Computational Aircraft and Armament Stability and Control Techniques Applied to the F-16

Scott A. Morton*, Stefan Görtz, David R. McDaniel
United States Air Force Academy, Department of Aeronautics, CO

John Dean
46 SK/SKE, Eglin Air Force Base, FL

Abstract

This paper documents some of the early results of a three year project to develop a computational method for accurately determining static and dynamic stability and control derivatives of a fighter with various weapons configurations as well as the aircraft response to pilot input. In this first year of the project computational data is gathered for a rigid F-16 with no control surface movement in forced motion that approximates flight test maneuvers and wind-tunnel testing techniques. The data is then post-processed to determine the resulting static and dynamic stability derivatives. Also, data is gathered for static F-16 simulations with various control surface positions and post-processed to determine the control derivatives. Derived stability and control data is then compared to available flight test data to show validity of the method. In addition to the computational results, this first year effort will produce two critical pieces of software to aid in quick simulation of aircraft in these maneuvers: a control surface implementation toolkit and a GUI-based maneuver file generation tool. The current status of these two software elements is reviewed. The main benefits of this effort are: 1) early discovery of complex aerodynamic phenomena that are typically only present in dynamic flight maneuvers and therefore not discovered until flight test, and 2) rapid generation of an accurate aerodynamic database to support aircraft and weapon certification by reducing required flight test hours and complementing current stability and control testing.

1. Introduction

PRACTICALLY every fighter program since 1960 has had costly nonlinear aerodynamic or fluid-structure interaction issues discovered in flight test. The main reason for these "failures" is that the predictive methods used were not able to reveal the onset and nature of the problems early in the design phase. To keep the budget overshoot under control, fixes tend to be *ad hoc* and are applied without a sound basis of fundamental understanding of the physics concerned. Unfortunately, in future aircraft designs, the problems will only become more complex as thrust vectoring, active aeroelastic structures, and other related technologies are implemented for stability and control augmentation. Unmanned combat vehicles will operate in flight regimes where highly unsteady, nonlinear, and separated flow characteristics dominate since there are no man-rating requirements [1]. In order to decrease the costs incurred by extensive flight-tests and the post-design phase modifications, it would be helpful to have a tool which enables aircraft designers to analyze and evaluate the non-linear flight-dynamic behavior of the aircraft and/or associated armament, in the form of stability and control (S&C) derivatives, early in the design phase.

Three traditional methods exist to determine stability and control derivatives. The first, and most accurate method, involves flight-testing the actual aircraft. These tests are very expensive, time consuming, and require an operational aircraft, which may not exist in the early stages of the design

process. The second method is to use wind tunnel testing of scale models. This is also a time consuming and expensive process. Additionally, there are blockage, scaling, and Reynolds-number effects together with support interference issues that prevent the proper modeling of the full-scale vehicle behavior. The final method employs a combination of data sheets, linear aerodynamic theory, and empirical relations. This method has met with great success due to its simplicity, but its accuracy is limited.

A relatively new tool in this quest is Computational Fluid Dynamics (CFD). Navier-Stokes solvers have reached a level of robustness and maturity to support routine, everyday use on relatively inexpensive computer clusters. However, the generation of properly refined and converged grids is still a serious bottleneck. The multidisciplinary method of computing solutions across the envelope proposed here will allow CFD to impact the design cycle as opposed to simply diagnosing an existing problem, saving significant acquisition costs. The computation of static stability derivatives can be done with present off-the-shelf CFD tools. However, the prediction of dynamic derivatives requires a time-dependent prescribed motion capability in the flow solver and proper prescribed motions. The computation of control derivatives with high-fidelity CFD is less researched partly due to the overhead involved with the generation of grids for various control surface deflections and the lack of computing resources. Adding conventional control surfaces to existing structured or unstructured grids is not a trivial undertaking, especially considering the refinements necessary to properly model the separated flow regions spawned by large surface deflections [2]. Moreover, an efficient dynamic control surface movement capability is needed to support quick-turn static stability and control derivative computations as well as time-accurate maneuver simulations.

The present paper provides an update on the first-year effort to develop a high-fidelity simulation environment that will bring together aerodynamics, aeroelasticity and flight mechanics into a time accurate simulation tool. The benefits from such a tool to the areas of aircraft stability and control, flight simulation, and aircraft and weapon certification could potentially result in savings reaching into the billions of dollars [3]. The paper begins with a review of previous research in the field, followed by the objectives of this research. Next, the status of the tools being developed to support this effort is discussed. Finally, some preliminary results are presented.

Previous Research

The status, opportunities, and challenges of applying Computational Fluid Dynamics (CFD) methodology to current and future issues in the field of aircraft stability and control (S&C) was discussed at a NASA-sponsored symposium on Computational Methods for Stability and Control (COMSAC) [4,5]. The proceedings of the symposium highlight the different approaches to modeling S&C with CFD, and it is evident from this review paper that there is a lack of organized direction in this quest.

Researchers at NASA Ames, for example, have attempted to perform a “brute force” approach to filling a stability and control database for vehicle design [6,7]. They found that a reasonable database for static stability and control derivatives would include on the order of “30 different angles-of-attack, 20 different Mach numbers, and 5 different side-slip angles, each for a number of different geometry configurations or control surface deflections,” [7]. They demonstrated a simulation approach on a large parallel machine with good success but the simulations were limited primarily to the Euler equations and fairly coarse grids. This approach does not account for dynamic derivatives necessary for an all-encompassing vehicle design, and the method assumes the discrete set of points

computed captures all of the relevant nonlinearities in the stability derivatives when clearly a "bump in the curve" could exist between chosen test points.

Another approach proposed by NASA Ames researchers with some success was to combine many low order solutions with a few high order solutions to make this process more computationally tractable [8,9]. This improved approach is obviously more computationally efficient but relies on the ability of the algorithm/user to know when the low order solution is applicable and when it is not, and this method also suffers from the "bump in the curve between points" problem discussed above. Clearly, further development is needed before CFD can efficiently support the development of new aerodynamic concepts.

Green *et al* [10,11,12] applied automatic differentiation to a potential flow solver and a serial CFD code in Euler mode to predict S&C derivatives, including input uncertainty propagation error bounds. Their approach enables the efficient calculation of first and second order derivatives of the forces and moments with respect to a variety of code inputs. However, they demonstrated the potential flow solver's inability to predict the frequency dependent behavior of dynamic S&C quantities. The fidelity of the results computed with the serial Euler code was shown to be better but at the cost of lengthy execution times.

Other modeling approaches that have been investigated to compute S&C derivatives with CFD include reduced-order modeling [13], sensitivity analysis [14,15], response surface approximation [16].

A number of researchers have integrated flight mechanics into CFD simulations, but a thorough treatment of the problem with appropriate validation still seems to be lacking. NASA's COMSAC (Computational Stability and Control) program [4,5,17] initially extended the well-known transonic small disturbance code CAP-TSD to represent control surfaces but note that restrictions in the code prevented proper implementation. The majority of techniques applied to Navier-Stokes/Euler solvers are applicable only to structured grids. A transpiration method was used to model the Benchmark Active Controls Technology (BACT) wing with its moving controls [18]. Here, the velocity components at the boundary were altered to present the control surface with the desired deflection to the flow field. One of the more common approaches is to generate multiple grids representing key surface deflections and then use an interpolation technique to compute intermediate deflections. Trans-finite interpolation is then used to deform the volume grid due to the moving surface [19,20,21]. Another popular technique is to represent the moving surface as a separate structured block and then use an overset capability to implement the surface and volume movement [22,23]. When the flow solver allows topology changes during a solution or if time-dependent surface deflections are not a requirement, a remeshing technique may also be included with an overset procedure to model the moving surface [24,25]. Another interesting technique in the literature includes the use of a meshless solver [26]. The ability to model large surface deflections is promising, but the proper representation of the moving boundaries appears to be a complex problem.

In contrast to the structured grid approaches, very few control surface implementations into unstructured meshes have been accomplished. This is unfortunate considering the advantages of unstructured domains such as better representation of complex geometries, shorter and more automated generation timeframes, and suitability to adaptive mesh refinement techniques [27]. The Arnold Engineering and Development Center's (AEDC) FD-CADRE software (Fluid Dynamics-Computational Analysis of Dynamically Responsive Environment) [22] applies an overset technique

to unstructured grids to model store separations. The issue of modeling of components that are attached and moving relative to a parent body is discussed as being complex and the focus of future work. The SIKMA (Simulation of Complex Maneuvering Aircraft) project from DLR [28] implements an overset technique for modeling moving controls on an aeroelastic vehicle using the TAU flow solver. A hierarchal motion-node structure is implemented to keep the overset grid assembly for the moving control connected properly as the elastic wing deforms. Inviscid results for surfaces with no gap representation are presented as part of the most recent SIKMA project update. Murayama *et al* [29] model an all-moving tailplane in the simulation of a high-speed civil transport aircraft. In their technique, the tailplane mesh is continuous with the fuselage mesh, and a mapping method is implemented to "slide" the surface mesh as the tailplane rotates. A spring analogy is used to deform the volume mesh.

One of the complex issues associated with the proper modeling of moving control surfaces in CFD meshes is the representation of the gaps between the moving surface and the fixed structure. Very few of the projects mentioned previously have properly modeled these gaps [20,24]. Many of the projects have "webbed" the control surfaces to the fixed structure to avoid the massive shearing associated with the large deflections of encased controls such as conventional ailerons and rudders. As such, many of the projects mentioned previously only report results for small deflection magnitudes (±10 deg). The Chimera/overset techniques most easily overcome this issue, but the increased computational load required at each time step and the associated memory requirements can be substantial. The required interpolations on unstructured grids and the associated flow conservation issues also raises questions [2]. A few "sliding grid" approaches have been reported that offer promise for overcoming this shearing issue. Allen *et al* [30] have reported favorable results using structured blocks with sliding interfaces to model control surfaces in structured grids. Marcum *et al* have implemented a similar technique for unstructured meshes [31] applied to a missile with a rotating tail section, but the technique involves local remeshing as the surface moves.

An alternative to overset techniques for representing moving components is the deformation of the mesh. A number of mesh deformation schemes have been successfully implemented over the years to support CFD studies of aeroelastic vehicles. Samareh includes a good review of many of these techniques in a fairly recent paper [32]. Specific to structured grids, trans-finite interpolation is a common way to relocate (regenerate) mesh nodes [33,34]. Applicable to both structured and unstructured meshes, a series of linear and torsional springs may be used to model the "stiffness" of edges and angles, respectively, in the grid [35,36,37]. After the surface deformation is implemented, the volume nodes are relocated based on the solution to the coupled system. The volume mesh may also be modeled as an elastic solid where updated node locations are determined by solving the elasticity equations [38]. Kholodar *et al* implement a hybrid technique for unstructured viscous meshes where the prismatic boundary layer region of the mesh is relocated according to the changing normal basis vectors of the moving surface and the volume mesh is then updated according to the spring analogies mentioned previously [39]. A couple of efficient algebraic techniques are also discussed in recent literature. Samareh implements an unstructured mesh deformation technique based on quaternion algebra [32], and Liu *et al* implement a point location technique based on a Delaunay tetrahedralization of the outer boundary and deformable surface [40].

Research Objectives

The large body of previous work [41,42,43,44,45,46,47] performed by researchers at the US Air Force Academy using the unstructured mesh solver *Cobalt* [48] coupled with a Detached-Eddy

Simulation (DES) turbulence treatment, adaptive mesh refinement, six degree of freedom (6 DOF) motion and deforming grids for aero-elasticity has led to a high-fidelity capability for computing S&C derivatives in a more efficient and accurate way than existing approaches. The proposed approach is to combine the demonstrated capabilities to perform full aircraft simulations at flight Reynolds numbers that include aircraft motion with well-developed flight test techniques for gathering the necessary data across the flight envelope through the use of aircraft maneuvers and post processing of the aircraft response. Although the prescribed aircraft motions and virtual 6 DOF maneuvers will be inspired by flight test techniques, they will take advantage of the tighter control possible with CFD to increase the efficiency of the maneuvers. Also, more complex maneuvers can be envisioned with CFD than is possible with flight test (e.g. combining pitch oscillation and plunge to remove $\dot{\alpha}$). This approach has the advantage of using flight test maneuvers such as a wind-up turn to generate the static and dynamic loads as well as the static and dynamic derivatives including important inertial and aeroelastic effects, which can give rise to non-linear results not easily predicted with the traditional approaches. Further, the approach may be more computationally tractable since a single calculation can expose the important dynamic effects that would normally need to be examined parametrically.

The existing capabilities will be extended to include flight control surface movements by incorporating an efficient grid deformation strategy. Most structured grid techniques for modeling control surface movement are not applicable to unstructured grids. Additionally, the unstructured Chimera/overset techniques currently available suffer from large computational expense and/or the inability to include relative movement between a fixed structure and a moving component. Also, existing techniques have been focused more towards high-speed aeroelastic solutions and therefore involve small surface deflections (±8 deg). In the current research, larger deflections of up to ±60 deg are desired. Another objective of this research is to automate the control surface creation and deformation setup process as much as possible. Instead of having to regenerate a full-aircraft volume mesh after manually inserting a control surface into the geometry, the aim here is to start with an existing converged, good-quality grid and then automatically embed the control surface into the volume mesh together with the required mesh deformation features. Since overset techniques have continually defied automation attempts [49], this research is focusing on a quick and efficient algebraic grid deformation technique.

The proposed research will culminate in a “virtual flight test” method that may be used to directly examine the classical dynamic aircraft responses, which define flying qualities and have certification requirements. The ultimate goal is to develop a methodology for efficiently and accurately screening for nonlinear aerodynamic phenomena such as spin, tumble, lateral instabilities, limit-cycle oscillations, and tail buffet of full aircraft using a combination of static-steady and unsteady single points, rigid body motion unsteady solutions, and 6 DOF simulations that include aeroelastic effects.

2. Simulation Environment

The main objectives of rapidly computing stability and control derivatives of various aircraft/armament configurations and simulating maneuvering aircraft requires the association of four main software modules: 1) a Navier-Stokes unstructured flow solver, 2) a tool to generate the necessary definition files for time-dependent motion and to integrate the flight mechanics equations of motion into the simulation loop, 3) a tool to embed the applicable control surfaces into the CFD mesh and then deform the grid according to commanded surface deflections, and 4) a control system

model for integrating feedback controls and autopilot functionality into the simulations. These components must be connected via a software interface module as shown in Figure 1.

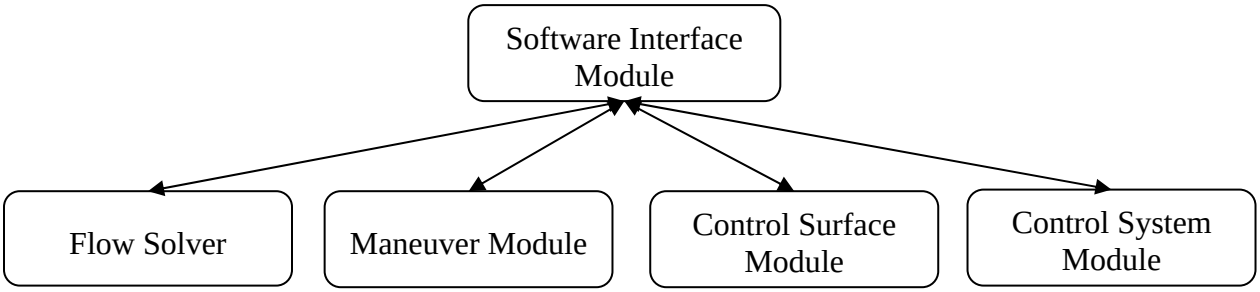

Figure 1: Stability and Control Simulation Environment

This software module will serve to marshal the requests among the three main software components. With this simulation structure in place, a number of computational scenarios are possible. Figure 2 shows the flow of information through the modules when computing static stability and control derivatives.

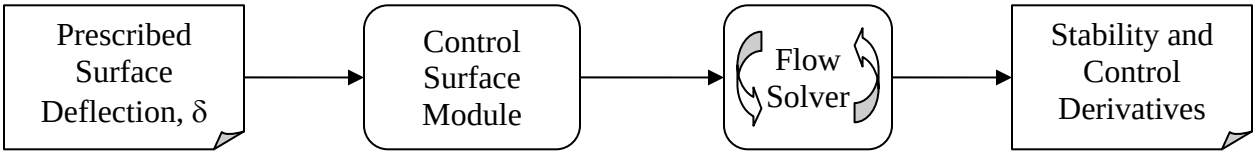

Figure 2: Computational process to determine static aerodynamic derivatives

Figure 3 shows the flow of information when computing the time-accurate results from a prescribed (forced) motion of the vehicle or when determining dynamic derivative values.

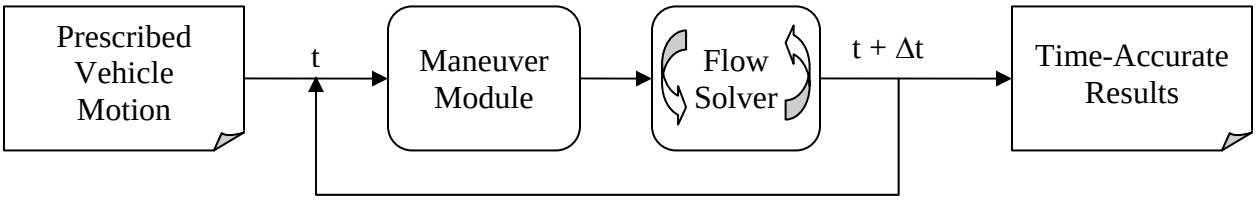

Figure 3: Computational process for time-accurate results of prescribed vehicle motion

Finally, Figure 4 shows the computational process when integrating the flight mechanics equations of motion to virtually "fly" the aircraft or armament under study based on a given surface deflection history and appropriate mass properties. Alternatively, it may be necessary for a pilot (or autopilot) command history to drive the simulation. In this case, an additional control system module is necessary for integration with the other simulation components.

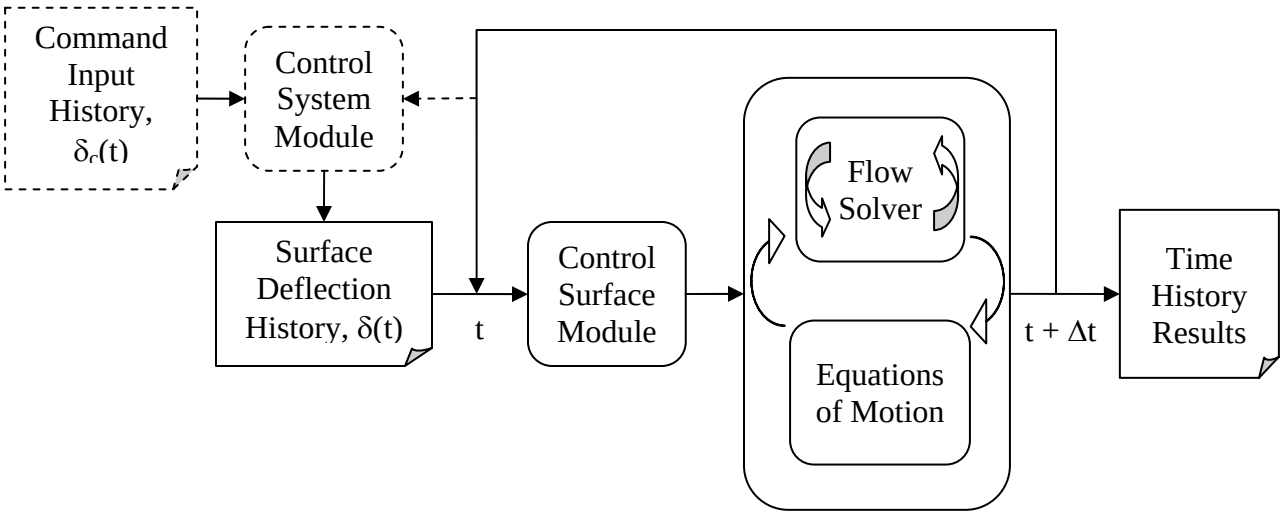

Figure 4: Computational process for "virtual" flight simulation

Flow Solver

Computations are performed using the commercial flow solver *Cobalt*. *Cobalt* is a cell-centered, finite volume CFD code. It solves the unsteady, three-dimensional, compressible Reynolds Averaged Navier-Stokes (RANS) equations on hybrid unstructured grids. Its foundation is based on Godunov's first-order accurate, exact Riemann solver. Second-order spatial accuracy is obtained through a Least Squares Reconstruction. A Newton sub-iteration method is used in the solution of the system of equations to improve time accuracy of the point-implicit method. Strang *et al* [48] validated the numerical method on a number of problems, including the Spalart-Allmaras model, which forms the core for the Detached Eddy Simulation (DES) model available in *Cobalt*. Tomaro *et al* [50] converted the code from explicit to implicit, enabling CFL numbers as high as 10^6. Grismer *et al* [51] parallelized the code, yielding linear speed-up on as many as 2,800 processors. The parallel METIS (PARMETIS) domain decomposition library of Karypis *et al* [52] is also incorporated into *Cobalt*. New capabilities include rigid-body and 6 DOF motion, equilibrium air physics and overset grids. A coupled aeroelastic simulation capability is also being developed.

Maneuver Module

The maneuver module builds onto two components, the rigid-body motion module implemented in *Cobalt* and a newly developed interactive GUI, which takes inputs from the user of what type of maneuver, and a few key parameters, and converts those inputs into the fairly complicated rigid grid movement description called a motion file. This file then forms part of the required input deck for *Cobalt*.

Cobalt uses an Arbitrary Lagrangian Eulerian (ALE) formulation to perform the grid movement, where the grid is neither stationary nor follows the fluid motion but is reoriented without being deformed. The coordinate values change, but the relative positions between the grid points are unchanged. As a result, terms such as cell volume and face area remain constant and equal to the values in the original grid. The motion can include both translation and rotation.

There are three motion type categories available in *Cobalt* – specified, external-controlled and free motion. The motion type and corresponding parameters are specified in a motion file in ASCII format. Several motion types can be combined in a single motion file to simulate more complex motions. However, the separate motions cannot overlap in time. Yet, complex motions can be defined by specifying arbitrary rotations and displacements of the grid. Users supply the basis-vectors of a reference frame, $\hat{X}$, attached to the grid and the origin of $\hat{X}$ at discrete points in time. The origin of $\hat{X}$ is also the point about which the grid rotates. The user-supplied data is cubic-splined between discrete time values. The motion is governed by:

$$\vec{x}^{n+1} - \vec{x}_c = TM\left\{\vec{x}^n - \vec{x}_c\right\} + \Delta\vec{x}_c, \tag{1}$$

where $\vec{x}^{n+1}$ is the new position vector at time-step *n+1*, $\vec{x}^n$ is the previous position vector at time-step *n*, $\vec{x}_c$ is the location of the center of rotation, *TM* is the transformation matrix necessary to rotate the grid from one time-step to the next and $\Delta\vec{x}_c$ is the displacement of the center of rotation. *Cobalt* calculates *TM* from the user's input.

The generation of the motion file is tedious and error-prone for complex maneuvers. Analytical functions, e.g. for Gaussian pulses or higher harmonics, cannot be specified directly. The idea behind the interactive GUI is to enable the user either to specify sophisticated maneuvers analytically, to choose from a set of predefined maneuvers, or to load actual flights test data. Either way, the GUI translates the maneuver into corresponding rotations and displacements of the grid following Eq. (**1**) and outputs a motion file in *Cobalt* format.

The GUI is based on the *parameter* MATLAB class by Bleeck [53].It can be made up of several objects of this class. Each object is created in a single line of code by calling upon the *parameter* class with a parameter type. From here user interaction with the object is possible. Available parameter types include: floating point variables (with or without a unit), integer values, sliders, string values, checkboxes, radio buttons, pop-up menus, buttons for selecting file and directory names, and push buttons that call a call back routine. The basic behavior and placement of the different controls is consistent throughout the application due to the use of the *parameter* class, making the GUI easy to use. The range of floats can be restricted by setting the min and max values. These are checked automatically by the GUI. A warning box pops up if a value is entered that is not valid.

The GUI is designed such that there are only a few features at the top level. First, the user has to choose a motion category from a radio button menu. When the user hits the 'Next' button, the top-level window will be closed and a new window will appear, giving the user a choice of motions/maneuvers from only that category. Other motion types and maneuvers are tucked away behind drop-down panels and radio button menus, accessible when needed but not staring the user in the face. In the next window, the user can define the selected motion by entering a few key parameters. For each parameter, the GUI provides either a list of data values to select from or a default value. The editable fields are identified with a white box around them so that it is intuitively obvious which fields can be changed. Parameters having a unit feature a drop-down menu next to the value field, giving the user a choice of different equivalent units (e.g. s, ms, μs for parameters associated with time). This is illustrated in Figure 5, which shows the GUI window used for specifying the parameters of a plunge motion. Similar parameters are grouped together in boxes,

making the use of the GUI a more intuitive step-by-step editor. Each GUI window has a 'help' push button, which, when pushed, opens a secondary windows with detailed information about the choices in the first window, such as motion types or motion parameters. The user can always return to the previous window/menu by hitting the 'Back' button available in each window. A call back push button called 'Save and exit' is used to call a MATLAB function that retrieves and evaluates the user input, converts it into a motion file (units are automatically converted into the *Cobalt* unit system) and exits the GUI. Alternatively, the user can append another motion or maneuver description to the motion file by hitting the 'Save and continue' button instead. The plotting functions available in MATLAB are used to display the corresponding maneuver in a figure. This is illustrated in Figure 5, which shows a sample plot of a plunge maneuver created with the GUI.

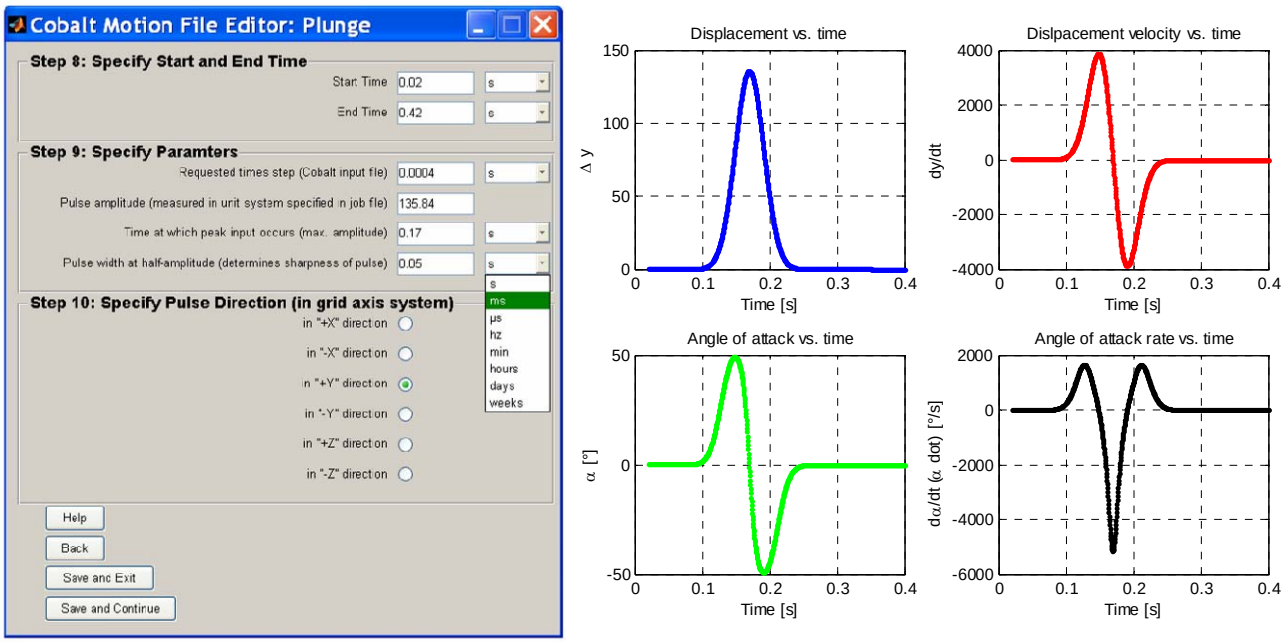

Figure 5: Interactive GUI to generate *Cobalt* motion miles (left) and preview of plunge motion generated with GUI (right)

The GUI has been compiled into a "stand-alone" application using the MATLAB Compiler. The compiled GUI comprises an executable and an associated Component Technology File (CTF) archive. The archive contains all the MATLAB based executable content (M-files, MEX-files, etc.) associated with the GUI. To run the "stand-alone" version of the GUI, the user has to install the MATLAB Component Runtime (MCR), which is a stand-alone set of shared libraries that enable the execution of M-files, the executable and the CTF archive. A MATLAB license is not required.

Control Surface Module

The control surface module currently under development consists of two parts. The first part is a stand-alone tool that automates the process of embedding a moving control surface definition into an existing grid. The second part is a module that handles the time-dependent deformation of the grid due to control surface movements.

Three views of the F-16C wing with its trailing edge flaperon included are shown in Figure 6. In order to realistically model large control deflections, gaps between the moving surface and the fixed

structured must be adequately modeled. This obviously means that additional meshed surface areas must be incorporated into the original grid. Since one of the goals here is to avoid the complete regeneration of the surface and volume meshes from the original geometry information (which is often times not readily available), the lofting of hinge lines, control surface edges, etc. through the existing surface mesh is required. Careful localized remeshing is also necessary to reconnect the existing volume grid with the new surface mesh topology while maintaining a consistent viscous layer structure.

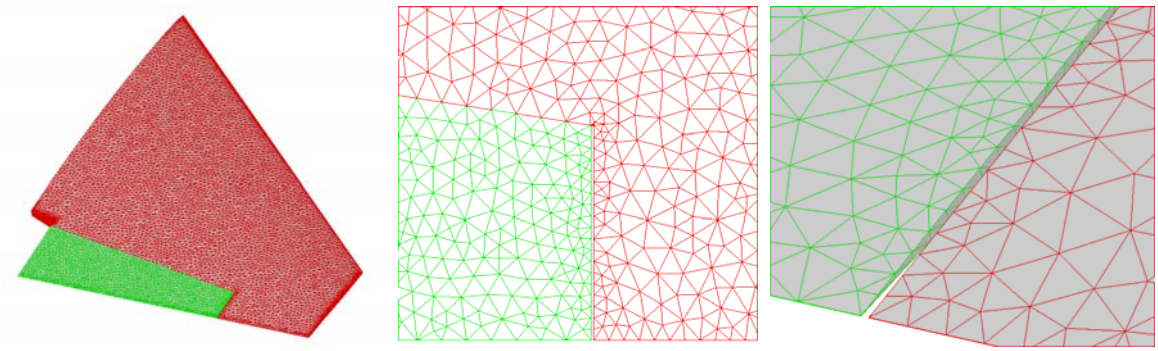

Figure 6: F-16C Trailing edge flaperon surface

Various grid utilities to include arbitrary surface spline tools, advancing front remeshing routines, and grid subsetting capabilities have been developed to facilitate this automated process of inserting control surfaces into existing grids.

Focused efforts are being made to keep this grid manipulation process computational tractable even though full aircraft/armament grids are pushing into the tens of millions of cells. This tool provides the ability to subset out portions of the full vehicle grid so that the work may be accomplished on a local workstation. Figure 7 shows an example of this subsetting process for the F-16C main wing. The smaller, more manageable subset grids may be manipulated locally on a desktop workstation and then later re-merged with the main grid located on the high-performance computing file servers.

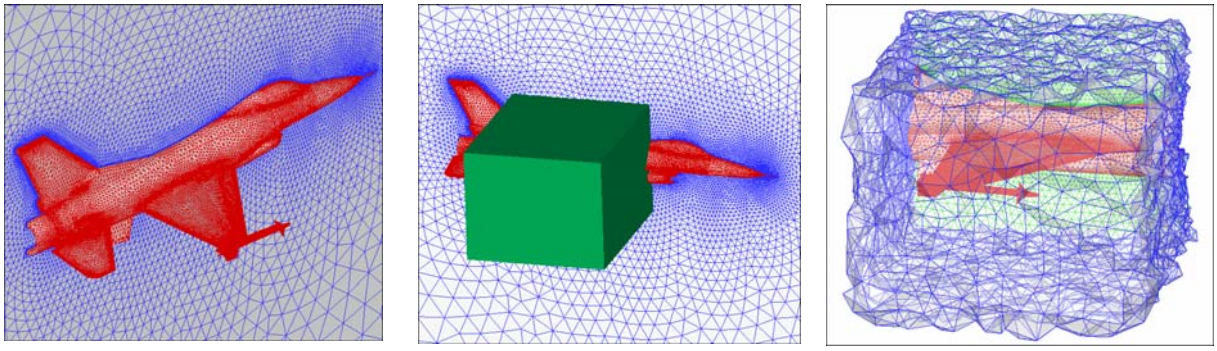

Figure 7: Grid subsetting of the F-16C main wing

The second part of the control surface module is the manipulation of the grid to accommodate the surface and volume mesh deformation due to control surface movement. Regarding the surface mesh movement, the assumption here is that the hinge line of the control surface is a single line where the continuous surface is "creased" to accommodate deflection of the surface. In other words, no gaps along the hinge line are present as with high-lift surfaces, for example. This modeling assumption is consistent with the physical structure of all modern high-performance aircraft.

However, because of this assumption, the surface mesh must deform slightly as the control surface rotates about the real hinge line as opposed to the creases in the surface mesh. Currently, this circular rotation about the hinge line is not modeled, and the surface nodes are deflected only in the vertical direction based on simple trigonometry using the deflection angle and the location of the node with respect to the hinge line plane. Obviously, more errors are introduced with larger control surface deflections under this methodology. Proper stretching of the surface nodes to keep the actual control surface geometry intact throughout the deflection will be properly handled in the near future. Figure 8 shows four deflection angles for the F-16C flaperon using this methodology.

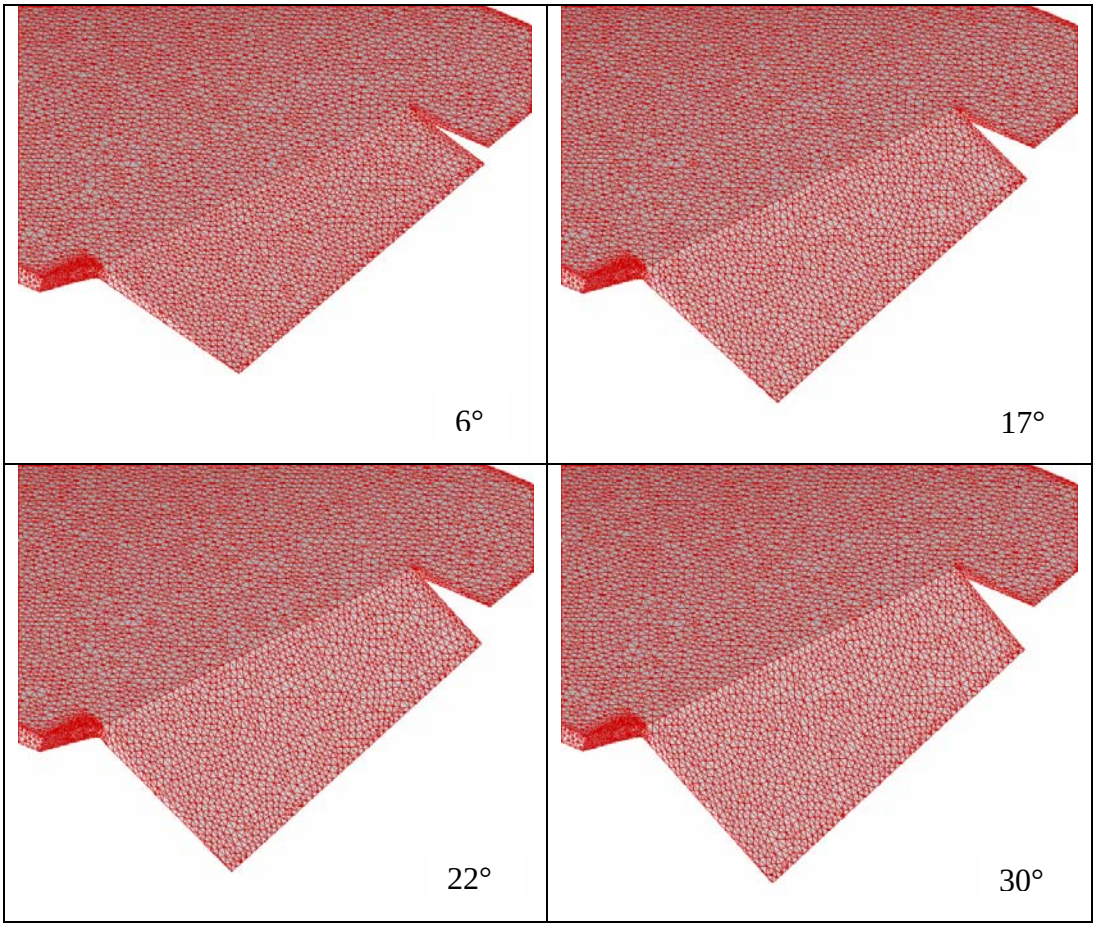

Figure 8: Surface mesh deflection of F-16C flaperon (6, 17, 22, and 30 degrees)

Once the movement (deformation) of the surface mesh has been accomplished, the volume mesh must be properly deformed to accommodate the moving boundaries. For encased control surfaces where gaps exist (e.g. the outboard edge of the F-16C flaperon), a massive amount of cell shearing occurs, and deformation of the grid in this region is virtually impossible. Therefore, a sliding plane methodology is being investigated for use in this area. Currently, deformation due only to movement of a full-span surface type is considered in which case cell shearing in the gapped regions does not exist. Figure 9 shows a fictitious wing modeled after the F-16C with a full-span flap at 0 and 30 degrees deflection using the surface deformation technique discussed previously. The deformation of the volume mesh is accomplished based on the algebraic technique developed by Liu *et al* [40].

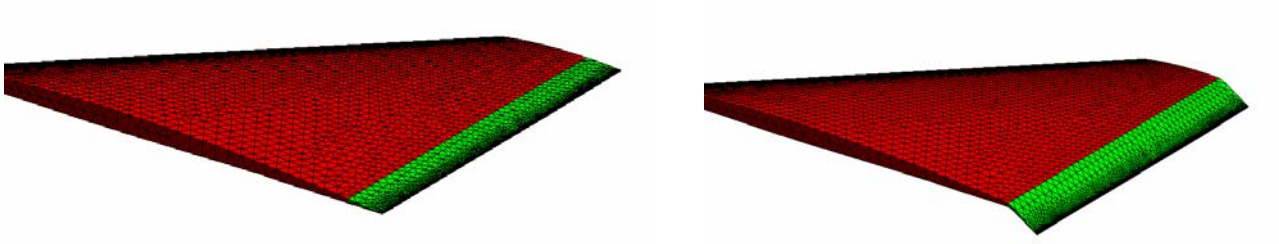

Figure 9: F-16–like wing with full-span trailing edge flap deflected at 0 and 30 degrees

When applying this deformation technique to the control surface problem, it is not necessary to deform the entire grid volume when the control surface deforms. It is acceptable to identify a smaller subset of the grid as the deformable region. Figure 10 shows the deformable region identified for the movement of the trailing edge flap on the fictitious F-16 wing shown in Figure 9. The definition of this deformation subset volume is arbitrary, and the quality of the grid cells throughout the deformation due to the full range of motion of the control surface is highly dependent upon the size and shape of this volume. Fortunately, the technique proposed here represents a pre-processing step, and the quality of the grid as the control surface deflects through its full range of motion may be monitored and appropriate actions taken prior to beginning the integrated computation.

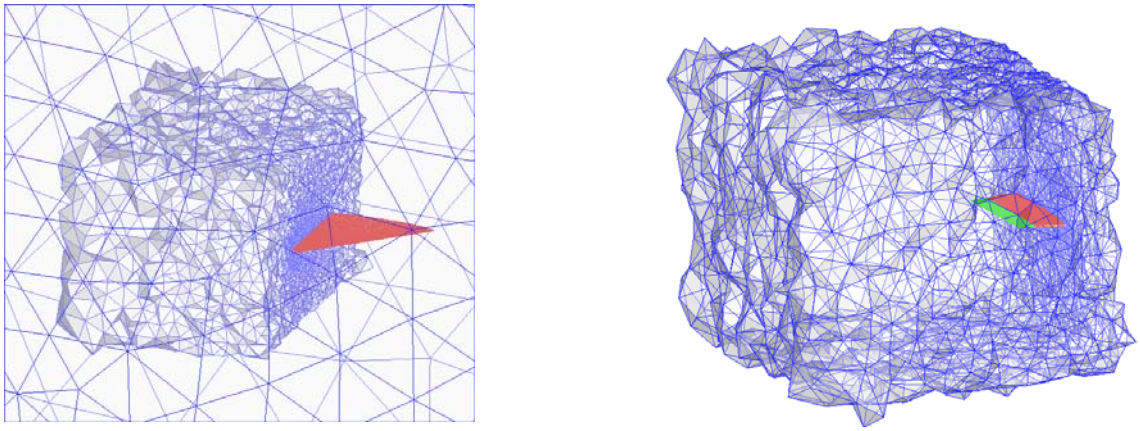

Figure 10: External and internal views of grid deformation region for fictitious F-16 wing

As discussed in Liu *et al* [40], the deformation technique used here is based on the generation of a Delaunay tetrahedralization of the moving surface nodes within a fixed outer boundary [54,55]. Figure 11 shows three views of increasing resolution of the tetrahedralization applied to the F-16 wing with the full-span trailing edge flap. The first image in Figure 11 shows the deformation region as well as the eight external points used to initialize the Delaunay graph. Although only the corner points are needed to define the deformation region in the tetrahedralization, the current research has shown that the distribution of points around the boundaries of the deformable region is important for large surface deflections to avoid negative cell volumes in the deformed Delaunay graph. Some of the individual cells in the tetrahedralization are more apparent in the second and third images in Figure 11.

Once the Delaunay graph is generated, each of the nodes in the volume mesh is located within the cells of this tetrahedralization. Then, since lines connecting the volume node to each of the nodes of the applicable cell in the Delaunay graph divides the cell into four smaller tetrahedral volumes, a set of four volume ratios are computed and stored for each node.

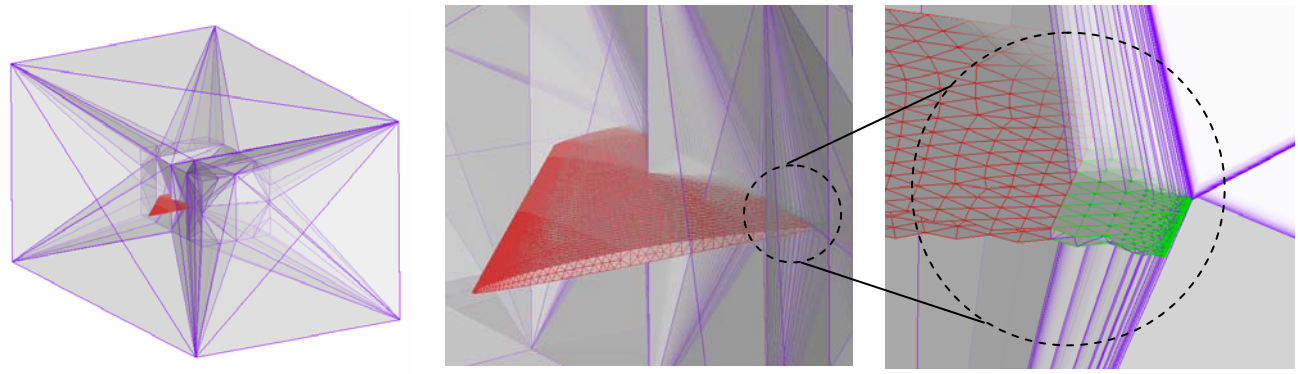

Figure 11: Delaunay tetrahedralization of trailing edge flap of fictitious F-16 wing

As the surface mesh is deformed due to the prescribed control surface movement, the tetrahedralization is naturally deformed. After the deformation, the volume ratios stored for each volume node may be used to re-locate the node within its respective cell in the updated tetrahedralization. This algebraic technique is nice in that it has very few input requirements (i.e. easy to automate) and does not degrade the cell quality through multiple cycles of control surface movement. Figure 12 displays a sequence of images showing the grid deformation using this technique for the volume cells on a cutting plane at the approximate mid-span point of the F-16 wing with the trailing edge flap. Although quantitative studies of the cell quality throughout the surface deflection cycle are yet to be accomplished, the viscous layer remains intact and no unacceptable skewing of volume cells is apparent.

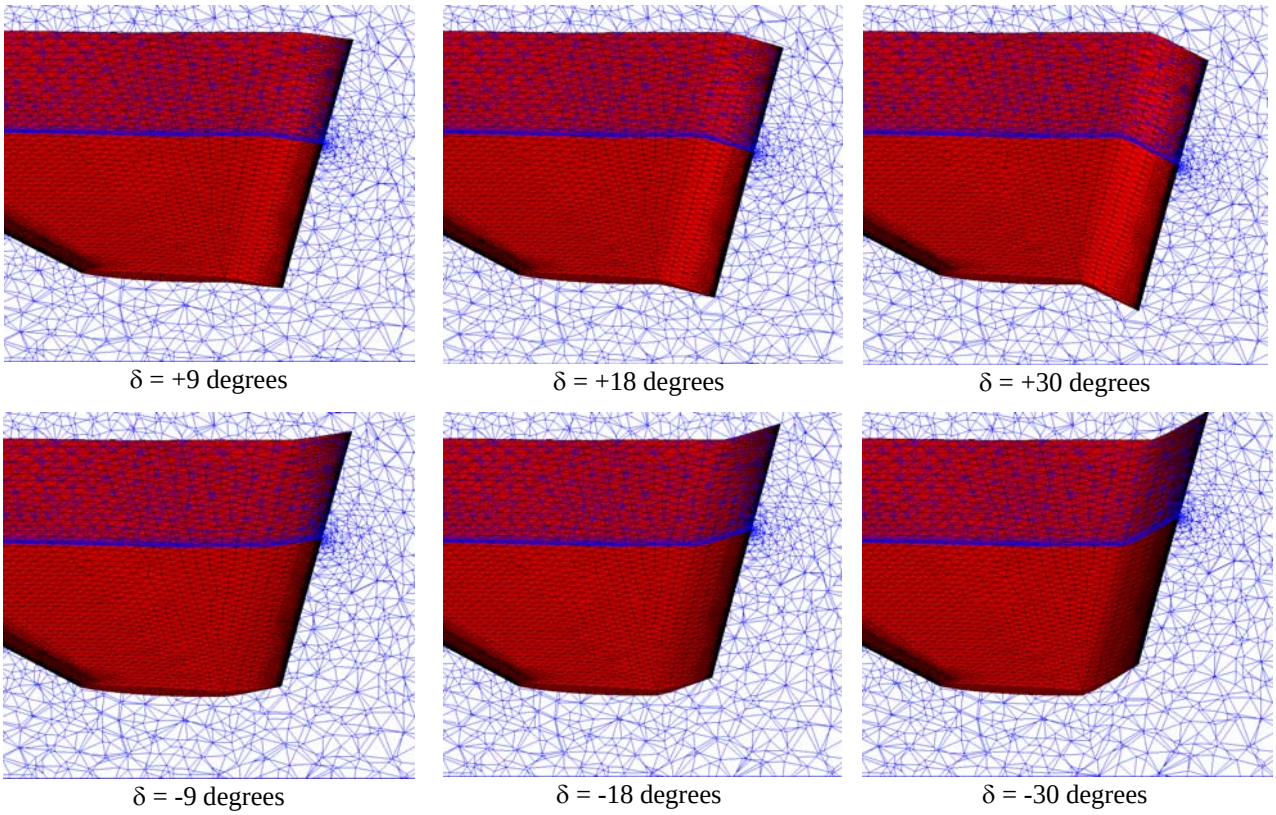

Figure 12: Volume cell deformation due to control surface deformation of trailing edge flap

The deformation technique discussed above has been implemented in a framework designed to accommodate both offline and online deformation of the volume grid. The information obtained from the preprocessing steps of generating the tetrahedralization of the deformation region and identifying and locating the volume nodes within the resulting graph is stored in data structures within binary files suitable for use in separate computational processes. Additionally, because of the architecture of the technique, multiple control surfaces (and the associated deformation regions) may be included in the same grid and manipulated independent of one another to allow for complex flight maneuvers.

Control System Module

For the purposes of the current research, the F-16's flight control system is being programmed in the MATLAB/Simulink environment for up-and-away flight conditions. Simulink models for both analog Block 25 and digital Block 40 are being developed. Current work is focused on incorporating the control laws for the high angle of attack flight regime. Validation of the control law module is being performed using data acquired during instrumented flight tests conducted at Eglin AFB and Edwards AFB. To date, subsonic and supersonic maneuvers have been modeled and validated from an F-16 flight test with tip AIM-9 missiles only, including wings level sideslips, wind-up turns and loaded rolls. F-16 configurations with known S&C and handling qualities concerns are being investigated as well, including air-to-air configurations and a variety of symmetric and asymmetric air-to-ground configurations. This module will be integrated into the simulation environment in the last year of the project.

3. Results

To date, a full-scale F-16 undergoing the following prescribed motions has been simulated:

- continuous α sweep
- sinusoidal pitching
- coning motion
- oscillatory coning
- configuration plunge pulse

These motions represent typical wind-tunnel techniques for stability and control testing. They were defined using the interactive GUI. In all cases the flow conditions were a Mach number of 0.25, a Reynolds number of 14,789,444, a static pressure of 1562.4 lb/ft^2 and a static temperature of 519° R.

Numerical Grid and Boundary Conditions

The grid used here is for a half-span, full-scale model of the F-16. The model includes the forebody bump, diverter, and ventral fin. The engine duct is modeled and meshed up to the engine face, compare Figure 13. The wing-tip missile and corresponding attachment hardware are not modeled, however, nor is the nose boom. The 3D hybrid grid was generated using the grid generation package GRIDTOOL [56] and VGRIDNS [57], as well as the commercial grid management utility BLACKSMITH [58]. The surface grid comprises 167,382 elements and is shown in Figure 13. Off the surface, there are eight prismatic layers. The height of the first prismatic layer corresponds to an

average wall y^+ value of less than four. In total, there are 790,109 nodes in the volume grid, corresponding to 3,171,892 cells. Cells are concentrated in the strake vortex.
The boundary conditions are symmetry, adiabatic solid wall for the surface of the aircraft and the engine duct, and modified Riemann invariants for the far-field boundaries. A source boundary condition based on Riemann invariants is used to create an inflow condition at the engine exhaust. A sink boundary condition is used at the engine face to model the corrected engine mass flow.

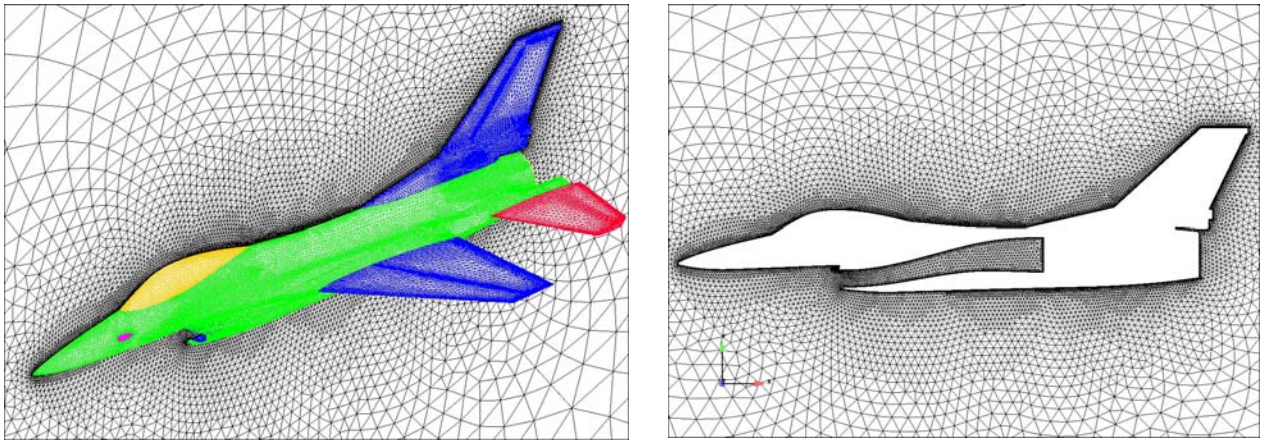

Figure 13: Unstructured numerical surface grid for the half-span full-scale model of the F-16 (left) and symmetry plane of the hybrid volume grid showing the meshed inlet duct (right)

Numerical parameters

The unsteady maneuvers were simulated using the DES, Spalart-Allmaras one-equation turbulence model with rotation correction to predict the effects of fine scale turbulence. Fully turbulent flow was assumed. The outer (physical) time step was set to Δt=0.0004s, corresponding to a non-dimensional time step of Δt^*=0.01. The number of Newton sub-iterations was set to 5. The temporal damping coefficients for advection and diffusion were set to 0.05 and 0.0, respectively. The unsteady numerical simulations were initialized by steady-state solutions computed with the Spalart-Allmaras model with rotation correction.

The computations were run on 64 processors on 'Iceberg', an 800-processor IBM Power4 system operated by the Arctic Region Supercomputing Center (ARSC). Iceberg is comprised of a combination of 92 p655+ servers, each with 8 processors and 16 GB of shared memory, 2 p690+ servers each with 32 processors and 256 GB shared memory, and 4 p655 I/O servers. The entire system has 25 TB of disk and a theoretical peak performance of five TFlops.

Sinusoidal Pitching Motion and Continuous α Sweep

A conventional wind-tunnel technique used for many years to estimate dynamic damping derivatives such as C_{m_q} is to perform 1 DOF, planar, forced-oscillation test. For this method the model is placed at various angles of attack in a wind tunnel and allowed to undergo forced sinusoidal oscillations at different frequencies and usually relatively small amplitudes. Oscillations are usually done about the pitch, roll and yaw axes. This technique produces combined, or lumped, derivatives. This occurs because the angle-of-attack rate and pitch rate are kinematically constrained to be equal.

Inspired by this experimental technique, a forced pitch oscillation was simulated numerically. The initial angle of attack of 30° was held for 0.2 s to give the flow some time to develop. Thereafter, it was varied according to the following relation:

$$\alpha(t) = 30° + 15° \times \sin(2\pi \times 1.0Hz \times t) \tag{2}$$

A total of 5,500 time steps, corresponding to two complete cycles or 2.2 s of simulation time, were computed. The simulation took about 1,860 CPU hours on 64 IBM Power4 CPUs.

Figure 14 is a sequence of images visualizing the flow computed for this sinusoidal pitching motion with imposed amplitude and angular frequency. Each image depicts an instantaneous vorticity iso-surface colored by velocity magnitude. The angle of attack is shown as a function of time in the lower left corner of each image. Note that the flow field is seen to be unsteady even at the beginning of the simulation where the angle of attack is static. This is due to the strake vortex experiencing vortex breakdown, and the massive flow separation over the main wing. At dynamic angles of attack, the flow field undergoes drastic changes, including the appearance and disappearance of a forebody vortex, the burst and reformation of the strake vortex, and the formation of a burst main-wing vortex.

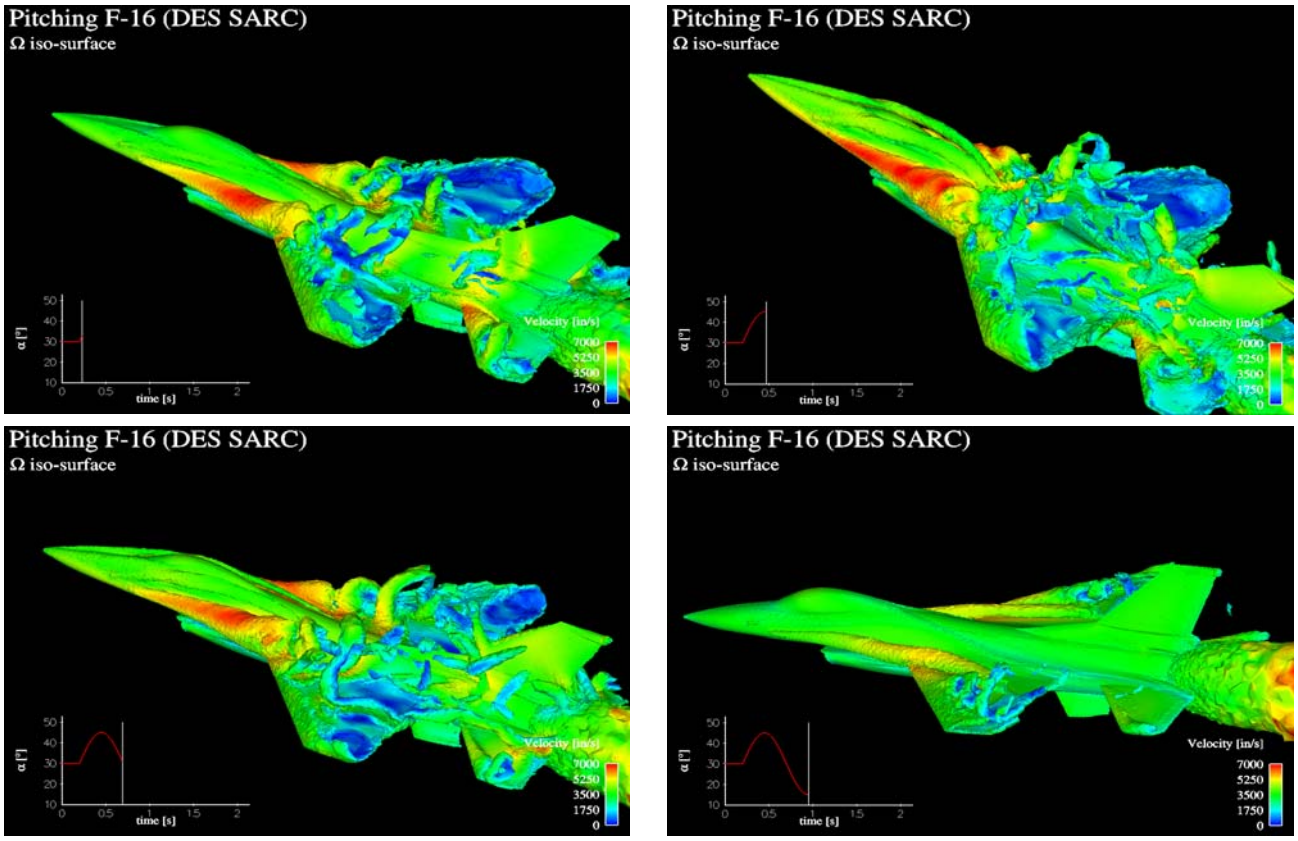

Figure 14: DES of F-16 in sinusoidal pitching motion; instantaneous vorticity iso-surface colored by magnitude of velocity, $\alpha(t) = 30° + 15° \times \sin(2\pi \times 1.0Hz \times t)$

These nonlinear phenomena give rise to nonlinear behavior of the aerodynamic forces and moments. Figure 15 shows the lift coefficient C_L as a function of the angle of attack. The green curve corresponds to the first, the black curve to the second pitch cycle. For this case, the dynamic lift

curve features a hysteresis loop that occurs because the flow at increasing angle of attack features different characteristics of that at decreasing angle of attack. The "jump" in the lift coefficient at the beginning of the first cycle is due to the infinite acceleration that occurs when the sinusoidal motion starts at t = 0.2 s. The infinite acceleration is due to a discontinuity in the second derivative of the angle of attack. As a result, the lift coefficient increases from its static value to the dynamic value that corresponds to the angle of attack rate give by the sine wave. The associated transient is seen to have disappeared in the second cycle.

Also shown in Figure 15 are the results of a continuous angle-of-attack sweep. The angle of attack was increased from 15° to 45° at a constant angular velocity of 10°/s using *Cobalt*'s rigid grid motion capability. A total of 8,000 time steps, corresponding to 3.2 of simulation time, were computed. Note that the initial angle of attack of 15° was held for 500 iterations to give the flow some time to develop. The computed lift coefficient data was smoothed in MATLAB using a 500-point moving average. The dark-blue and red lines in Figure 15 correspond to the raw and post-processed data, respectively. The fluctuations in the unprocessed data are due to unsteady nature of the flow field at the relatively high angles of attack simulated here. Note that the unsteadiness is inherent to the flow and is not caused by the grid motion, as in the case of the dynamic pitching motion. The computed results are compared to static wind-tunnel data reported in Nguyen *et al* [59]. The nine experimental data points are shown as filled circles in Figure 15.

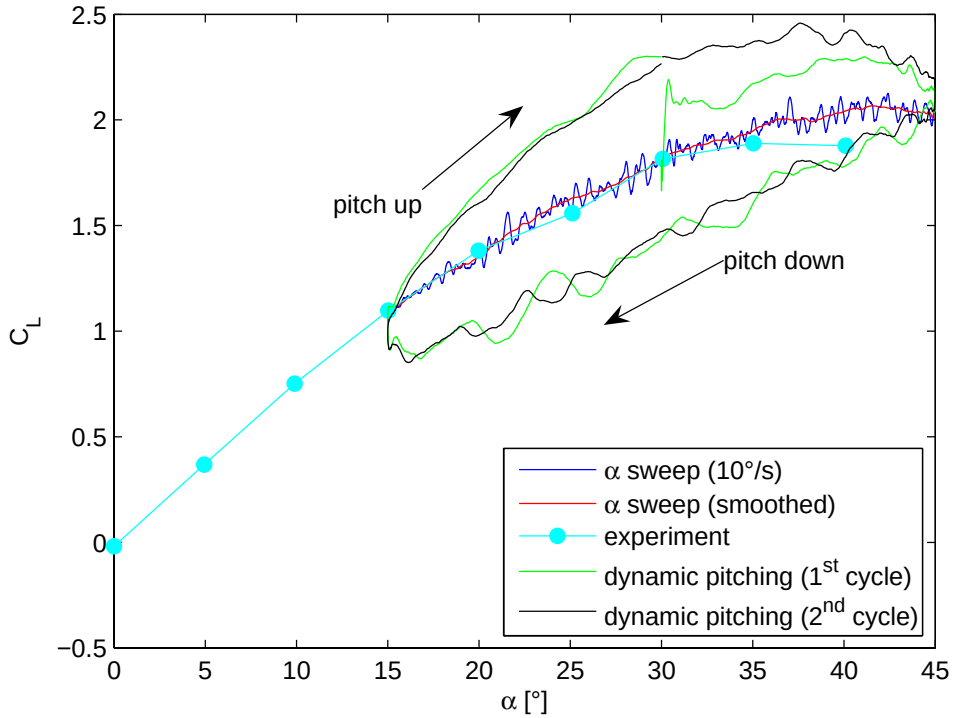

Figure 15: Lift coefficient vs. angle of attack for sinusoidal pitching motion, $\alpha(t) = 30° + 15° \times \sin(2\pi \times 1.0Hz \times t)$ **and continuous α sweep, compared to static experimental data [59].**

The discrete experimental data points fall on top of the high-density numerical C_L-alpha curve between 15° and 35° angle of attack. The difference between the computed and experimental data at α = 40° can be attributed to the fact that the Reynolds and Mach numbers do not match; the numerical simulation was performed for a full-scale model, whereas the experimental data is for a sub-scale wind-tunnel model. Additionally, the wind-tunnel model featured wing-tip missiles,

whereas the numerical model did not. Finally, the corrected engine mass flow was modeled numerically, but not in the wind-tunnel experiment. Nonetheless, the good agreement with the experimental data suggests that the angular rate used here was low enough not to give rise to a dynamic effect, resulting in a quasi-steady simulation. It can be concluded that the time-accurate α sweep offers a faster and more efficient approach to predicting the entire C_L-alpha curve than the conventional point-by-point steady-state approach. It can provide high-density, continuous data which may not be practical to obtain by steady-state CFD.

Conventional Coning Motion

A rolling motion about the velocity vector is often referred to as "coning motion." In this motion, the axis of rotation is coincident with the wind direction. The angle of attack is constant but non-zero. Coning motion is used in rotary balance testing, a common experimental method used to provide dynamic data. This motion also occurs when an aircraft is at a constant angle-of-attack, and commands a roll around the velocity vector. It is common for fighter aircraft at higher angles of attack when they are performing tracking maneuvers. This is exemplified by the so-called Herbst maneuver where coning motion is used to point the nose of the aircraft in the right direction after pitching up beyond stall [60]. At angles of attack approaching 90°, the coning motion resembles a flat spin and is also of great interest for general aviation vehicles.

Before simulating this motion with *Cobalt* the grid was mirrored at the plane of symmetry and rotated to an initial pitch angle of 30°. The axis of rotation was aligned with the wind direction, leaving the aircraft at a constant angle-of-attack. The prescribed roll rate around the velocity vector was 180°/s. A total of 4,200 time steps, corresponding to 1.68 s of simulation time, were computed. The simulation took about 2,978 CPU hours on 64 IBM Power4 CPUs. Figure 16 shows snapshots of the coning aircraft model and instantaneous streamlines. Note that time increases from left to right, top to bottom. The time interval between images is 0.33 s. No quantitative data has been extracted from this simulation as its main purpose was to demonstrate the feasibility of reproducing advanced S&C testing techniques with CFD.

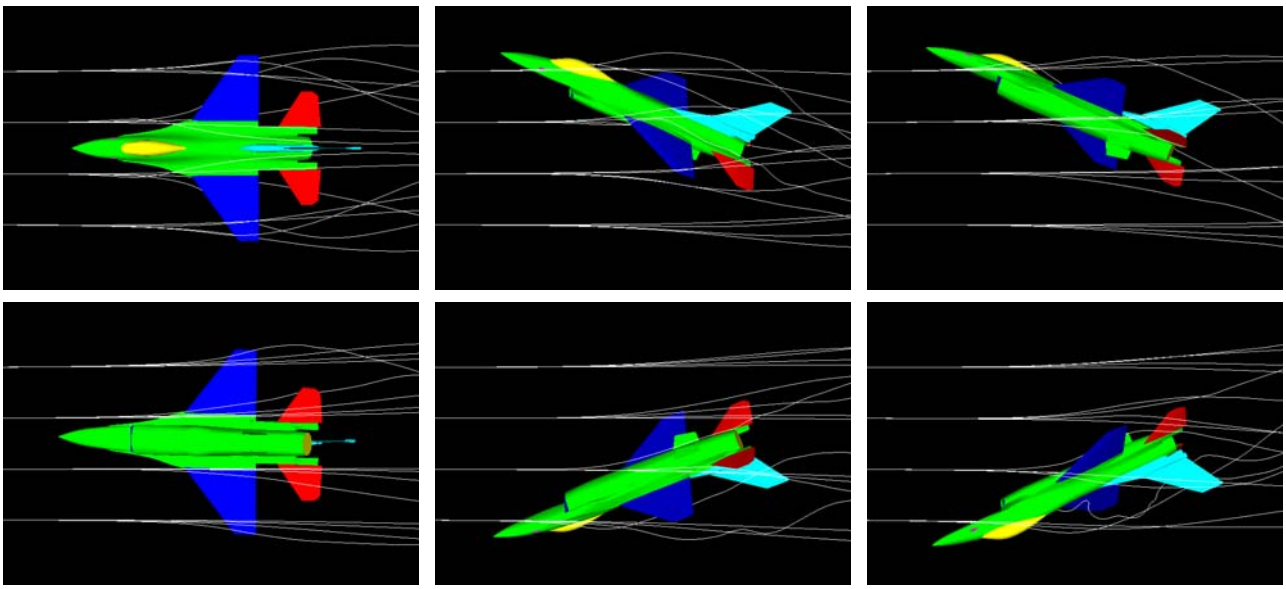

Figure 16: DES of conventional coning motion, $\alpha = \theta = 30°$, $\Omega = 180°/s$, full-span model; snapshots of instantaneous streamlines

Oscillatory Coning Motion

Oscillatory coning motion, also called inclined axis roll, is another advanced experimental technique used to determine dynamic S&C derivatives. It is identical to conventional coning motion, except that the axis of rotation is not aligned with the wind direction. This motion results in an oscillating angle of attack and sideslip at the frequency of the coning, and with magnitudes that are equal to the angle that the rotation is skewed from the velocity vector.

Figure 17 shows the results of a detached-eddy simulation of this motion with *Cobalt*. The grid was mirrored at the plane of symmetry and rotated to an initial pitch angle of 30° before the motion was initiated. The axis of rotation formed a 60° angle with the wind direction. Consequently, the angles-of-attack range was between +30° and –90°. The rotational velocity was set to a 180°/s. The angle of attack, roll rate and free-stream conditions were chosen simply to demonstrate the capabilities of the simulation environment and do not reflect actual tunnel or flight conditions. A total of 8,800 time steps, corresponding to 3.12 s of simulation time, were computed. The images in Figure 17 visualize the flow field at different instances in time during the motion. Note that time increases from left to right, top to bottom. The time interval between images is 0.33 s. Note the asymmetric flow field caused by the side-slip angles.

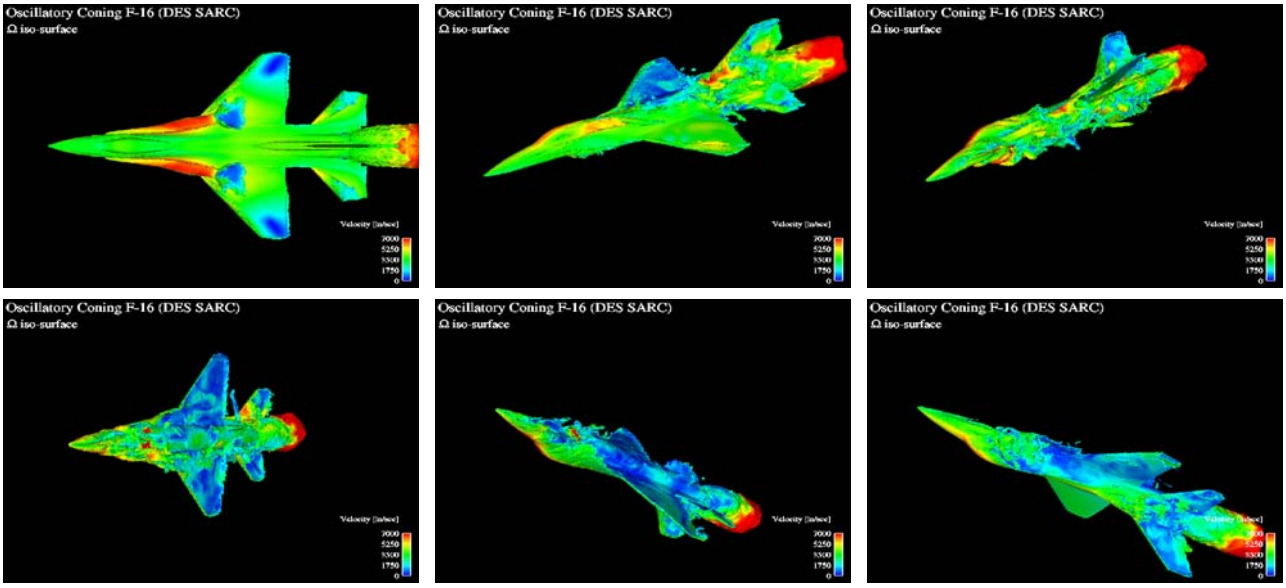

Figure 17: DES of oscillatory coning motion (inclined axis roll), α = +30° to -90°, Ω = 180°/sec, full-span model; snapshots of instantaneous iso-surface of vorticity colored by velocity magnitude

Configuration Plunge Pulse

For the half-span model of the F-16, three unsteady detached-eddy simulations with rotation correction (DES SARC) were run. Each simulation used a forced translational input in the direction normal to the free stream that varied temporally as a Gaussian pulse, i.e. the plunging input, *h*, was given by

$$h(t) = \bar{h}\, e^{-\ln(0.5)(2\cdot(t-t_0)/t_{12})^2} \tag{3}$$

where $\bar{h}$ is the amplitude of the pulse, t_0 determines the time at which the peak input occurs, and t_{12} is the pulse width at half amplitude. The parameter t_{12} determines the sharpness of the pulse and, therefore, the range of frequencies excited in the system. The three cases considered here used $\bar{h} = 0.01c$, $0.1c$ and $1.0c$ (where c is the mean aerodynamic chord), t_0 = 0.17 s, t_{12} = 0.05 s, and t = 0.02 … 0.42 s. A total of 1,050 time steps, corresponding to 0.42 s of simulation time, were computed. The simulation took about 356 CPU hours on 64 IBM Power4 CPUs.

A configuration plunge pulse gives the static lift curve slope C_{L_α} and dynamic derivatives due to angle-of-attack rate, $\dot{\alpha}$. The $\dot{\alpha}$ derivatives owe their existence to the fact that the pressure distribution on a wing or tail does not adjust itself instantaneously to its equilibrium value when the angle of attack is suddenly changed [61]. The calculation of this effect, or it measurement, involves unsteady flow, justifying the DES approach taken here. Thus, contrary to the α, β, u, p, q, r derivatives, the $\dot{\alpha}$ derivatives cannot be determined on the basis of steady-state aerodynamics. The technique gives considerable detail in the frequency domain with significant cost reduction over alternative methods of calculating multiple oscillatory responses. It can be used to extract pitch rate derivative when combined with a pitch pulse. A similar procedure can be used for lateral derivatives due to yaw rate and sideslip rate.

In Figure 18 we show a sequence of snapshots of the computed plunge maneuver with $\bar{h} = 1.0c$. The time interval between images is 0.03 s. The angle of attack is shown as a function of time in the lower left corner of each image. The snapshots depict an instantaneous vorticity iso-surface colored by velocity magnitude and a set of instantaneous streamlines.

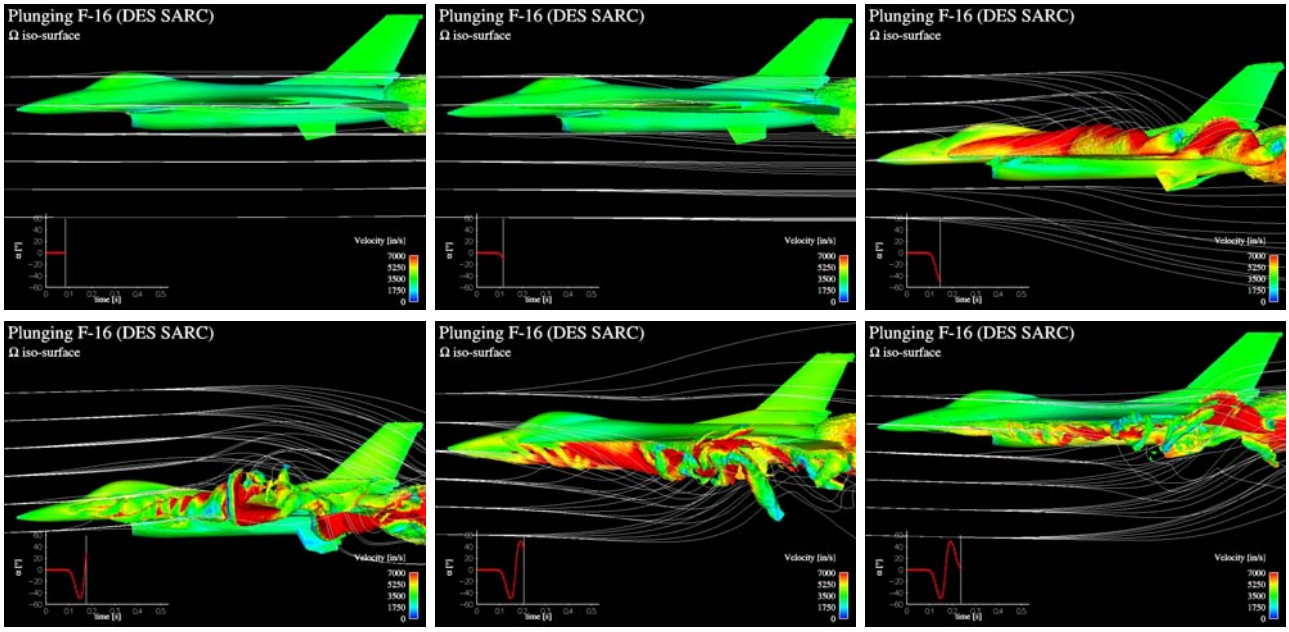

Figure 18: DES of configuration plunge pulse, $\bar{h}$ =1.0c; snapshots of instantaneous iso-surface of vorticity colored by velocity magnitude and instantaneous streamlines

The figure shows that during the maneuver, the flow over the wing and tail separates due to high induced angles of attack. For the chosen combination of pulse parameters, the angle of attack range is -50° to +50°. Note that higher amplitudes result in higher angles of attack and angle of attack rates, assuming that the pulse length is kept constant. If the amplitude is kept constant instead, longer

pulses result in lower angles of attack and angle of attack rates. Part of the problem is to find combinations of the pulse parameters that minimize the computational time but give an accurate response. Ideally, the pulse should be as sharp as possible to minimize computational time. Unfortunately, sharper pulses correspond to higher angles of attack. Thus, the pulse amplitude must be lowered to stay within the linear range.

At the time of this writing, we have not extracted any quantitative data from the simulation results yet. The procedure will be to compute the transfer function from the input and the response to the input in the form of the lift coefficient time history. The transfer function can be derived by dividing the complex Fourier transform of the response by the transform of the input. The character of the transfer function at zero frequency defines the static lift curve slope and the dynamic S&C derivative due to angle of attack rate.

4. A CFD Challenge for Stability and Control

The pitch instability of the F-16 reported in [59] and [62] illustrates critical stability and control characteristics that need to be understood early in the design phase. It also illustrates the complexity of the S&C challenge for CFD. The critical conditions are associated with separated flows with a combination of flow features and issues concerning Reynolds number effects. To the knowledge of the authors, the pitch instability of the F-16 has not been reproduced with CFD. Figure 19 shows experimental force and moment data for the F-16C (Block 25). The experimental data are for a free-stream Mach number of 0.8 and for a Reynolds number of 2.5×10^6 and 3.63×10^6, respectively, and were obtained as part of NASA's abrupt wing stall program. The pitch-instability is seen to occur at angle of attack between 12° and 15°, depending on the Reynolds number.

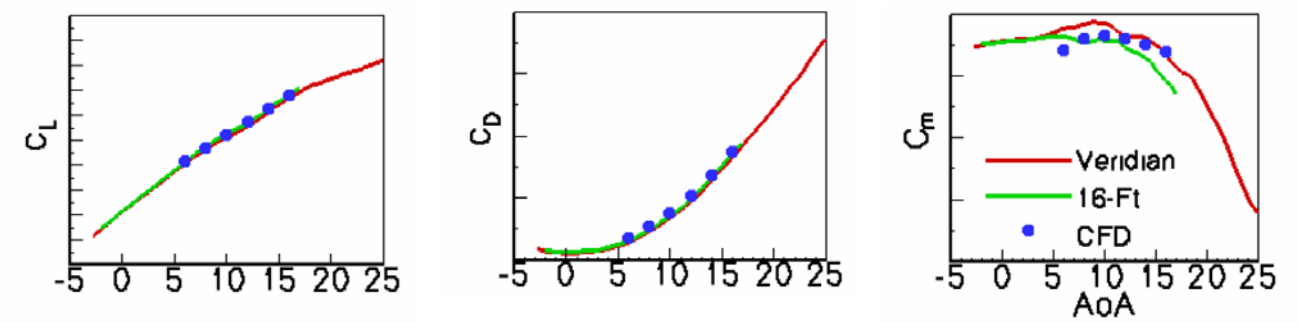

Figure 19: CFD-experiment comparison for F-16C (Block 25) configuration, M_∞=0.8; reproduced from [62]

At a free-stream Mach number of 0.2 the pitching moment instability occurs at significantly higher angles of attack. This is illustrated for three different elevator deflections by the experimental data shown in Figure 20.

Prior to going deeper into computing dynamic stability derivatives, we use the experimental data reported in [59] and [62] to formulate a CFD challenge for static stability and control. The goal is to reproduce the pitch instability of the F-16 at different Mach/Reynolds numbers and elevator deflections by performing a time-accurate, continuous α sweep using the DES capability of *Cobalt*. Time-accurate CFD may offer a faster and more efficient approach to predicting the entire C_m-alpha curve than the conventional point-by-point steady-state approach. It can provide high-density, continuous data which may not be practical to obtain by steady-state CFD. This is important for populating an S&C database, because often derivatives change sign over very small ranges of flow

conditions, so simple finite differencing over angle of attack ranges of a degree or more may not yield sufficient accuracy. This alternate approach is also more representative of a typical measurement sequence in a wind tunnel. The use of DES offers a better quantitative assessment of S&C derivatives when nonlinearities (stall, post-stall, hysteresis) and separated flows are involved.

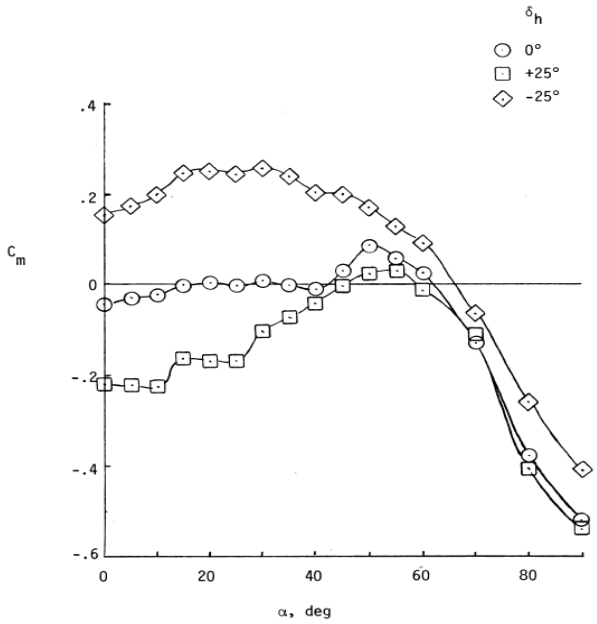

Figure 20: F-16 pitching moment coefficient as a function of angle of attack and elevator deflection angle, M_∞=0.2; reproduced from [59]

Numerical Grid and Boundary Conditions

The CAD model and numerical grid for the transonic case were generated at NASA Langley Research Center as part of another study dealing with abrupt wing stall [62]. An attempt was made to closely reproduce as many of the geometric details of the wind-tunnel test model as practical including the wing tip missile and its attachment hardware, the node boom, as well as flow through ducts representing the engine flow path. Only one side of the symmetric aircraft configuration was modeled, utilizing a plane-of-symmetry boundary condition along the aircraft centerline.

A 3D all-tetrahedral viscous grid with 1,205,202 nodes, corresponding to 6,929,816 tetrahedral cells, was generated for a half-span model of the F-16C (Block 25) at NASA Langley Research Center using the grid generation package GRIDTOOL [56] and VGRIDNS [57]. This grid was then converted to a hybrid grid in *Cobalt* format using the commercial grid management utility BLACKSMITH [58] from Cobalt Solutions, LLC. BLACKSMITH reduced the cell count to a total of 5,451,498 cells by combining highly stretched tetrahedral cells into 690,127 prismatic cells. The program generated 14 (incomplete) prismatic layers. The incomplete layers were completed with a total of 98,064 pyramids as "end caps." The transition between the prismatic layers and the tetrahedral grid is very smooth due to additional viscous tetrahedral layers that could not be combined into prisms. The surface of the half-span model of the F-16C is discretized with 141,817 triangular elements (including the flow-through duct). The surface grid is shown in Figure 21. It has been mirrored at the symmetry plane for illustrative purposes. The grid resolution off the surface is sufficient to provide a grid spacing of y^+=1 right next to the body surface based on the wind-tunnel test Reynolds number.

The boundary conditions are symmetry, adiabatic wall for the surface of the aircraft and the engine duct, and modified Riemann invariants for the far-field boundaries.

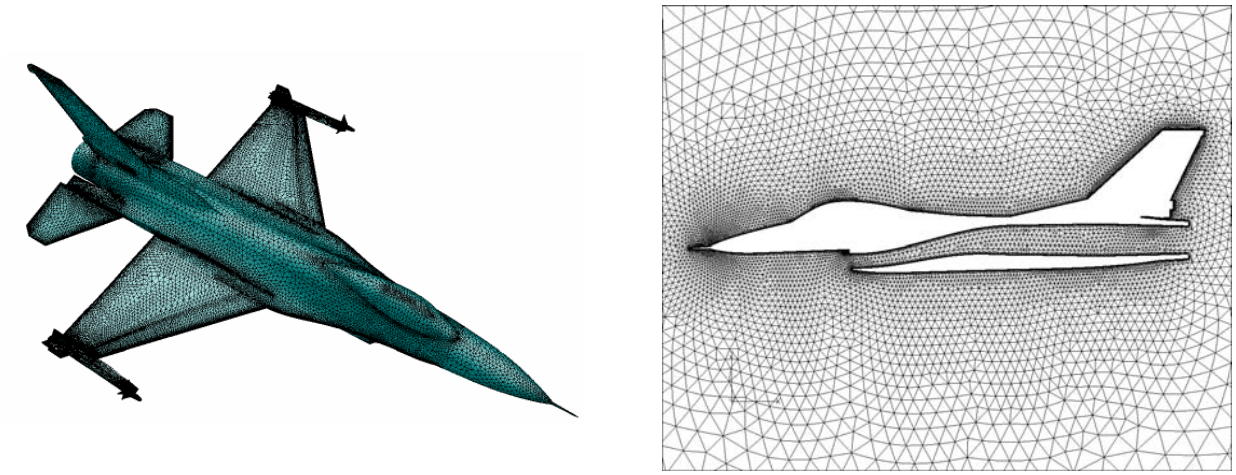

Figure 21: Unstructured numerical surface grid for the sub-scale model of the F-16C (left, reproduced from [62]) and symmetry plane of the hybrid volume grid showing the meshed flow-through duct (right)

At the time of this writing, work was under way to reproduce the C_m-alpha curve in Figure 19 numerically by slowly rotating the rigid grid around the moment reference point.

5. Conclusions and Outlook

The status of a three-year project to develop a computational method for accurately determining static and dynamic stability and control derivatives of a fighter with various weapons configurations as well as the aircraft response to pilot input has been given. Now, just over half-way through the first year for the project, the development status of three tools key to project has been given as well as some initial simulation results.

A user-friendly interface has been developed to facilitate the generation of prescribed motion definitions for the *Cobalt* flow solver, which include sinusoidal oscillations, general rotations and translations, and arbitrary motions. Functionality has also been developed to set up 6-DOF simulations. The interactive GUI will be extended to include more predefined maneuvers to choose from, such as pitch-over pull-ups, steady-sideslips, rudder doublets, wind-up turns, and coning motion. It is also planned to let the user convert flight test maneuver data into a *Cobalt* motion file. It would be beneficial to have a GUI module for previewing the motion and an expert-type input checker. Ultimately, the GUI should be able to automatically generate all the input files required for populating an S&C database (automatic test matrix generation).

The core components needed to implement moving control surfaces into unstructured CFD meshes have been developed. An efficient and robust algebraic technique has been implemented to deform volume cells due to the moving surface. Currently, the technique has only been applied to full-span control surface configurations. The technique suffers somewhat from floating point accuracy issues related to computational geometry. As a result, the preprocessor computations are fairly expensive and not as automatic as desired. As such, modifications to the core triangulation technique will be implemented in the future. Likewise, the issue of alleviating the massive cell shearing resulting from the movement of encased control surfaces will be addressed. The technique will soon be automated

for rapid modification of existing, converged grids, so that static solutions of aircraft and armament with different control surface deflections may be accomplished. Finally, the deformation scheme will be dynamically integrated into the simulation loop so that maneuvering aircraft may be evaluated.

The dynamic motions simulated so far include forced pitching oscillation, coning motion, oscillatory coning motion, and plunging motion and a continuous angle-of-attack sweep. So far, the results of the simulations were mostly qualitative in nature, illustrating the ease with which advanced flight and wind-tunnel testing techniques can be simulated with CFD. In an ongoing effort, maneuvers will be developed to efficiently determine S&C derivatives. Potential maneuvers are multi-axis forced motion, combined/arbitrary motions, wide-band input forced oscillation, 1 DOF translation motion (e.g. swaying), coning motion with superimposed forced oscillations and flight-test maneuvers. As part of this effort, the potential of different modeling approaches, such as reduced-order modeling, sensitivity analysis, response surface approximation, and indicial functions, will be investigated. Further comparisons with flight/wind tunnel S&C data and linear theory (vortex lattice) will be performed.

Acknowledgements

This research was performed while the second author held a National Research Council Research Associateship Award at the US Air Force Academy. This work was supported in part by a grant of HPC resources from the Arctic Region Supercomputing Center. The authors gratefully acknowledge the expert advice of the following individuals: Kelly Cohen, Thomas Yechout, Russell Cummings, Steven Senator at the US Air Force Academy; James Forsythe of Cobalt Solutions; Paresh Parikh, Shahyar Pirzadeh, Ed Parlette at NASA Langley; William Mason at Virgina Tech.

References

1. M. Withrow, “Dr. Raymond Gordnier Discusses the Research Direction of Advanced Computational Methods,” Air Force Research Laboratory Horizons, April 2005.
2. J. B. Vos, A. Rizzi, D. Darracq, E. H. Hirschel, “Navier-Stokes Solvers in European Aircraft Design,” Progress in Aerospace Sciences, Vol. 38, 2002.
3. F. T. Johnson, E. N. Tinoco, N. J. Yu, “Thirty Years of Development and Application of CFD at Boeing Commericial Airplanes, Seattle,” AIAA Paper 2003-3439, 16th AIAA Computational Fluid Dynamics Conference, Orlando, Florida, June 2003.
4. C. M. Fremaux, and R. M. Hall, Compilers, “*COMSAC: Computational Methods for Stability and Control*,” NASA/CP-2004-213028/PT1, April 2004.
5. C. M. Fremaux, and R. M. Hall, Compilers, “*COMSAC: Computational Methods for Stability and Control*,” NASA/CP-2004-213028/PT2, April 2004.
6. S. M. Murman, N. M. Chaderjian, S. A. Pandya, S.A., “Generation of Aerodynamic Data Using Design of Experiment and Data Fusion Approach,” *AIAA Paper 2002-0259*, Jan 2002.
7. S. E. Rogers, M. J. Aftomis, S. A. Pandya, N. M. Chaderjian, E. T. Tejnil, and J. U. Ahmad, “Automated CFD Parameter Studies on Distributed Parallel Computers,”, *AIAA 2003-4229*, Jun 2003.
8. C. Y. Tang, K. Gee, V. M. Hawke, “Development of Intelligent Agents for the Generation of Aerodynamic Data,” *AIAA Paper 2003-0458*, Jan 2003.
9. C. Y. Tang, K. Gee, S. L. Lawrence, “Generation of Aerodynamic Data Using Design of Experiment and Data Fusion Approach,” *AIAA Paper 2005-1137*, Jan 2005.
10. M. A. Park, L. L. Green, R. C. Montgomery, and D. L. Raney, “Determination of Stability and Control Derivatives Using Computational Fluid Dynamics and Automatic Differentiation,” AIAA Paper 99-3136, 1999.
11. L. L. Green, and A. M. Spence, “Applications of Computational Methods for Dynamic Stability and Control Derivatives,” *AIAA Paper 2004-0377*, 2004.

12. L. L. Green, A. M. Spence, and P. C. Murphy, “Computational Methods for Dynamic Stability and Control Derivatives,” *AIAA Paper 2004-0015*, 2004.
13. S. M. Murman, “A Reduced-Frequency Approach for Calculating Dynamic Derivatives,” *AIAA Paper 2005-0840*, 2005.
14. A. G. Godfrey, and E. M. Cliff, “Direct calculation of aerodynamic force derivatives - A sensitivity-equation approach,” AIAA *Paper 1998-393*, 1998.
15. A. C. Limache and E. M. Cliff, “Aerodynamic sensitivity theory for rotary stability derivatives,” *AIAA Paper 1999-4313*, 1999.
16. P. Crisafulli, M. Kaufman, A. A. Giunta, W. H. Mason, B. Grossman, L. T. Watson, and R. T. Haftka, “Response surface approximations for pitching moment including pitch-up in the MDO design of an HSCT,” *AIAA Paper 1996-4136*, 1996.
17. D. M. Schuster, J. W. Edwards, “Application of Computational Stability and Control Techniques Including Unsteady Aerodynamics and Aeroelastic Effects,”, NASA Symposium on Computational Methods for Stability and Control (COMSAC), September 2003.
18. C. H. Stephens, A. S. Arena, Jr., K. K. Gupta, “CFD-Based Aeroservoelastic Predictions with Comparisons to Benchmark Experimental Data,” *AIAA Paper 99-0766*.
19. M. R. Allan, K. J. Badcock, B. E. Richards, “CFD Based Simulation of Longitudinal Flight Mechanics with Control,”, *AIAA Paper 2005-0046*, 40rd AIAA Aerospace Sciences Meeting and Exhibit, Reno, Nevada, January 2005.
20. A. M. Rampurawala, K. J. Badcock, “Treatment of Forced Flap Motions for Aeroelastic Simulations of an Arrow Wing,” *AIAA Paper 2005-4962*, 23rd AIAA Applied Aerodynamics Conference, Toronto, Ontario, Canada, June 2005.
21. O. J. Boelens, B. B. Prananta, B. I. Soemarwoto, M. R. Allan, K. J. Badcock, W. Fritz, “Towards an Aero-Servo-Elastic Simulation Capability for High-Performance Fighter Aircraft,” RTO-MP-AVT-123, Paper No. 30, 2005.
22. G. D. Power, J. A. Calahan, D. L. Hensley, “A System for Moving-Body CFD Simulations on Overset Structured and Unstructured Grids,”, *AIAA Paper 2004-0716*, 42nd AIAA Aerospace Sciences Meeting and Exhibit, Reno, Nevada, January 2004.
23. M. Rizk, B. Jolly, “Aerodynamic Simulation of Bodies with Moving Components using CFD Overset Grid Methods,”, *AIAA Paper 2006-1252*, 44th AIAA Aerospace Sciences Meeting and Exhibit, Reno, Nevada, January 2006.
24. G. Fillola, M. Le Pape, M. Montagnac, “Numerical Simulations Around Wing Control Surfaces”, 24th International Congress of the Aeronautical Sciences (ICAS), 2004.
25. S. Obayashi, G. Guruswamy, “Navier-Stokes Computations for Oscillating Control Surfaces”, *AIAA Paper 1992-4431*, 1992.
26. M. Lesoinne, V. Kaila, “Meshless Aeroelastic Simulations of Aircraft with Large Control Surface Deflections,”, *AIAA Paper 2005-1089*, 43rd AIAA Aerospace Sciences Meeting and Exhibit, Reno, Nevada, January 2005.
27. J. S. Shang, “Three Decades of Accomplishments in Computational Fluid Dynamics,” Progress in Aerospace Sciences, Vol. 40, 2004.
28. A. Schutte, G. Einarsson, A. Raichle, B. Schoning, W. Monnich, J. Neumann, J. Arnold, T. Alrutz, J. Heinecke, T. Forkert, H. Schumann, “Prediction of the Unsteady Behavior of Maneuvering Aircraft by CFD, Aerodynamic, Flight-Mechanic, and Aeroelastic Coupling,”, RTO-MP-AVT-123, Paper No. 11, 2005.
29. M. Murayama, F. Togashi, K. Nakahashi, K. Matsushima, T. Kato, “Simulation of Aircraft Response to Control Surface Deflection using Unstructured Dynamic Grids,” Journal of Aircraft, Vol. 42, No. 2, March-April 2005.
30. C. L. Fenwick, C. B. Allen, “Development and Validation of Sliding Grid Technology for Time-Domain Aeroservoelastic Simulations,” *AIAA Paper 2005-4844*, AIAA, 23rd AIAA Applied Aerodynamics Conference, Toronto, Ontario, Canada, June 2005.
31. E. L. Blades, D. L. Marcum, “A Sliding Interface Method for Unsteady Unstructured Flow Simulations,” *AIAA Paper 2005-5233*, 23rd AIAA Applied Aerodynamics Conference, Toronto, Ontario, Canada, June 2005.
32. J. A. Samareh, Jamshid, “Application of Quaternions for Mesh Deformation,” NASA TM 2002-211646., April 2002.
33. A. L. Gaitonde, S. P. Fiddes, “Three Dimensional Moving Mesh Method for the Calculation of Unsteady Transonic Flows,” Aeronautical Journal, Vol. 99, No. 984, 1995.
34. S. A. Morton, R. B. Melville, M. R. Visbal, “Accuracy and Coupling Issues of Aeroelastic Navier-Stokes Solutions on Deforming Meshes,” Journal of Aircraft, Vol. 35, No. 5, September-November 1998.
35. J. T. Batina, “Unsteady Euler Algorithm with Unstructured Dynamic Mesh for Complex-Aircraft Aerodynamic Analysis,” AIAA Journal, Vol. 29, No. 3, March 1991.

36. C. Farhat, C. Degand, B. Koobus, M. Lesoinne, "Torsional Springs for Two-Dimensional Dynamic Unstructured Fluid Meshes," Computational Methods in Applied Mechanical Engineering, No. 163 (1998), 231-245.
37. C. O. E. Burg, "A Robust Unstructured Grid Movement Strategy Using Three-Dimensional Torsional Springs," *AIAA Paper 2004-2529*, 34th AIAA Fluid Dynamics Conference and Exhibit, June 2004.
38. K. Stein, T. Tezduyar, R. Benney, "Mesh Moving Techniques for Fluid-Structure Interactions with Large Displacements," Transactions of the ASME, Vol. 70, January 2003.
39. D. B. Kholodar, S. A. Morton, R. M. Cummings, "Deformation of Unstructured Viscous Grids," *AIAA Paper 2005-0926*, 43rd AIAA Aerospace Sciences Meeting and Exhibit, Jan. 2005.
40. X. Liu, N. Qin, X. Hao, "Fast Dynamic Grid Deformation Based on Delaunay Graph Mapping," Journal of Computational Physics, No. 211 (2006), 405-423.
41. J. R. Forsythe, K. A. Hoffmann, and F. F. Dieteker, "Detached-Eddy Simulation of a Supersonic Axisymmetric Base Flow with an Unstructured Flow Solver," *AIAA Paper 2000-2410*, June 2000.
42. J. R. Forsythe, and S. H. Woodson, "Unsteady CFD Calculations of Abrupt Wing Stall using Detached-Eddy Simulation," *AIAA Paper 2003-0594*, Jan 2003.
43. S. A. Morton, M. B. Steenman, R. M. Cummings, and J. R. Forsythe, "DES Grid Resolution Issues for Vortical Flows on a Delta Wing and an F-18C," *AIAA Paper 2003-1103*, Jan. 2003.
44. J. R. Forsythe, K. D. Squires, K. E. Wurtzler, and P. R. Spalart, "Detached-Eddy Simulation of Fighter Aircraft at High Alpha," *Journal of Aircraft*, Vol. 41, No. 2, pp. 193-200, 2004.
45. J. R. Forsythe, C. M. Fremaux, and R. M. Hall, "Calculation of Static and Dynamic Stability Derivatives of the F/A-18E in Abrupt Wing Stall Using RANS and DES," International Conference for CFD, Toronto, Canada, 2004.
46. M. Claus, S. A. Morton, and R. Cummings, "DES Turbulence Modelling on the C-130 Comparison Between Computational and Experimental Results," *AIAA-2005-884*, 2005.
47. S. A. Morton, and J. R. Forsythe, K. D. Squires, R. M. Cummings, "Detached-Eddy Simulations of Full Aircraft Experiencing Massively Separated Flows," C*omputational Fluid Dynamics Journal ISCFD Japan*, Vol. 13, No. 3, January, 2005.
48. W. Z. Strang, R. F. Tomaro, and M. J. Grismer, "The defining methods of $Cobalt_{60}$: a parallel, implicit, unstructured Euler/Navier-Stokes flow solver," *AIAA Paper 1999-0786*. 1999.
49. K.J. Badcock, B.E. Richards, M.A. Woodgate,"Elements of Computational Fluid Dynamics on Block Unstructured Grids Using Implicit Solvers," Progress in Aerospace Sciences, Vol. 36, 2000.
50. R. F. Tomaro, W. Z. Strang, and L. N. Sankar, "An implicit algorithm for solving time dependent flows on unstructured grids," *AIAA Paper 1997-0333*, 1997.
51. M. J. Grismer, W. Z. Strang, R. F. Tomaro, and F. C. Witzemman, "Cobalt: A Parallel, Implicit, Unstructured Euler/Navier-Stokes Solver," *Adv. Eng. Software*, Vol. 29, No. 3-6, pp. 365-373, 1998.
52. G. Karypis, K. Schloegel, and V. Kumar, "*Parmetis: Parallel Graph Partitioning and Sparse Matrix Ordering Library, Version 3.1*," Technical report, Dept. Computer Science, University of Minnesota, 2003.
53. S. Bleeck, 2004, "*Painless MATLAB graphical user interfaces for everybody*," University of Cambridge, http://www.soton.ac.uk/~bleeck/parameter/index.htm, accessed 7 Feb 2006.
54. A. Bowyer, "Computing Dirichlet Tessellations," The Computer Journal, Vol. 24, No. 2, 1981.
55. D. F. Watson, "Computing the n-Dimensional Delaunay Tessellation with Application to Voronoi Polytopes," The Computer Journal, Vol. 24, No. 2, 1981.
56. S. Samareh, GridTool: A Surface Modeling and Grid Generation Tool, Proceedings of the Workshop on Surface Modeling, Grid Generation, and Related Issues in CFD Solutions, NASA CP-3291, May 9-11, 1995.
57. S. Pirzadeh, Progress Toward A User-Oriented Unstructured Viscous Grid Generator, *AIAA Paper 96-0031*, 1996.
58. Cobalt Solutions, "Blacksmith User's Manual," Version 3.0, 2005.
59. L. T. Nguyen, M. E. Ogburn, W. P. Gilbert, K. S. Kibler, P. W. Brown, and P. L. Deal, "*Simulator Study of Stall/Post-Stall Characteristics of a Fighter Airplane With Relaxed Longitudinal Static Stability*," NASA Technical Paper 1538, Dec. 1979.
60. B. N. Pamadi, "*Performance, Stability, Dynamics, and Control of Airplanes*," AIAA Educational Series, 1998.
61. B. Etkin, "*Dynamics of flight: stability and control*", 3rd ed., Wiley, New York, 1996.
62. P. Parikh, and J. Chung, "A Computational Study of Abrupt Wing Stall (AWS) Characteristics for Various Fight Jets: Part I, F/A-18E and F-16C," *AIAA Paper 2003-0746*, 2003.