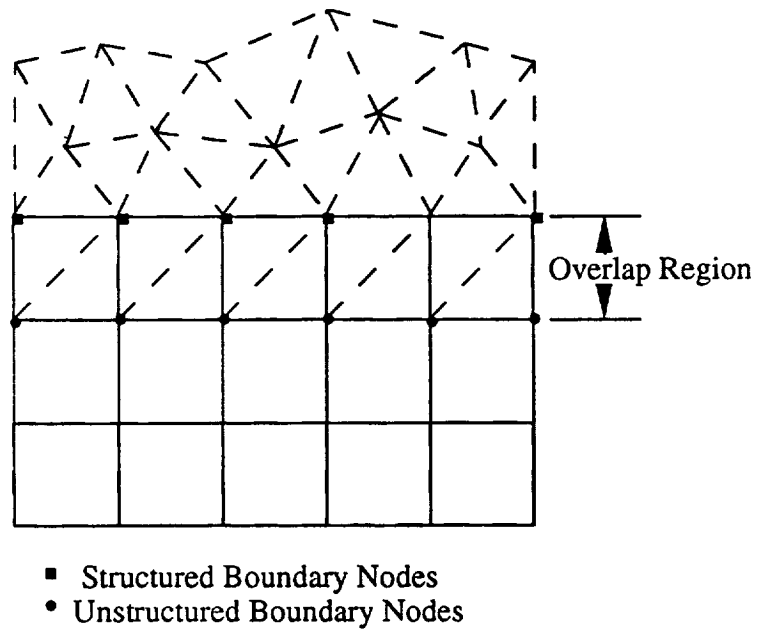
Figure 8. Unstructured control volume schemes.

- ■ Structured Boundary Nodes
- ● Unstructured Boundary Nodes

Figure 9. Grid interface.

CHAPTER IV.

GRID GENERATION EXAMPLES

A series of test cases were conducted to demonstrate the ability of the code to generate and manipulate the structured/unstructured composite grid. For this study a simple rectangular "H" grid is used in the far field. The structured grids were generated using the I3G grid generator[11] developed at Wright Patterson Air Force Base. I3G is an interactive grid generator with a graphical user interface. It can create two dimensional grids or the bounding surfaces required for three dimensional grids. It was run on an Iris workstation.

4.1 Embedded "O" Grid

The first test case is of an "O" grid inside an "H" grid. This case is used to demonstrate the basic concept of the composite grid and the process used to create it. This could have application to such problems as a cylinder or airfoil in a rectangular test section. The two structured grids were generated separately and are shown in figures 10 and 11. Next, the boundary conditions are specified for the interface and overlap grid points. Figure 12 pictures the boundaries of the grids as they will appear when joined together. The composite code is then run. The code reads in the input information including boundary conditions and the structured grid. The unstructured grid is generated as an interface between the two structured grids (figure 13). At the interface, the unstructured grid is generally the structured cell bisected to form two triangles. However, when the cells formed in this manner do not meet the aspect ratio

21

criteria (i.e. when the ratio of the radius of the cell's circumcircle to twice the radius of
the largest circle contained entirely inside the cell is greater than a specified tolerance)
the grid generator adds additional cells in the interface region. This does not affect the
interface because the existing nodes corresponding to the structured nodes are
maintained.

4.2 Embedded "O" and "H" Grids

A second internal structured grid was added to demonstrate the ability to
manipulate the grids. For this case a square "H" grid (figure 14) is embedded with the
circular grid into the far field mesh (figure 15). The two internal grids are coupled
together by the unstructured mesh (figure 16). If the internal grids are translated or
rotated the unstructured grid can be easily regenerated without changing any inputs to
the code (figure 17). This is possible because the information on the position of the
grid points is carried by the grid files. The boundary conditions only specify which
points serve as the unstructured interface. The translation of the structured grids is
done by simply reading in the initial structured grid files and modifying them before
input into the composite code.

4.3 Airfoil

An "O" grid about a NACA 0012 airfoil is shown in figure 18. Because the
"H" grid is to be used in the far field the entire airfoil grid is not used in the composite
grid (figure 19). The interface boundary conditions for the airfoil mesh are specified on

an internal grid surface. The portion not specified as part of the computational domain is simply ignored. When embedded into the rectangular outer grid, the completed mesh represents an airfoil in a wind tunnel test section (figure 20). The airfoil can be easily rotated to any angle of attack in a preprocessing step. The unstructured interface will be regenerated automatically. Figure 21 shows the airfoil at 3.86 degrees angle of attack. The angle of attack 3.86 degrees was chosen to match experimental conditions for code validation.

4.4 Airfoil and Nacelle

A wing/nacelle installation was generated to illustrate the ability of the code to create the grid about a fairly complicated geometry. Of course the flow about the real configuration in this case would be highly three dimensional, so a flow solution on this two dimensional grid would be unrealistic. This case is used only as a grid generation example. Solution of the flow about a wing/nacelle installation is the type of case which would require extending this code to three dimensions. The NACA 0012 airfoil grid was used for the wing. A generic nacelle grid was generated in two dimensions (figure 22). The structured grids are shown in figure 23. Note that the farfield portions of both the airfoil and nacelle grids are deleted in order to put the objects in close proximity to each other. A baseline grid is shown in figure 24. This method allows for efficient study of several perturbations of this design including wing angle of attack (figure 25) and nacelle position (figure 26).

Because the time required to generate a structured grid is dependent on the geometry, the experience of the user and the specific software chosen for the task, it is

difficult to quantify, in a general way, the time that would be saved by using this composite method. For this example however, it is estimated that a person experienced in grid generation could generate the structured airfoil and nacelle grids in several hours. Once the structured grids are generated, the time necessary to interface the grids in the composite code is minimal. If the same case were to be generated using only structured grids, it is estimated that grid generation time would be on the order of days instead of hours.
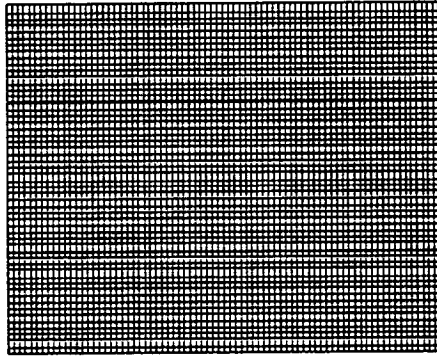
Figure 10. Far field "H" grid.
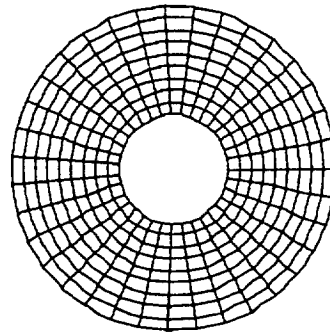
Figure 11. Interior "O" grid.
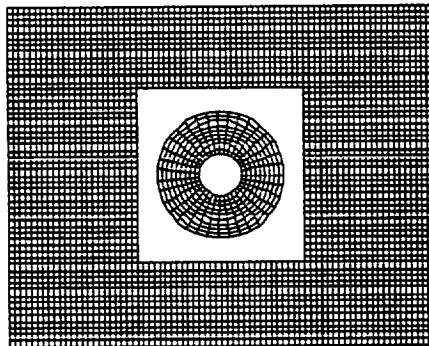
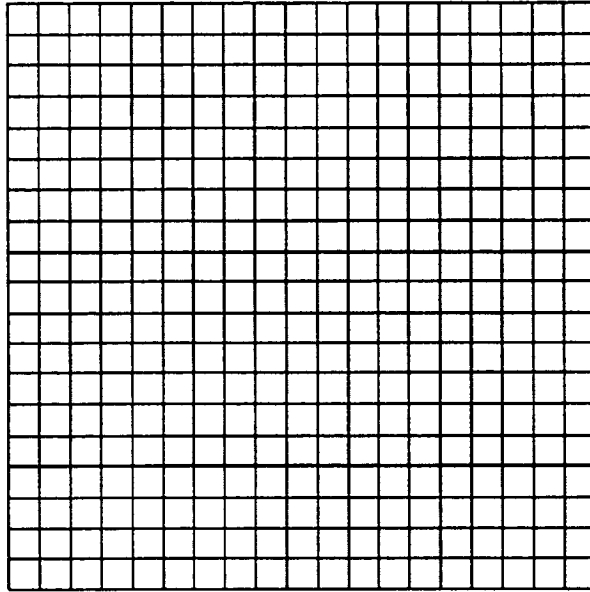Figure 12. Embedded structured grid.
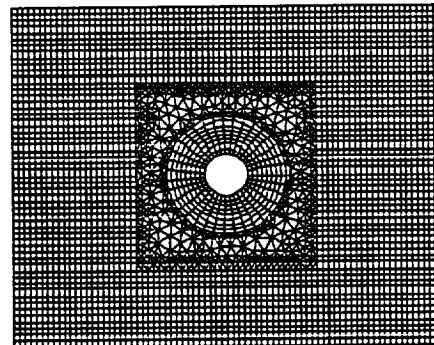
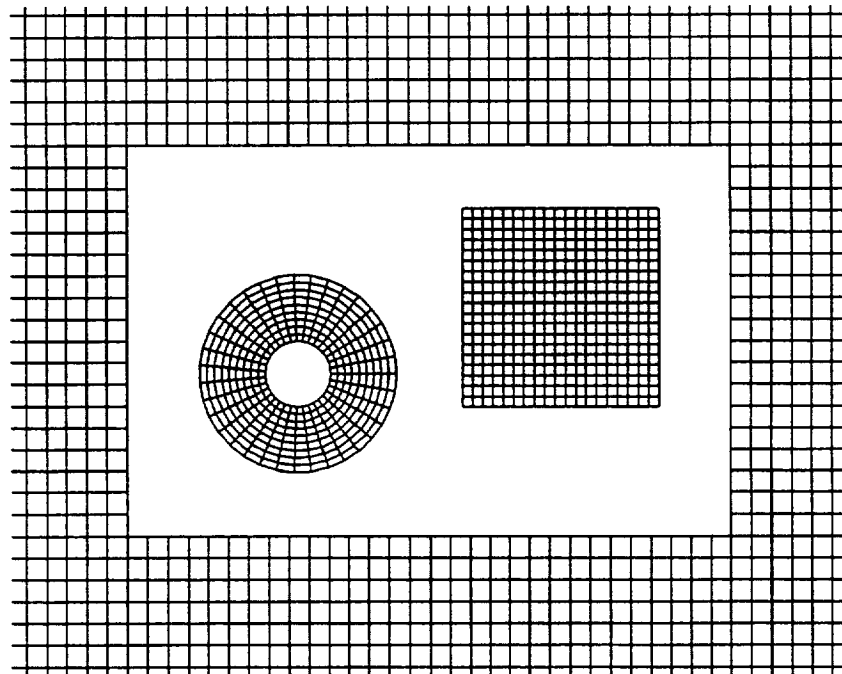Figure 13. Composite grid.

Figure 14. Interior "H" grid.

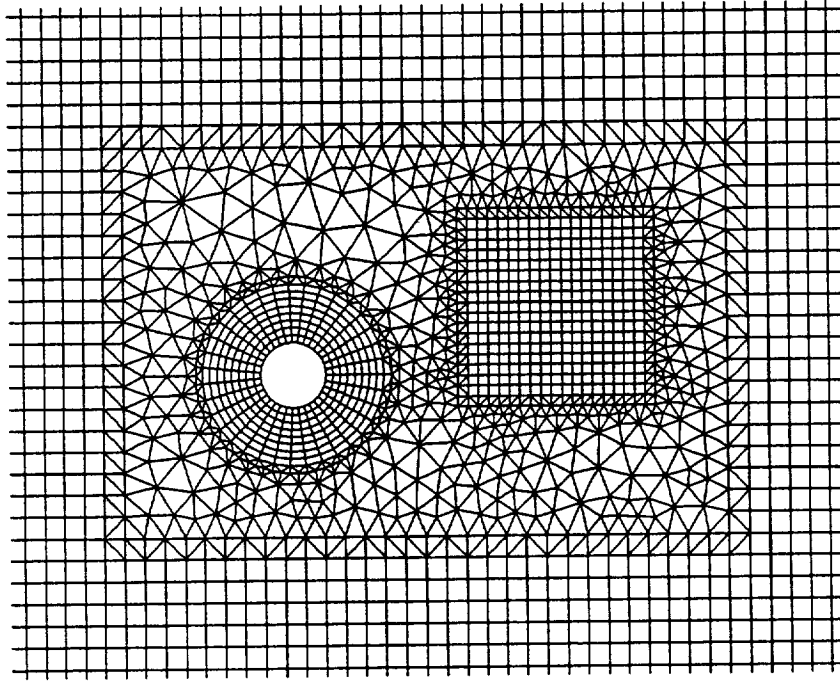Figure 15. Embedded structured grids.

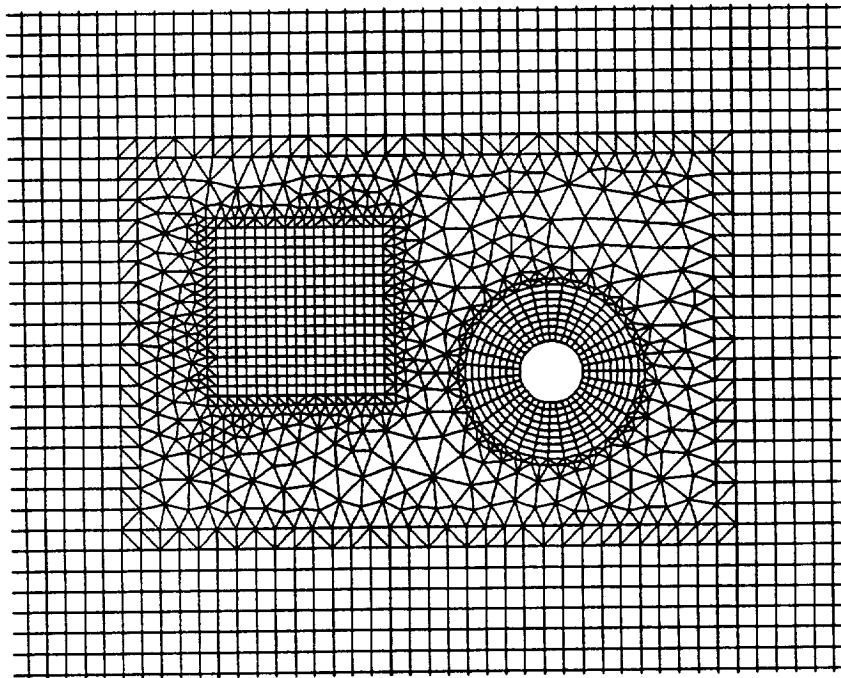Figure 16. Composite grid joining unlike grid types.

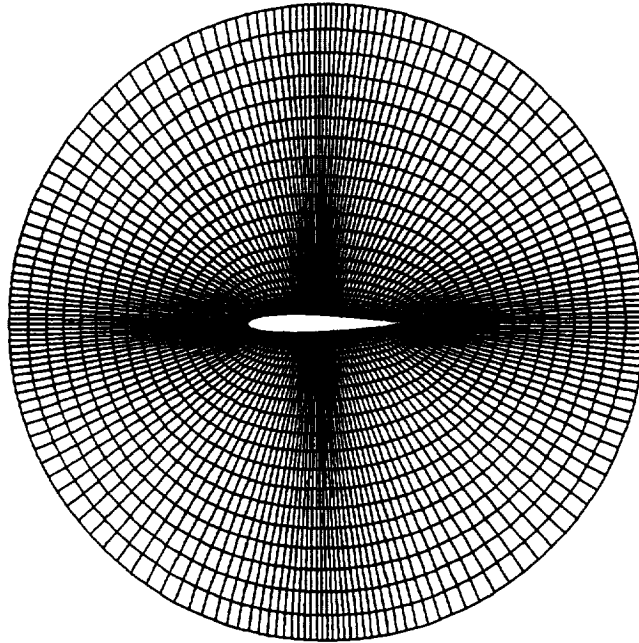Figure 17. Composite grid joining translated grid blocks.

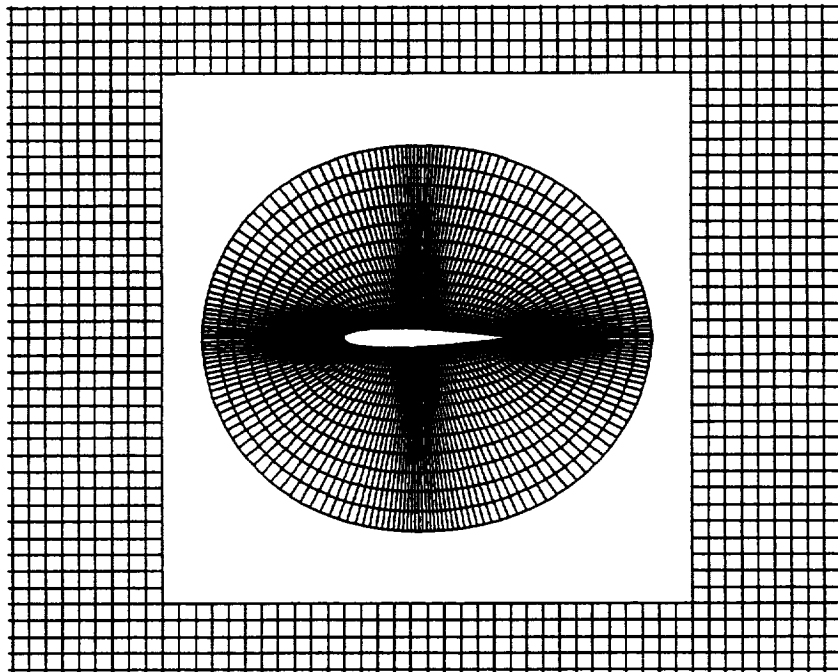Figure 18. NACA 0012 airfoil, "O" grid.
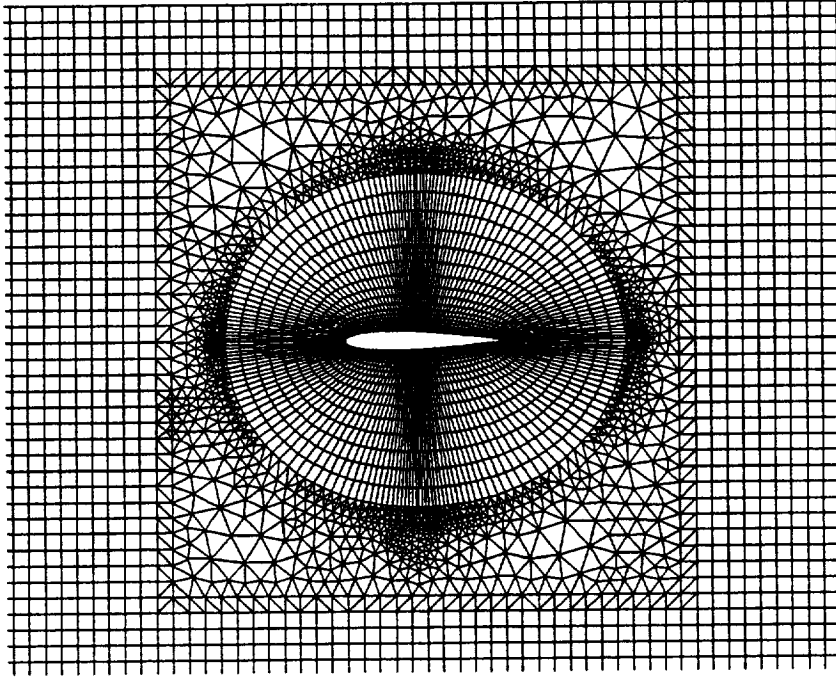

Figure 19. Airfoil embedded in "H" grid.

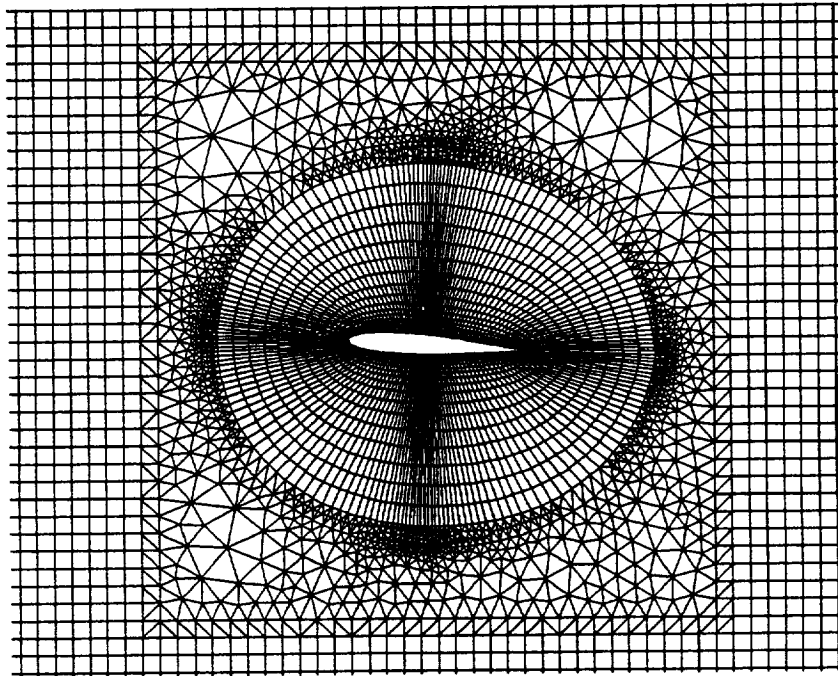Figure 20. Composite airfoil grid.

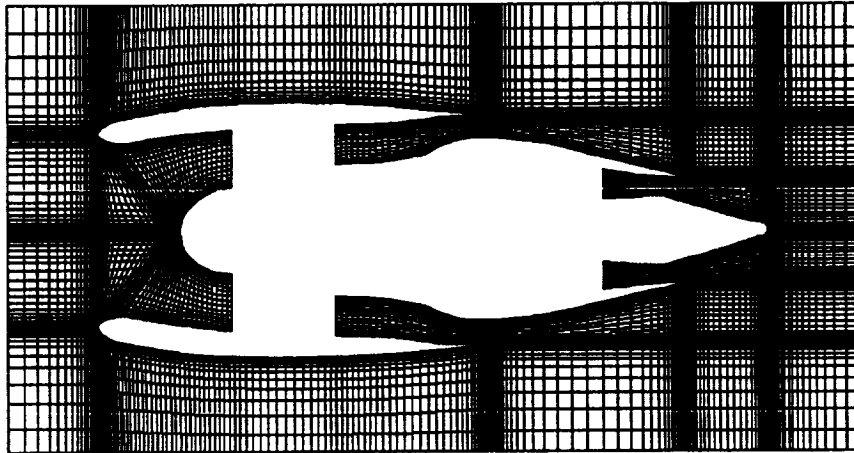Figure 21. Composite airfoil at angle of attack grid.
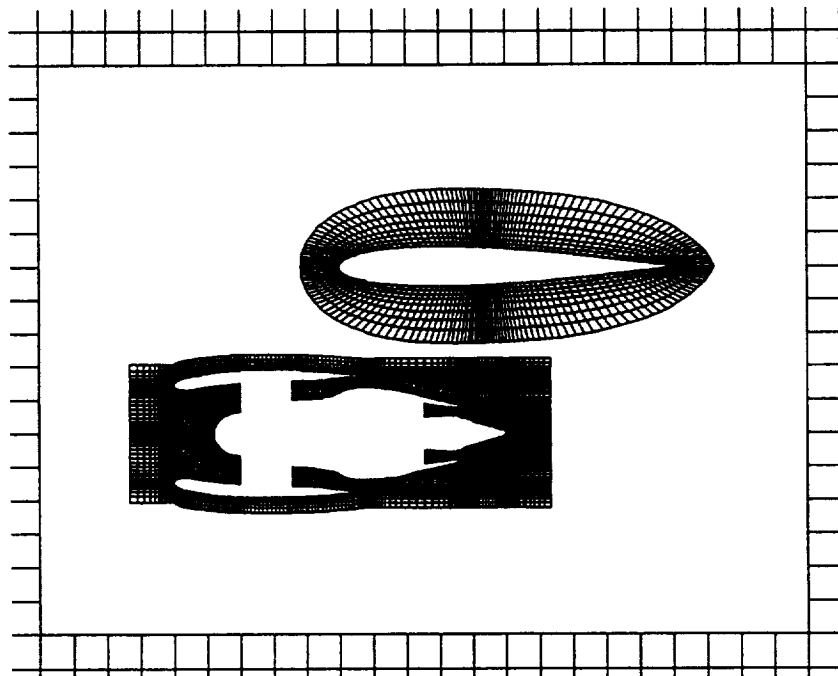
Figure 22. Structured nacelle grid.

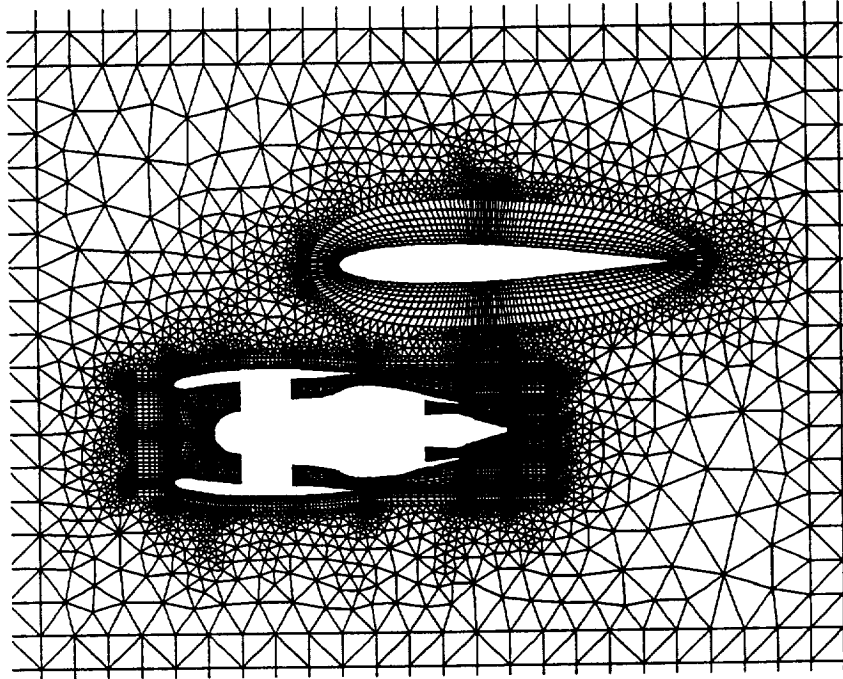Figure 23. Embedded structured grids in installed configuration.

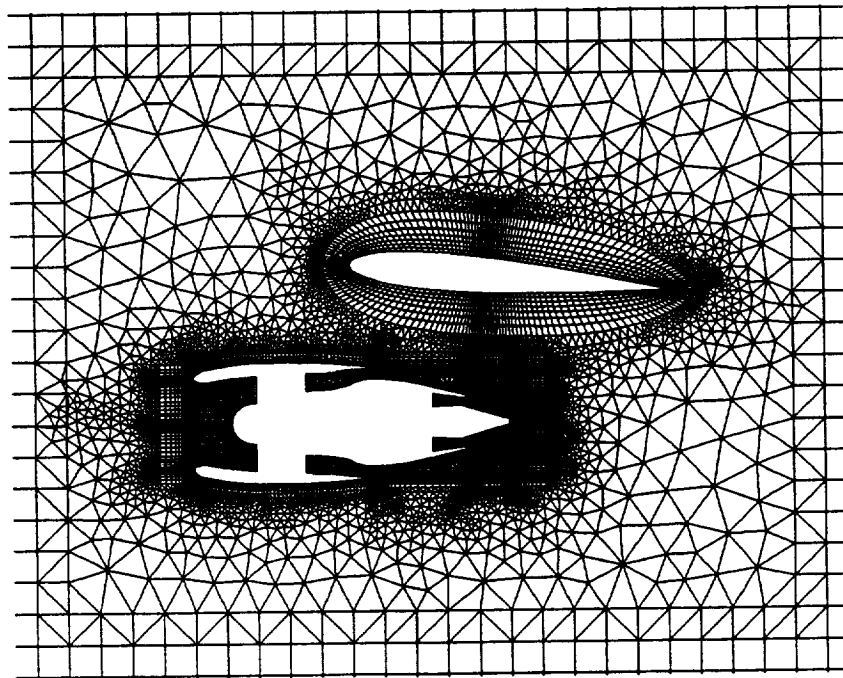Figure 24. Composite grid of nacelle installation.

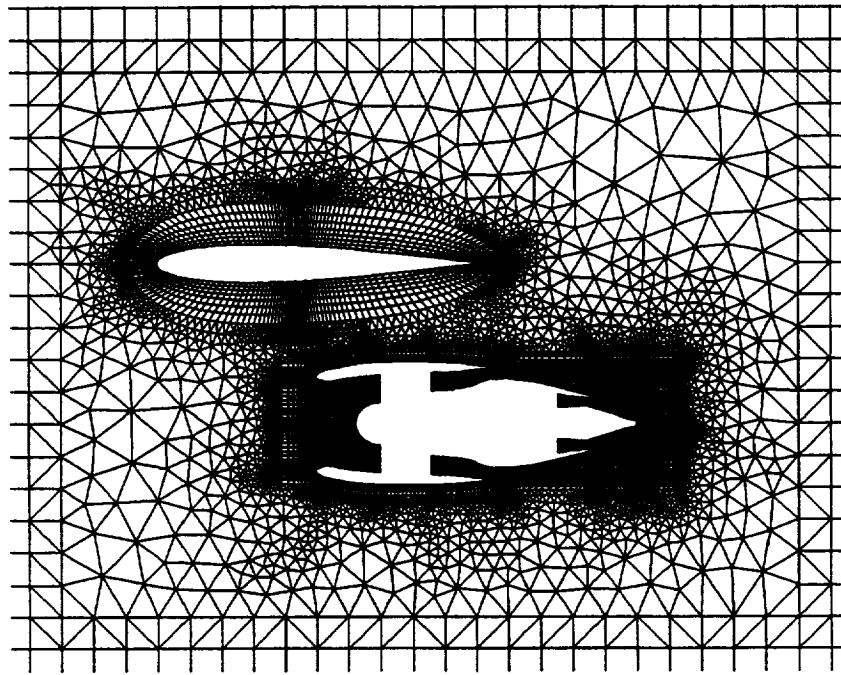Figure 25. Nacelle Installation with airfoil at angle of attack.

Figure 26. Composite grid with translated nacelle

CHAPTER V.

FLOW SOLUTIONS

The flow over a NACA 0012 airfoil was used as a test case for the composite flow solver. A total of eight cases consisting of inviscid and viscous solutions for two freestream Mach numbers and two angles of attack were run. Table 1 summarizes these cases. The freestream Mach numbers used were 0.6 and 0.8. Solutions were obtained for both 0 and 3.86 degrees angle of attack. All computations were done on the NASA Lewis Research Center's Cray Y-MP computer.

The computational results are compared to the experimental data of Harris[23]. These data were obtained in the NASA Langley 8-foot Transonic Pressure Tunnel. The data used for comparison to this code were obtained at a Reynolds number of 9.0×10^6. The boundary layer transition point was fixed at 5 percent of the airfoil chord using a thin band of carborundum grains attached to the surface with lacquer.

The grids used for the inviscid cases are the same as those given in the grid generation examples (figures 18-21). The airfoil grid consists of 178 points in the circumferential direction and 25 points in the radial direction. The viscous airfoil mesh is shown in figures 27 and 28. This grid contains 35 grid points in the radial direction. The additional grid points are necessary to resolve the strong velocity gradients in the boundary layer. For all cases the far field grid measured 101 x 81 points. The unstructured grid size was determined by the structured grid configuration, and depended on both airfoil grid size and angle of attack. Table 2 list grid sizes for each of the eight flow cases.

The total pressure and temperature were specified on the inflow boundary.

Static pressure was specified at the outflow. The upper and lower boundaries were specified as slip walls (no velocity component normal to wall). For the viscous cases the flow was assumed to be turbulent over the entire airfoil. The Thomas turbulence model was used because of its speed and ability to model both wall boundary layers and shear layers.

5.1 Mach 0.6, 0 Degrees Angle of Attack

Pressure coefficient on the airfoil surface is plotted versus the distance along the airfoil chord in figure 29. Because the NACA 0012 is symmetric, at zero degrees angle of attack the pressure distribution is identical on the upper and lower surfaces. The flow stagnates at the airfoil leading edge and then expands rapidly over the first 10 to 20 percent of the airfoil. The pressure then recovers to nearly the freestream value over the aft portion of the airfoil. The results of both calculations agree well with the experimental data. The proper trend is seen between the viscous and inviscid solutions. The displacement thickness of the boundary layer in effect increases the thickness of the airfoil. This means that an inviscid calculation, which does not have a boundary layer, should underpredict the expansion and the recompression of the flow compared to the viscous calculation and the experimental data. However, the Euler solution better matches the data. This is somewhat surprising because the Euler equations neglect all viscous effects and therefore do not model the displacement caused by the boundary layer. The inviscid solution here actually slightly over predicts the expansion and recompression.

5.2 Mach 0.6, 3.86 Degrees Angle of Attack

At angle of attack the flow over the upper and lower surfaces are no longer symmetric. The flow over the upper surface undergoes a very large expansion up to 10 percent airfoil chord. It then slowly compresses back to freestream conditions. The pressure on the lower surface expands from the stagnation point to near freestream pressure along the entire airfoil. Agreement with the experiment for both cases is very good (figure 30). On the upper surface, the expansion for the viscous solution is too large, indicating that the boundary layer is to thick on the leading edge, but the pressure recovery agrees well. Predictions for the lower surface both agree very well with the data.

In theory, calculations of inviscid airfoils at angle of attack require that the Kutta condition be imposed at the airfoil trailing edge. However, for the Euler solutions presented here it is not necessary. Because the artificial dissipation added to the right hand side of the equations ensures that the Kutta condition will be satisfied. However, the added dissipation is sufficiently small so that it does not adversely affect the solution in the rest of the flowfield.

5.3 Mach 0.8, 0 Degrees Angle of Attack

At a Mach number of 0.8 the flow is transonic. At 0 degrees angle of attack the flow over both surfaces accelerates to supersonics speeds and a shock wave is formed at approximately 50 percent chord. Both the viscous and inviscid solutions agree fairly well with the data (figure 31). The only difference between the results is near the

shock. Because of the lack of a boundary layer in the inviscid solution there is supersonic flow at the wall and therefore the shock intersects the airfoil surface. The surface pressure distribution shows a very sharp pressure change due to the shock. In the viscous case the flow near the wall is subsonic. The shock wave does not "sit" right on the wall. Therefore the wall pressure change is more gradual in the viscous solution. The calculation shows that the pressure change is smaller and more diffuse than in the experiment. This may indicate that the calculated boundary layer is too thick or that the shock is being smeared by the combination of real and artificial viscosity

5.4 Mach 0.8, 3.86 Degrees Angle of Attack

The pressure distribution for 3.86 degrees angle of attack is shown in figure 32. For the Euler solution, agreement with the experimental data is poor, especially with respect to the computed shock position on the upper surface and recompression of the flow at the trailing edge of the airfoil. This is due to the absence of the viscous boundary layer in the calculation. The presence of the boundary layer increases the effective airfoil thickness and reduces the amount of recompression on the airfoil surface. This lower pressure on the aft portion of the airfoil causes the shock to occur at a lower Mach number, hence at a location closer to the leading edge. The full Navier-Stokes calculation considerably improves the shock location and the calculated pressure distribution.

5.5 Density Contours

The inviscid solution at Mach 0.8 and 0 degrees angle of attack was chosen as a representative case. Contours of constant density for this case are presented in figure 33. The contours clearly show all the major features of the flow field including the leading edge stagnation point, the supersonic expansion and shock. The contours are symmetric about the chord line indicating that the flow solver has preserved the symmetry of the solution. The boundaries between the structured and unstructured grids are also shown. The contours lines are smooth and continuous through the grid boundaries. This indicates that the grid interface is working properly and has little effect on the solution.

Case Number	Mach Number	Angle of Attack	Solution
1	0.6	0.00	Inviscid
2	0.6	0.00	Viscous
3	0.6	3.86	Inviscid
4	0.6	3.86	Viscous
5	0.8	0.00	Inviscid
6	0.8	0.00	Viscous
7	0.8	3.86	Inviscid
8	0.8	3.86	Viscous

Table 1. Flow solution cases

	Structured Grid		Unstructured Grid	
Case	Circum-frential	Radial	Nodes	Cells
1	178	25	1478	2644
2	178	35	1488	2664
3	178	25	1469	2626
4	178	35	1480	2648
5	178	25	1478	2644
6	178	35	1488	2664
7	178	25	1469	2626
8	178	35	1480	2648

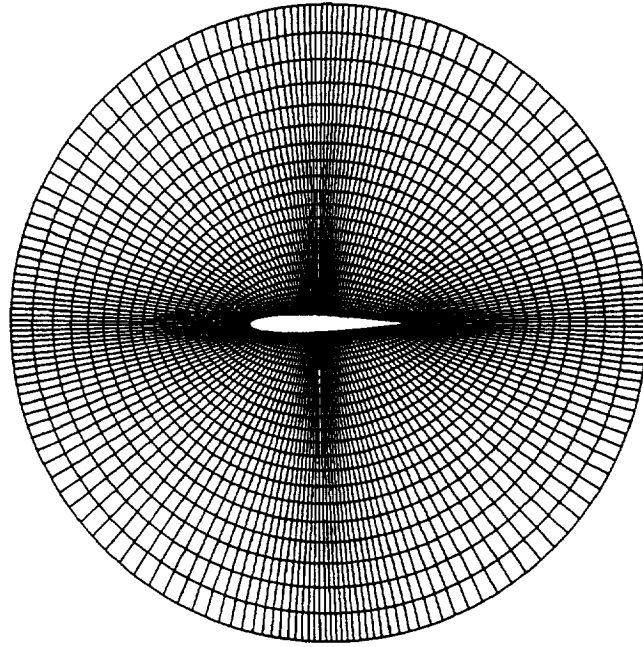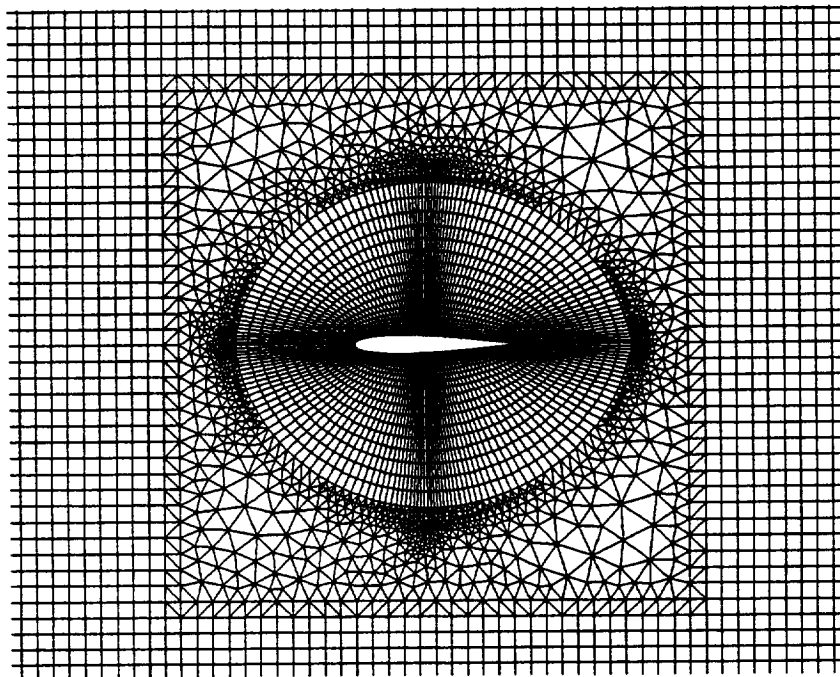Table 2. Grid sizes for flow solution cases

Figure 27. Viscous airfoil mesh.

Figure 28. Composite grid for viscous airfoil calculations.
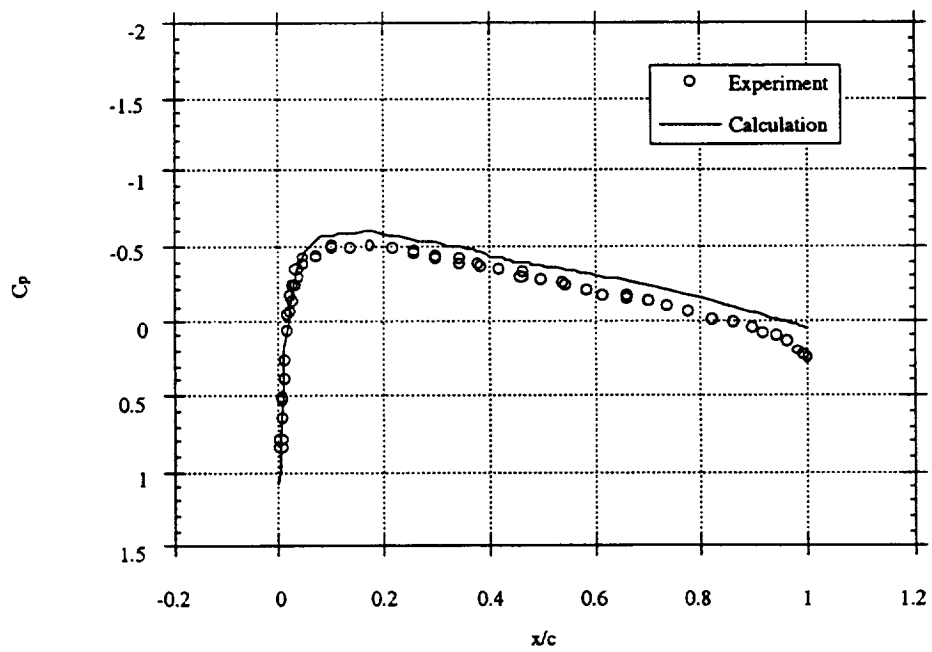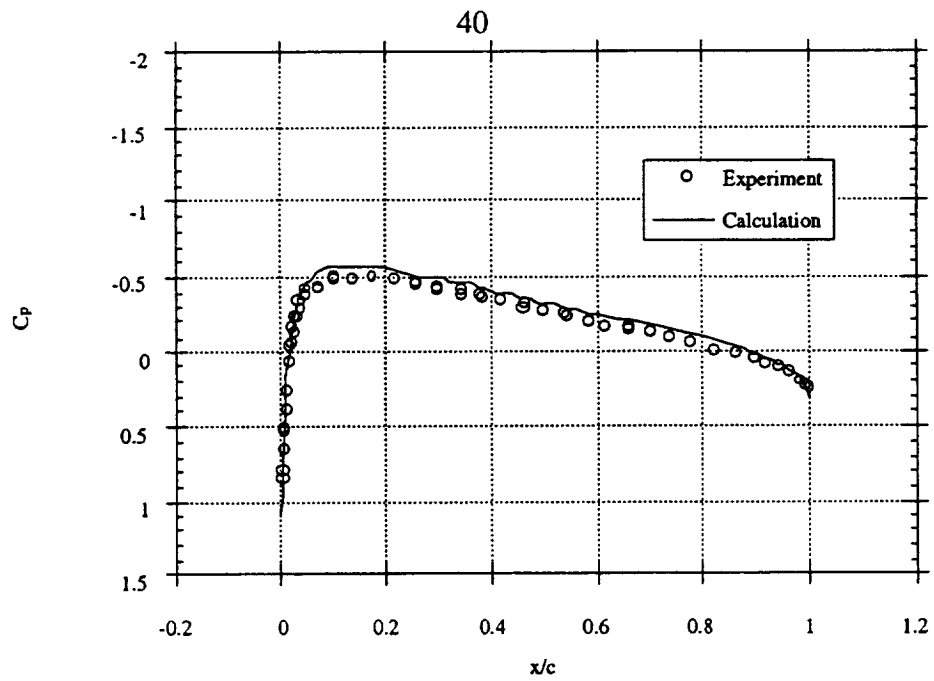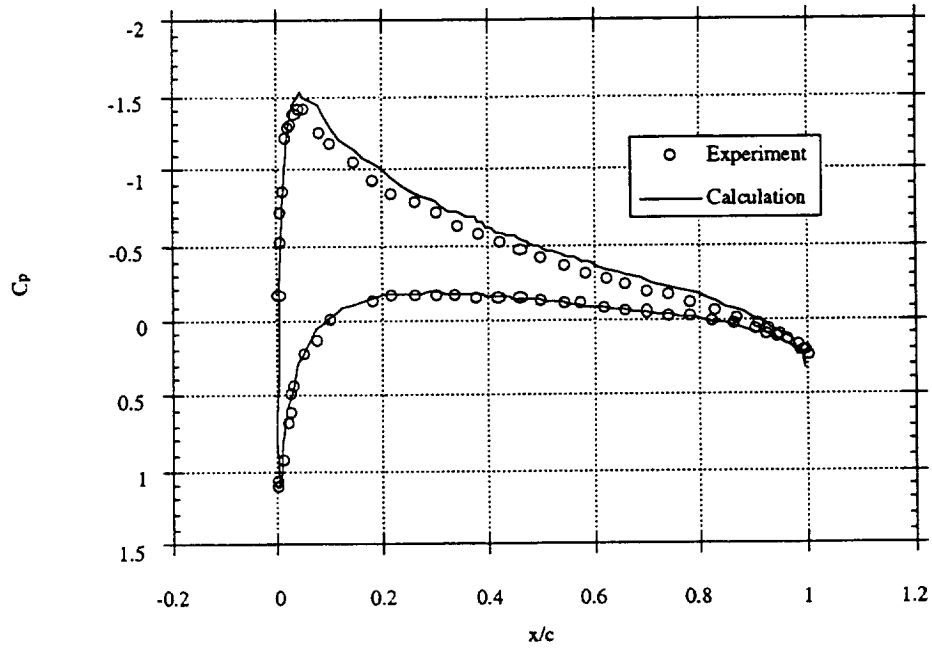
a. Inviscid solution

b. Viscous solution

Figure 29. Pressure distribution for a NACA 0012 airfoil
at Mach 0.6, 0 degrees angle of attack.

a. Inviscid solution

b. Viscous solution

Figure 30. Pressure distribution for a NACA 0012 airfoil
at Mach 0.6, 3.86 degrees angle of attack.
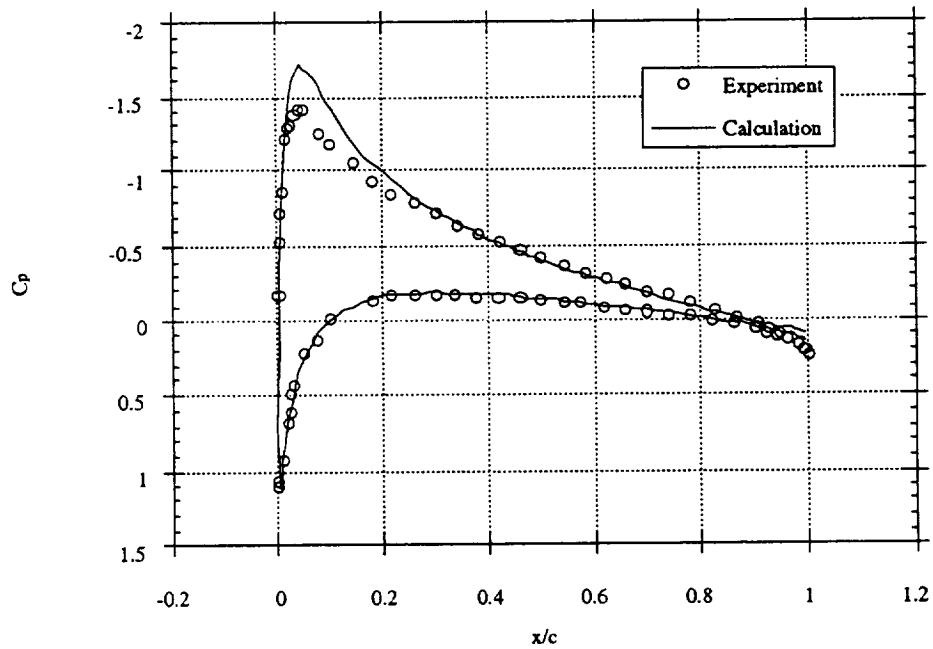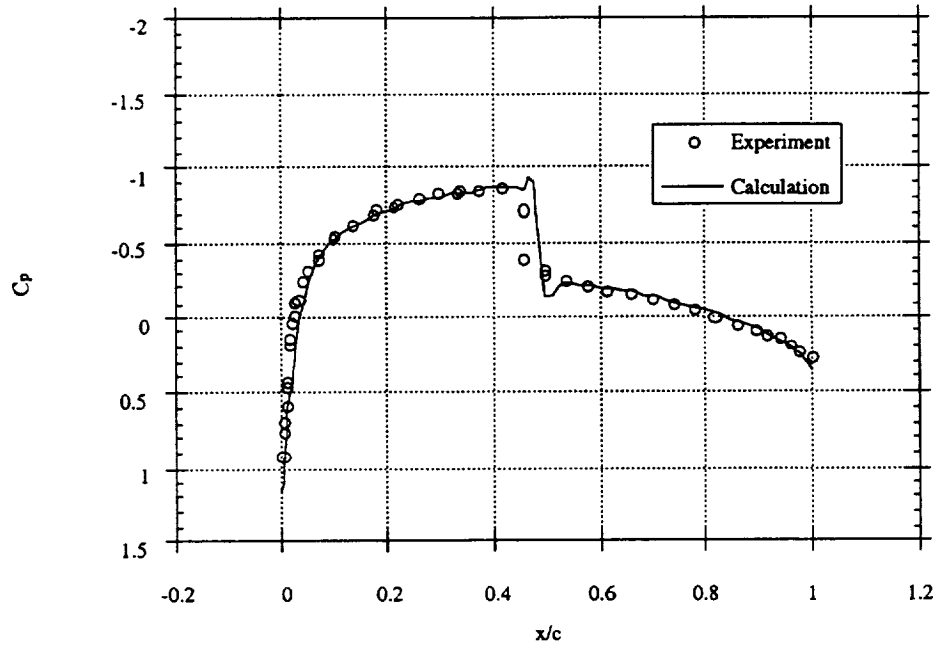
a. Inviscid solution
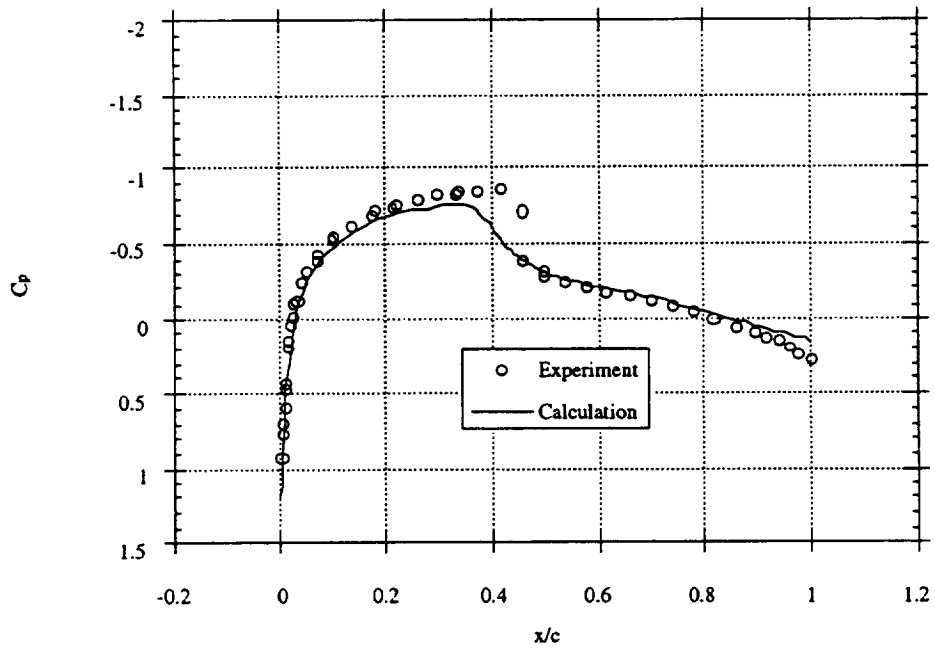

b. Viscous solution

Figure 31. Pressure distribution for a NACA 0012 airfoil
at Mach 0.8, 0 degrees angle of attack.

a. Inviscid solution

b. Viscous solution

Figure 32. Pressure distribution for a NACA 0012 airfoil
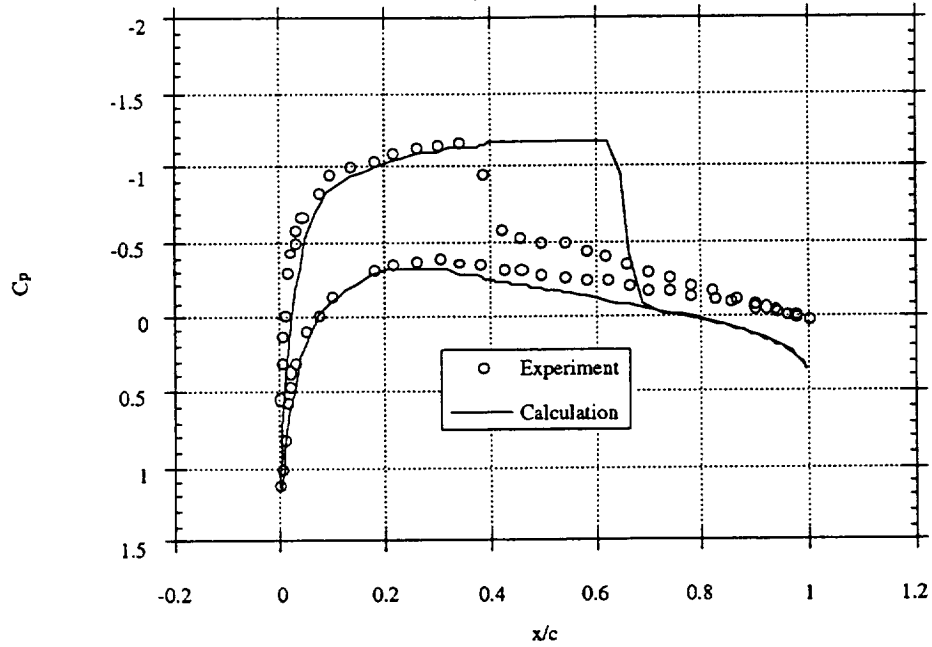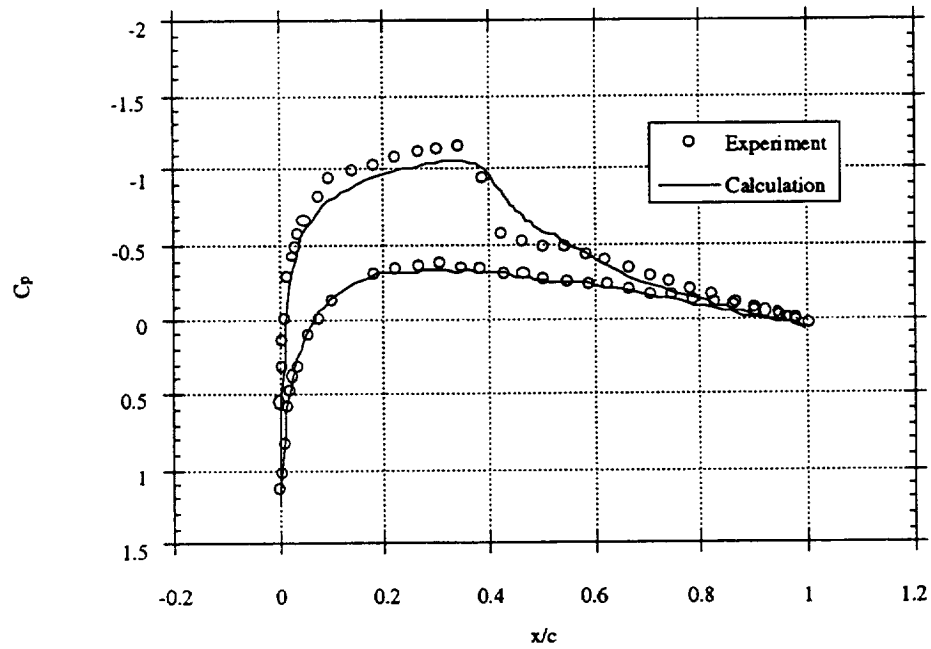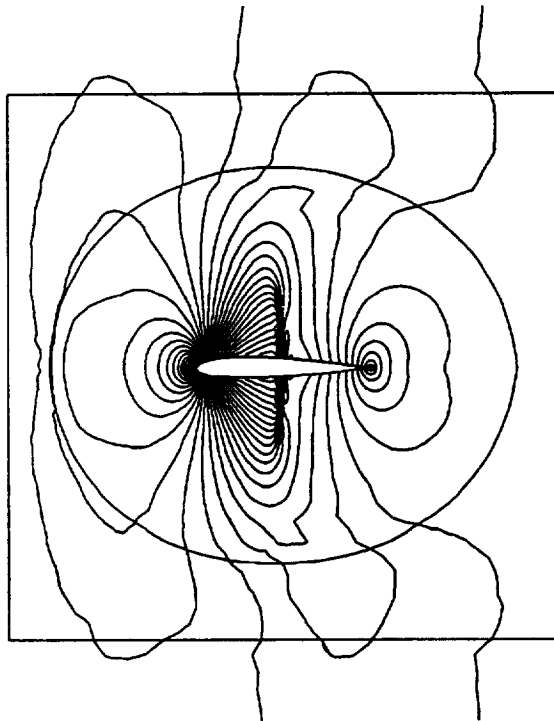at Mach 0.8, 3.86 degrees angle of attack.

Figure 33. Density Contours for a NACA 0012 airfoil
at Mach 0.8, 3.86 degrees angle of attack, inviscid solution.

CHAPTER VI.

SUMMARY AND CONCLUSIONS

A new method applying computational fluid dynamics to solve flows about complex geometries in two dimensions was developed. This method incorporates both structured and unstructured grids and flow solvers to form a composite mesh. The code was developed from existing structured and unstructured flow solvers and an unstructured grid generator. This composite meshing has two advantages. First it eases the task of grid generation. Second it also allows for greater flexibility in manipulating and modifying existing grids.

A composite mesh consists of individual structured grid blocks modeling the areas of interest. An unstructured grid is created to couple the structured grids together. The composite code is made up of a structured grid flow solver, an unstructured grid flow solver and an unstructured grid generator. Generation of the structured grid is done externally to the composite grid code. The unstructured portions of the mesh are automatically generated by the code from the structured boundary nodes and the specified boundary conditions. The grid generator uses Delaunay triangulation method to place the structured boundary nodes into the unstructured mesh and triangulate the mesh.

Several examples of this grid generation are shown. Two simple examples are given to demonstrate the grid generation process and to show the ability to easily manipulate existing grids. Grids for a NACA 0012 airfoil and a nacelle/wing installation are also generated to show realistic flow cases where this method could be used. These grids can be easily manipulated by rotating and translating the individual

structured grids illustrating the ability to vary parameters of the grid quickly and easily.

The structured flow solver used in the code is PARC2D. The structure of the PARC code is maintained and the unstructured solver and grid generator are incorporated into it. PARC2D is a full Navier-Stokes flow solver and is used to solve the flow in all major areas of interest. The unstructured flow solver FLO72 is an Euler solver and its main purpose to couple the structured blocks together by solving the flow field in between the structured blocks.

The flow about a NACA 0012 airfoil was used as the test cases for the flow solvers. Eight different cases were run. The cases included both viscous and inviscid solutions for two freestream Mach numbers and two angle of attacks. Pressure distributions on the airfoil were compared to experimentally obtained data. Generally agreement between calculation and experiment was very good for all cases. This shows that the use of a composite grid has no adverse affect on the flow solution.

The grid generation examples and flow solutions have shown that this method of using composite grids is a viable means for fast, easy and accurate solutions to complicated geometries in two dimensions. It appears that even larger benefits, in time savings and ease of use, could come by applying this method to geometries in three dimensions.

LITERATURE CITED

1. Bruns, J. E. and Smith, C. F., "Full Navier-Stokes Calculations on the Installed F/A-18 Inlet at a High Angle-of-Attack," AIAA paper 92-3175, July 1992.

2. Iek, C., Boldman, D. R. and Ibrahim, M., "Analysis of an Advanced Ducted Propeller Subsonic Inlet," AIAA Paper 92-0274, January, 1992.

3. DeBonis, J. R., "Full Navier-Stokes Analysis of a Two-Dimensional Mixer/Ejector Nozzle for Noise Suppression," NASA TM-105715, July 1992.

4. Steinbrenner, J. P., Chawner J.R., and Fouts, C.L., "The Gridgen 3D Multiple Block Grid Generation System. Vol. II: User's Manual," WRDC-TR-90-3022 Vol II, Wright Patterson AFB, OH, July, 1990.

5. Visich, M. et al, "Advanced Interactive Grid Generation Using RAMBO-4G," AIAA Paper 91-0799, January 1991.

6. Steger, J. L., "Thoughts on the Chimera Method of Simulation of Three-Dimensional Viscous Flow," Computational Fluid Dynamics Symposium on Aeropropulsion, NASA CP-10045, Cleveland, Ohio, April, 1992, pp. 1.1-1.10.

7. Weatherill, N. P., "Mixed Structured-Unstructured Meshes for Aerodynamic Flow Simulation," Aeronautical Journal of the Royal Aeronautical Society, April, 1990, pp. 111-123.

8. Lee, K. D. "Grid Quality Control in Computational Fluid Dynamics," Control and Dynamic Systems Series, Vol. 57, Academic Press.

9. Holmes, D. G. and Connell S. D., "Solution of the 2D Navier-Stokes Equations on Unstructured Adaptive Grids," AIAA Paper 89-1932, June 1989.

10. Cooper, G. and Sirbaugh, J., "The PARC Distinction: A Practical Flow Simulator," AIAA Paper 90-2002, July 1990.

11. Amhdal, D., "Interactive Multi-Block Grid Generation," Numerical Grid Generation in Computational Fluid Dynamics, Pine Ridge Press, Swansea, Wales, UK, 1988, pp.579-588.

12. Soni, B. K., "Two- and Three-Dimensional Grid Generation for Internal Flow Applications of Computational Fluid Dynamics," AIAA Paper 85-1526, July 1985.

13. Anderson, W. K., "Grid Generation and Flow Solution Method for Euler Equations on Unstructured Grids," NASA TM-4295, April 1992.

14. Bowyer, A., "Computing Dirichlet Tessellations," Computer Journal, Vol. 24, No. 2, 1981, pp. 162-166.

15. Pulliam, T. H. and Steger, J. L., "Implicit Finite Difference Simulations of Three Dimensional Compressible Flow," AIAA Journal, Vol. 18, No. 2, February 1980, pp. 159-167.

16. Pulliam, T. H., "Euler and Thin Layer Navier-Stokes Codes: ARC2D, ARC3D," Notes for Computational Fluid Dynamics User's Workshop, The University of Tennessee Space Institute, Tullahoma Tennessee, (UTSI Publication E02-4005-023-84), March 1984, pp. 15.1-15.85.

17. Beam, R. and Warming R. F., "An Implicit Finite Difference Algorithm for Hyperbolic Systems in Conservation Law Form," Journal of Computational Physics, Vol. 22, No. 1, September 1976, pp. 87-110.

18. Thomas, P. D., "Numerical Method for Predicting Flow Characteristics and Performance of Nonaxisymmetric Nozzles-Theory," , NASA CR-3147, September 1979.

19. Baldwin, B. S. and Lomax, H., "Thin Layer Approximation and Algebraic Model for Separated Turbulent Flows," AIAA Paper 78-257, January 1978.

20. Chien, K. Y., "Prediction of Channel and Boundary Layer Flows with a Low Reynolds Number Turbulence Model," AIAA Journal, Vol. 20, January 1982, pp. 33-38.

21. Georgiadis, N. J., "An Evaluation of Turbulence Models for Propulsion Flows," M.S. Thesis, University of Akron, May 1993.

22. Mavriplis, D. "Solution of the Two-Dimensional Euler Equations on Unstructured Triangular Meshes," Ph.D. Dissertation, Princeton University, June 1987.

23. Harris, C. D., "Two-Dimensional Aerodynamic Characteristics of the NACA 0012 Airfoil in the Langley 8-Foot Transonic Pressure Tunnel," NASA TM-81927, April 1981.

REPORT DOCUMENTATION PAGE

1. AGENCY USE ONLY (*Leave blank*)	2. REPORT DATE July 1994	3. REPORT TYPE AND DATES COVERED Technical Memorandum

4. TITLE AND SUBTITLE

A Method for Flow Simulation About Complex Geometries Using Both Structured and Unstructured Grids

5. FUNDING NUMBERS

WU-537-02-23

6. AUTHOR(S)

James R. DeBonis

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

National Aeronautics and Space Administration
Lewis Research Center
Cleveland, Ohio 44135-3191

8. PERFORMING ORGANIZATION REPORT NUMBER

E-8955

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

National Aeronautics and Space Administration
Washington, D.C. 20546-0001

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

NASA TM-106633

11. SUPPLEMENTARY NOTES

Responsible person, James R. DeBonis, organization code 2740, (216) 433-6581.

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Unclassified - Unlimited
Subject Category 02

12b. DISTRIBUTION CODE

13. ABSTRACT (*Maximum 200 words*)

A computational fluid dynamics code which utilizes both structured and unstructured grids was developed. The objective of this study was to develop and demonstrate the ability of such a code to achieve solutions about complex geometries in two dimensions. An unstructured grid generator and flow solver were incorporated in to the PARC2D structured flow solver. This new unstructured grid capability allows for easier generation and manipulation of complex grids. Several examples of the grid generation capabilities are provided. The coupling of different grid topologies and the manipulation of individual grids is shown. Also, grids for realistic geometries, a NACA 0012 airfoil and a wing/nacelle installation, were created. The flow over a NACA 0012 airfoil was used as a test case for the flow solver. Eight separate cases were run. They were both the inviscid and viscous solutions for two freestream Mach numbers and airfoil angle of attacks of 0 and 3.86 degrees. The Mach numbers chosen were for a subsonic case, Mach 0.6, and a case where supersonic regions and a shock wave exists Mach 0.8. These test case conditions were selected to match experimentally obtained data for code comparison. The results show that the code accurately predicts the flow field for all cases.

14. SUBJECT TERMS	15. NUMBER OF PAGES 59
Computational fluid dynamics; Composite grids; Structured grids; Unstructured grids	16. PRICE CODE A04

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	