African Journal of Mathematics and Computer Science Research Vol. 2(7), pp. 142-156, August, 2009
Available online at http://www.academicjournals.org/AJMCSR
© 2009 Academic Journals

Full Length Research Paper

Slip formulation for numerical simulations of jumping paratroopers

Victor Udoewa

Centre for Engineering Research and Computational and Applied Mathematics-Cerecam, Menzies Bldg, 5th Floor, Upper Campus, University of Cape Town, Rondebosch 7701, South Africa. E-mail: udwvic001@uct.ac.za. Tel.:+27 21 650-3817. Fax +27 21 685-2281.

Accepted 18 May, 2009

The aim is to develop computational techniques for studying aerodynamic interactions between multiple objects when an object exits and separates from an aircraft. The object could be a paratrooper jumping out of a transport aircraft or a package of emergency aid dropped from a cargo plane. In all these cases, the computational challenge is to predict the dynamic behavior and path of the object, so that the separation process is safe and effective. This is a very complex problem because it has an unsteady, 3D nature and requires the solution of complex equations that govern the fluid dynamics of the object and the aircraft together, with their relative positions changing in time. In advection-dominated flows, the numerical boundary layer is excessively thick and not physically real. Computational methods used to solve such problems much address this as a numerically thick boundary layer can artificially affect the dynamics and trajectory of moving objects. In the presented research, numerically thick boundary layers were reduced for 3D flows specifically focusing on paratrooper-aircraft separation. Slip formulations were found to provide an excellent numerical approximation of a thin boundary layer for this application. A real paratrooper trajectory was numerically simulated.

Key words: Mesh resolution, boundary layer, mesh generation, boundary layer resolution, boundary layer elements, slip conditions, Navier-stokes equation, incompressible flow

INTRODUCTION

Here large-scale 3D Fluid-Object interactions (FOI) between multiple objects are numerically explored. All objects will be treated as rigid bodies. The example of a paratrooper jumping from a cargo aircraft illustrates a new application. Such a problem requires 3 solution components, the fluid dynamics (FD) simulation, the Newton particle-force calculations and the mesh moving calculations.

The gravitational and aerodynamic forces acting on the object determine its dynamic behavior and path. The aerodynamic forces greatly depend on the unsteady flow field around the aircraft. The computational tools developed, here are based upon the simultaneous solution of the 3D time-dependent Navier-stokes equations governing the incompressible airflow around the aircraft and the separating object, as well as the equations governing the motion of that object. These computational methods include suitable mesh update techniques to be used in conjunction with a computational technique, the deforming-spatial-domain/stabilized space-time (DSD/SST) formulation (Tezduyar, 1991; Tezduyar et al., 1992a, 1992b).

Previously, techniques such as Arbitrary Lagrangian-Eulerian formulation were used for moving problems with finite difference modeling, finite volume modeling (Noh, 1964; Franck and Lazarus, 1964; Trulio, 1966; Hirt et al., 1974) and finite element modeling (Donea et al., 1977; Belytschko and Kennedy, 1978; Belytschko and Liu, 1985; Hughes et al., 1981). Because the relative positions of the aircraft and the separating object are changing in time, the Navier-stokes equations governing the motion of the surrounding air need to be solved over a computational domain that changes in shape and time. The DSD/SST formulation is written over the corresponding space-time domain of the problem and can, therefore, automatically handle the changes in the spatial domain. Recently, this method has been tested on many problems, including those involving 3D domains, high

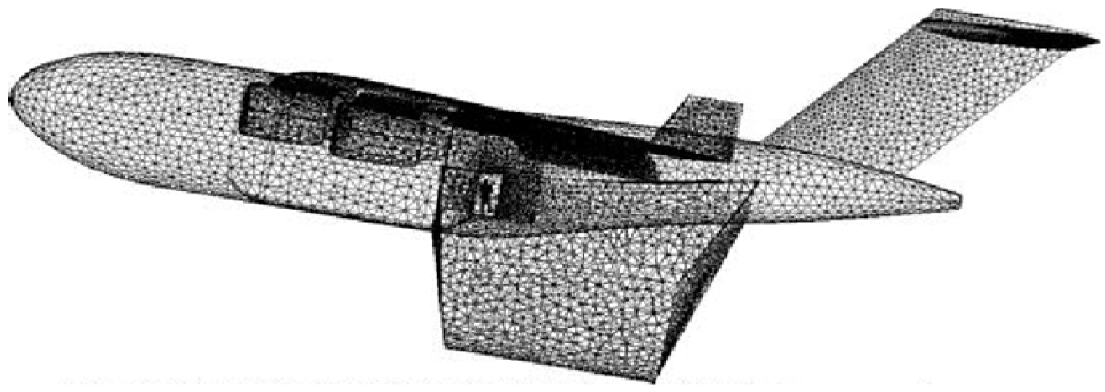

Figure 1. 3D Surface mesh of air force cargo aircraft.

Mach numbers and high Reynolds number flows and moving boundaries and interfaces (Aliabadi and Tezduyar, 1992; Behr and Tezduyar, 1994; Johnson and Tezduyar, 1994). In this work, the formulation is applied to mesh moving problems with fluid object interaction applications.

Whether using the DSD/SST method with mesh moving (Udoewa, 2005) or using a mesh moving alternative like fluid object interaction subcomputation technique (FOIST) (Udoewa, 2009), there still remains the problem of the thick numerical boundary layer around objects. This thick numerical boundary layer is not physical, but rather a result of poor resolution on the surface of the objects over which the flow is being calculated, such as an air force cargo aircraft. When calculating the flow around such an object alone, this boundary layer is less important. However, in combination with a small, separating object like a cargo payload or a jumping paratrooper, a non-physical, excessively thick numerical boundary layer can artificially affect the path of such an object.

In the actual test jumps in the field with certain cargo aircraft, paratroopers sometime experience crossover-the crossing of paths of 2 paratroopers who simultaneously jump from opposite doors of the same aircraft. In order to design computer-aided geometry changes to the aircraft to beneficially affect the paratrooper trajectories and eliminate crossover, we must first accurately simulate a real paratrooper trajectory. To do this, the boundary layer must be resolved.

Boundary layer elements are an important topic in the modeling area of computational fluid dynamics (CFD). In solid dynamics, whenever there are areas of relatively high displacement or high displacement gradients or velocities, increased refinement is required to resolve those areas properly. The same is true in fluid dynamics. When there are areas of high circulation or vorticity, for example, more elements are needed to capture such a flow with high velocity gradients.

As research has shown, to resolve the boundary layer, usually, one must utilize elements of the proper size or scale (Soinne, 2000; Gerdes et al., 1991). Sometimes, depending on the method, one must choose the proper scale factor for certain spaces of functions (Noack and Eckelmann, 1991). As an alternative to increasing the number of elements or decreasing the size of the elements, one may choose to increase the order of elements (Soinne, 2000). Higher order elements are better able to capture high order flows or flows with high gradients of the unknown.

However, if one wishes to employ a lower order of elements and solve the problem through refinement, there are also options. You can increase the number of elements indirectly through multiple grids or overset grids (Korpus, 2005). This allows one to obtain the accuracy of a structured grid while maintaining a simple unstructured one for complex geometries. Here, the attempt to solve the problem utilizes only one mesh, or one grid.

Increasing refinement within one grid presents a problem for current 3D mesh generators. The algorithm used by most 3D mesh generators distributes increased refinement throughout a 3D space in an uneven way. To clarify, imagine a domain in the shape of a rectangular prism. Inside the domain is an air force cargo aircraft. Let's assume a 3D surface mesh has been created on both the aircraft and the domain walls, so each is filled with triangular elements and nodes. The goal is to create a 3D fluid mesh between the aircraft and the domain walls. If the refinement on the walls is equal to the refinement on the aircraft, then the tetrahedra in the 3D space will roughly be the same everywhere between the aircraft and the domain walls. If increased tetrahedra are desired only on the surface of the aircraft, the refinement on the surface of the aircraft can be increased, as seen in Figure 1 contrasting with the larger refinement on the surface of the domain box shown in Figures 2, 3, and 4.

The mesh generator will use the increased refinement on the aircraft to create many smaller tetrahedra on the surface of the aircraft. The logical deduction is that the

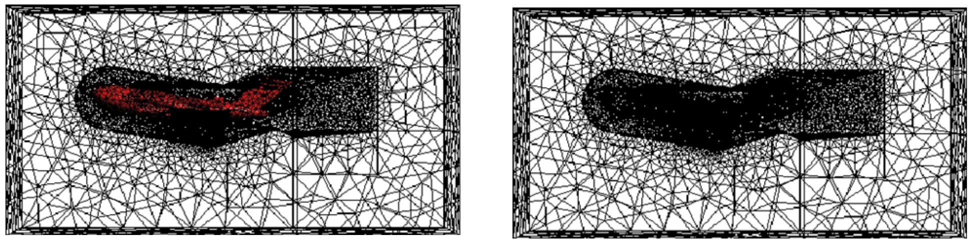

Figure 2. (A) Aircraft with Wire Frame (B) 3D Surface mesh of aircraft and domain box: front view.

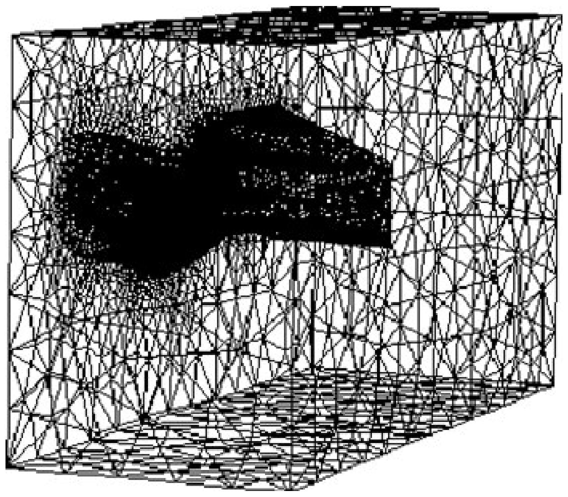

Figure 3. 3D Surface Mesh of Aircraft and Domain Box: Side View

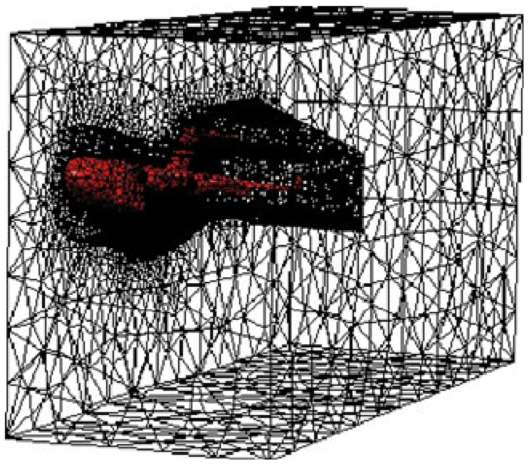

Figure 4. 3D Surface mesh of highlighted aircraft and domain Box: side view.

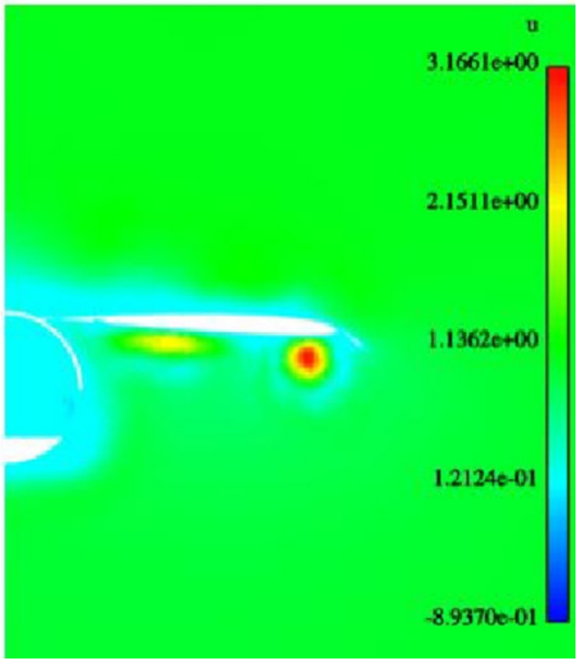

Figure 5. Color plot of velocity in boundary layer on 2D cross-section of aircraft

size of the tetrahedra will change roughly linearly between the small tetrahedra on the surface of the aircraft and the large tetrahedra bordering the domain walls. However, most automatic 3D mesh generators will favor favor the large tetrahedra corresponding to the larger refinement on the domain walls. In other words, in the 3D fluid mesh between the aircraft and the domain walls, more of the tetrahedra will be closer to the size governed by the refinement on the domain walls than that governed by the refinement on the aircraft.

Another problem with a simple refinement increase on the object with a boundary layer is that many times the refinement elsewhere in the mesh is perfectly acceptable . Only the number of elements at the surface of the object need be increased. When increasing refinement on the object in question, the increased elements throughout the fluid mesh can be wasteful as only the extra elements in close proximity to the surface of the particular object are desired. Using an automatic 3D mesh generator with the given refinement shown in Figure 1, the resulting numerical boundary layer was produced in Figure 5. Yet, after doubling the refinement and incurring extra elements

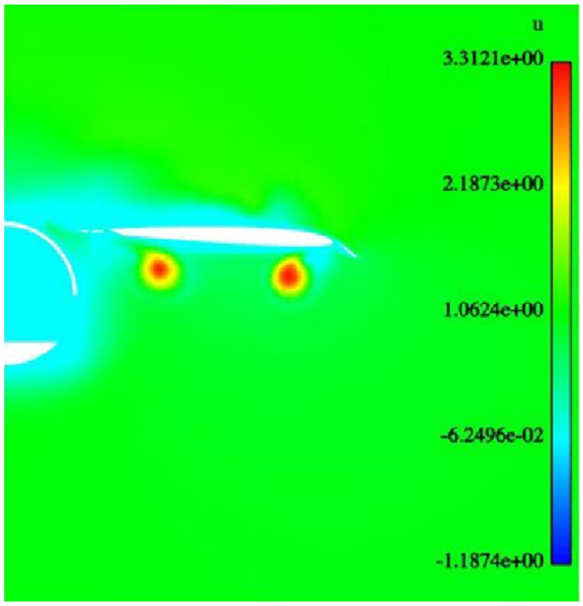

Figure 6. Color Velocity Plot in Boundary Layer with Double Refinement on the Aircraft Surface

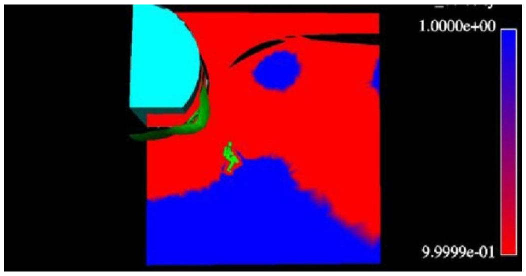

Figure 8. Dichotomous velocity picture: blue-free-stream value vs red < 100% Free stream value.

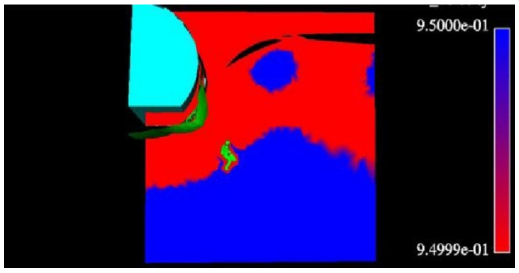

Figure 9. Dichotomous velocity picture: Blue ≥ 95% freestream value vs red < 95% freestream value.

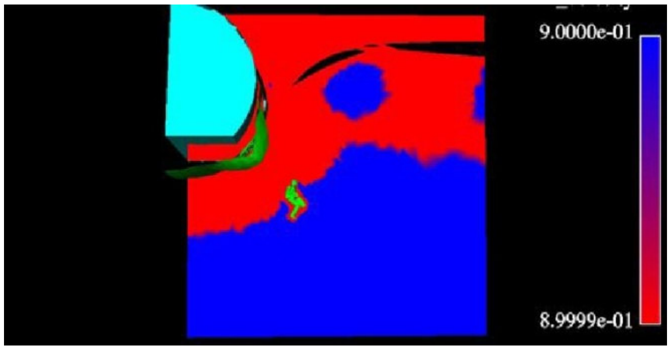

Figure 10. Dichotomous velocity picture: blue ≥ 90% free-stream value vs Red < 90% freestream value.

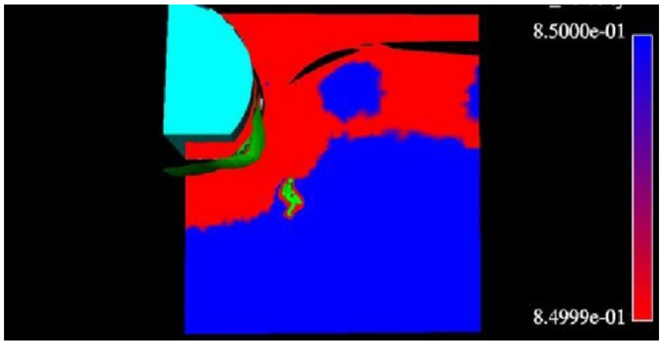

Figure 11. Dichotomous velocity picture: blue ≥ 85% freestream value vs Red < 85% freestream value.

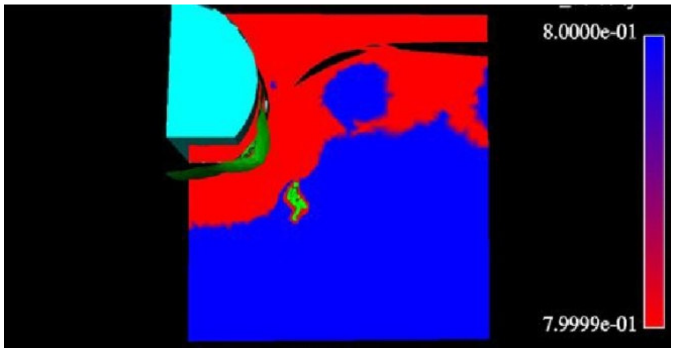

Figure 12. Dichotomous velocity picture: blue ≥ 80% free-stream value vs Red < 80% freestream value

away from the surface of the aircraft, little change can be seen from this general increased effort (Figure 6).

Boundary elements are both useful and cost efficient in that they provide increased elements only at the surface or boundary needing refinement. Developed by Johnson (1995), the automatic 3D mesh generator used here, does not have the capability to create boundary layer elements. It is possible to manually create boundary layers on boundaries aligned with the Cartesian coordinate system, but the creation of boundary layers on arbitrary geometries is a tough area of research in which Johnson and others are still currently working. Figures 8 – 13 explicate how the thickness of the numerical boundary

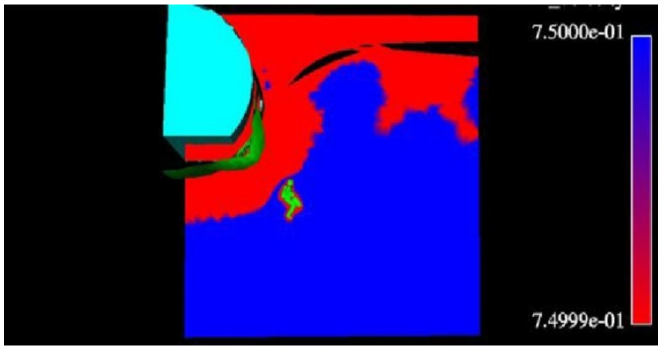

Figure 13. Dichotomous velocity picture: Blue ≥ 75% freestream value vs Red < 75% freestream value

layer affects the speed of the air hitting the paratrooper. For this reason, slip conditions are used for this new application of a jumping paratrooper. Slip conditions have been applied in numerous applications in 2D simulations such as flow around turbomachinery blades (Wooley and Hatton, 1973). It has been shown that slip conditions can simulate or approximate real behavior well (Jensen, et. al., 1980; Beetstra et al., 2007) When it is an appropriate approximation, using slip conditions is another option to resolve the boundary layer. To safely assume slip conditions, the boundary layer must be very thin relative to the object around which there is flow.

In this work, we commence with the governing equations for fluid flow. The fluid flow is governed by the Navier-stokes equations for incompressible flow. Section II is a presentation of those equations with the constitutive relations and the boundary and initial conditions. Instead of directly resolving the turbulent flow features present at the Reynolds numbers of the problems presented in this research, turbulence affects are accounted by using a zero-equation Smagorinsky turbulence model (Smagorinsky, 1963). The finite element formulations for the FD governing equations are highlighted in Section III. In this section, for the fluid-dynamics, the DSD/SST method is outlined. In the following sections, the boundary layer of an air force cargo aircraft is minimized and eliminated through 2 methods. In section IV the addition of boundary layer elements to the 3D mesh generator will be presented and its effect will be analyzed. In section v physically thin boundary layers will be approximated through three dimensional slip conditions newly implemented and applied to the 3D paratrooper-aircraft simulation.

Unless otherwise stated, computations were carried out on a CRAYT3E parallel supercomputer. All photo-graphs were obtained by permission of the US army and US air force either directly or through their web sites.

A finalizing overview of contributions and implications are made in the concluding section. This is followed by proposed areas for future research.

MATERIALS AND METHODS

Navier-stokes equations of incompressible flows

The fluid is assumed to be incompressible and will be modeled with the Navier-stokes equations of incomepressible flow. Let $\Omega_t \subset R^{n_{sd}}$ be the spatial fluid mechanics domain and $(0,T)$ be the temporal domain, where n_{sd} is the number of space dimensions, letting Γ_t denote the boundary of the spatial domain Ω_t. The subscript *"t"* evinces the time-dependence of the spatial domain and its boundary. The spatial and temporal coordinates are denoted by $\mathbf{x} = (x, y, z)$ and $t \in (0,T)$. The Navier-stokes equations of incompressible flow can be written as:

$$\rho\left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u} + \mathbf{f}\right) - \nabla\cdot\sigma = \mathbf{0} \text{ on } \Omega_t \forall t \in (0,T), \quad (1)$$

$$\nabla\cdot\mathbf{u} = 0 \text{ on } \Omega_t \forall t \in (0,T), \quad (2)$$

Where ρ is the constant fluid density, f is an external force term such as gravity and σ is the stress tensor. Equation 1 ensures the conservation of momentum for the fluid system; equation 2, the conservation of mass. The flow variables that are being calculated are the velocity, u(x,*t*), and pressure, *p*(x,*t*), which is embedded in the stress tensor defined in the next section.

Stress and strain definitions

Considering Newtonian fluids, one assumes a linear relationship between the fluid stress and rate-of-strain tensors. The stress tensor is written as the sum of its isotropic and deviatoric parts:

$$\sigma(\mathbf{u}, p) = -p\mathbf{I} + 2\mu\varepsilon(\mathbf{u}), \quad (3)$$

Where μ is the dynamic viscosity coefficient, I is the identity tensor, *p* is pressure, and ε(u) is the strain rate tensor:

$$\varepsilon(\mathbf{u}) = \frac{1}{2}(\nabla\mathbf{u} + (\nabla\mathbf{u})^T). \quad (4)$$

Boundary and initial conditions

To appropriately represent the fluid dynamics, proper boundary conditions must be imposed on the outer boundaries of the fluid domain and on any inner surfaces, ensuring an accurate fluid environment. Both the (essential) Dirichlet and (natural) Neumann-type boundary conditions are accounted for and are represented as

$$\mathbf{u} = \mathbf{g} \quad \text{on } (\Gamma_t)_g, \qquad (5)$$
$$\mathbf{n} \cdot \sigma = \mathbf{h} \quad \text{on } (\Gamma_t)_h,$$

Where $(\Gamma_t)_g$ and $(\Gamma_t)_h$ are complementary subsets of the boundary Γ_t related to the Dirichlet and Neumann-type boundary conditions, respectively.

In time-dependent problems, an initial condition is required. The initial condition on the velocity is specified as $\mathbf{u}(\mathbf{x},0) = \mathbf{u}_0$ on Ω_0, where u_0 is divergence free.

Finite element formulations

To handle the time-variant spatial domains encountered in aerodynamic separation problems, the deforming-spatial-domain/stabilized space-time (DSD/SST) formulation is used.

In order to construct the finite element function spaces for the space-time method, the time interval $(0,T)$ must be partitioned into subintervals $I_n = (t_n, t_{n+1})$, where t_n and t_{n+1} belong to an ordered series of time levels $0 = t_0 < t_1 < \Lambda < t_n = T$. Letting $\Omega_n = \Omega_{t_n}$ and $\Gamma_n = \Gamma_{t_n}$, the space-time slab Q_n is defined as the domain enclosed by the surfaces Ω_n, Ω_{n+1}, and P_n , where P_n is the surface inscribed by the boundary Γ_t as t traverses I_n. The space-time concept is depicted in Figure 7 for the space-time slab Q_n.

The surface P_n is decomposed into $(P_n)_g$ and $(P_n)_h$ with respect to the type of boundary condition (Dirichlet or Neumann, respectively) being imposed. For each space-time slab, the corresponding finite element function spaces $((S^h)_u)_n$, $((V^h)_u)_n$, $((S^h)_p)_n$, and $((V^h)_p)_n$ are defined as follows:

$$((\mathrm{S}^h)_{\mathbf{u}})_n = \{\mathbf{u}^h \mid \mathbf{u}^h \in [H^{1h}(Q_n)]^{n_{sd}}, \mathbf{u}^h =\& \mathbf{g}^h \text{ on } (P_n)_g\}, \qquad (6)$$

$$((\mathrm{V}^h)_{\mathbf{u}})_n = \{\mathbf{w}^h \mid \mathbf{w}^h \in [H^{1h}(Q_n)]^{n_{sd}}, \mathbf{w}^h =\& \mathbf{0} \text{ on } (P_n)_g\}, \qquad (7)$$

$$((\mathrm{S}^h)_p)_n = ((\mathrm{V}^h)_p)_n = \{q^h \mid q^h \in H^{1h}(Q_n)\}. \qquad (8)$$

Here $H^{1h}(Q_n)$ is the finite-dimensional function space over the space-time slab Q_n. First-order polynomials in both space and time are used to form the element domain.

The interpolation functions, however, are continuous in space but discontinuous in time.

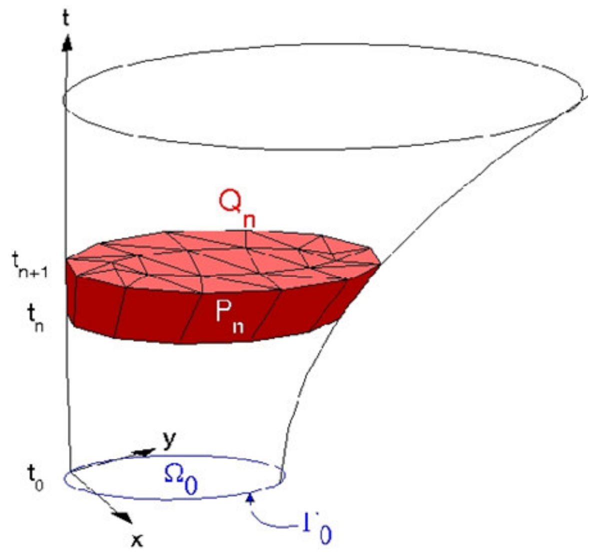

Figure 7. Space-time concept for a 2D spatial domain value.

The stabilized space-time formulation for deforming domains is now written as follows: given $(u^h)_n^-$, find $u^h \in ((S^h)_u)_n$, and $p^h \in ((S^h)_p)_n$ such that $\forall$ $w^h \in (V^h)_u)_n$ and $q^h \in ((V^h)_p)_n$,

$$\int_{Q_n} \mathbf{w}^h \cdot \rho\left(\frac{\partial \mathbf{u}^h}{\partial t} + \mathbf{u}^h \cdot \nabla \mathbf{u}^h + \mathbf{f}^h\right) dQ + \int_{Q_n} \varepsilon(\mathbf{w}^h) : \sigma(p^h, \mathbf{u}^h) dQ + \int_{Q_n} q^h \nabla \cdot \mathbf{u}^h dQ$$
$$+ \sum_{e=1}^{n_{el}} \int_{Q_n^e} \frac{\tau}{\rho}\left[\rho\left(\frac{\partial \mathbf{w}^h}{\partial t} + \mathbf{u}^h \cdot \nabla \mathbf{w}^h + \mathbf{f}^h\right) - \nabla \cdot \sigma(q^h, \mathbf{w}^h)\right] \cdot \left[\rho\left(\frac{\partial \mathbf{u}^h}{\partial t} + \mathbf{u}^h \cdot \nabla \mathbf{u}^h + \mathbf{f}^h\right) - \nabla \cdot \sigma(p^h, \mathbf{u}^h)\right] dQ$$
$$+ \sum_{e=1}^{n_{el}} \int_{Q_n^e} \delta \nabla \cdot \mathbf{w}^h \rho \nabla \cdot \mathbf{u}^h dQ + \int_{\Omega_n} (\mathbf{w}^h)_n^+ \cdot \rho((\mathbf{u}^h)_n^+ - (\mathbf{u}^h)_n^-) d\Omega = \int_{(P_n)_h} \mathbf{w}^h \cdot \mathbf{h}^h dP. \qquad (9)$$

This process is applied sequentially to all the space-time slabs Q_0, Q_1, Q_2, . . . , Q_{N-1}. The explanation of certain notation in Equation 9 follows:

$$(\mathbf{u}^h)_n^{\pm} = \lim_{\varepsilon \to 0} \mathbf{u}(t_n \pm \varepsilon), \qquad (10)$$

$$\int_{Q_n} (\Lambda) dQ = \int_{I_n} \int_{\Omega_t} (\Lambda) d\Omega dt P, \qquad (11)$$

$$\int_{P_n} (\Lambda) dP = \int_{I_n} \int_{\Gamma_t} (\Lambda) d\Gamma dt P. \qquad (12)$$

The computations start with

$$(\mathbf{u}^h)_0^- = \mathbf{u}_0. \qquad (13)$$

In the variational formulation given by Eq. 9, the first three terms, the sixth term, and the right-hand-side comprise the Galerkin formulation of the problem, the momentum balance equation and the mass balance equation. The 6th term weakly enforces continuity of the velocity field across the space-time slabs, since the interpolation functions are discontinuous in time, and since the equation is solved one space-time slab at a time. Due to numerical instabilities that occur in advection-dominated flows and from oscillations that occur when different combinations of interpolation functions for velocity and pressure are used, stabilizing terms are included. The first series of element-level integrals in Eq. 9 are least-squares terms based on the momentum equation. This term stabilizes the standard Galerkin form. The second series of element-level integrals are added to the formulation for numerical stability at high Reynolds numbers. These are least-squares terms based on the continuity equation. The stabilization coefficients τ and δ are defined at the element level. Both stabilization terms are weighted residuals, and therefore maintain the consistency of the formulation. For an exact solution, they converge to zero. Further discussions of these stabilization terms, including their derivations, can be found in literature (Mittal, 1992; Behr, 1992).

Figure 14. Colored cargo aircraft surface mesh.

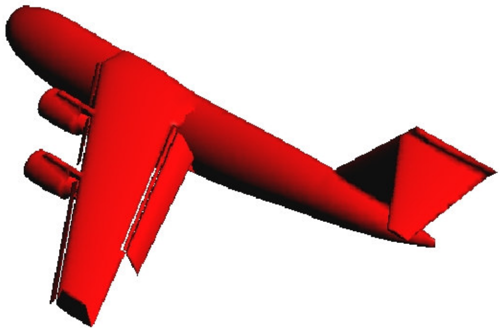
Figure 15. Colored cargo aircraft surface mesh posterior view.

Boundary layer mesh generation

The objective is to model a multi-body separation problem, but the excessively thick numerical boundary layer prevents the calculation of a realistic trajectory. To illutrate, look at Figures 8-13. They demonstrate the thickness of the numerical boundary layer through which a model paratrooper must jump. These 6 images are dichotomous pictures dividing the flow, colored by velocity, into two colors. Velocity above a certain magnitude receives a blue color. Velocity vectors with magnitudes less than the specified value receive a red color. In the first picture, Figure 8, the velocity is divided into the free-stream value (blue), denoted as 1 and any velocity with magnitude below the free-stream value (red). So in the 2D plane that slices through the model paratrooper body and the fuselage of the aircraft, we can observe a large area experiencing a magnitude of velocity less than 1, or less than the free stream value.

Moreover, we can still see a large boundary layer by examining Figure 9. Here the blue represents all magnitudes 95% of the free stream value and higher. Red represents all velocity magnitudes less than 95% of the free stream value (normalized to 1). The red area still appears very large. Figure 10 uses the same process to divide the flow through the plane into the portion containing velocity magnitudes that are 90% of the free stream value and higher (blue) and those that are less than 90% of the free stream value (red). Even here the red boundary layer is still thick. We continue in this same fashion until we reach a percentage for which the paratrooper has completely exited from the red boundary layer within the observed 2D plane (Figures 11 and 12). This point is not reached until we divide the flow field through the plane using 75% of the free stream value as the division point (Figure 13). Surprisingly, from Figures 14-16, the paratrooper does not exit the numerical boundary layer flows effectively until 75% of the free stream velocity magnitude and higher magnitudes are considered free stream velocity magnitudes. This means there are parts of the paratrooper body that feel reduced drag, reduced by up to 25%.

It is clear why the paratrooper does not feel the proper air drag. The numerical boundary layer is too unrealistically thick. It is not physical. This nonphysical numerical boundary layer must be reduced. The first attempt was to do it through mesh generation and the creation of boundary layer elements through increased refinement.

Here, 2 new companion programs were created to act on 3D meshes after they have been generated by the automatic 3D mesh generator. These programs add 3D boundary layer tetrahedra to arbitrarily shaped boundaries, representing 2 variants of boundary layer creation. Few programs can produce tetrahedral boundary layer elements. It is difficult to find programs that produce such boundary layer elements on arbitrarily shaped surface geometries. Usually if such a tetrahedral boun-

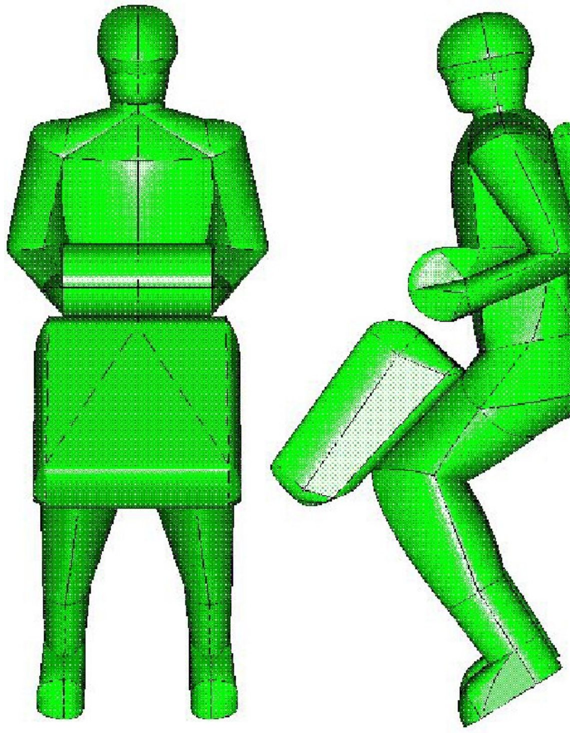

Figure 16. Paratrooper model with parachute pack.

dary layer mesh is needed on an arbitrarily shaped surface, it must be generated manually. These two programs do this automatically. A layer of tetrahedral elements must first be defined. A layer of tetrahedral elements on a surface or boundary is defined as all tetrahedral having a face or a point on the specified boundary or surface. A layer above that initial layer would include all tetrahedra bordering any tetrahedra in the first layer. They may border by a point or a face. The effects of both variant programs on the boundary layer will be observed.

The first program creates even sized layers of the same refinement. For example, one can create 2, 4, or 8 layers of tetrahedral boundary elements, but all layers are the same size. One must also remember that with each added layer of boundary elements, the computational costs increase. This motivated a second program. This program creates layers of tetrahedra with a growth factor of 2. In other words, there is a first layer of tetrahedral attached to the surface of the object. The second layer, attached to the first layer, will have elements that are roughly twice the size of those in the first layer; therefore it is less refined. Likewise, the third layer would have elements, on average, twice the size of those in the second layer and 4 times the size of those in the first layer. This was an attempt to save costs as the finite element program computed velocities and pressures farther away from the surface of the boundary in question. Results for both programs can be seen in section III.

3D slip formulation

Employing slip boundary conditions allows the fluid particles to slip, stress-free, across the arbitrarily shaped boundary. The numerical boundary layer should thus be severely reduced or eliminated. The effects should be more pronounced then the use of boundary layer elements. First, the implementation will be discussed referencing Le Beau's work (LeBeau, 1990) and following his structure, applied here in 3D for incompressible flows. Then its effectiveness on the numerical boundary layer of the cargo aircraft will be examined in section III. A slip boundary condition arises when the normal component of the velocity of a boundary is set to zero and the tangenttial components are assigned stress-free conditions. We wish to utilize this formulation to eliminate the boundary layer of the cargo aircraft. In the past, this was first done explicitly. The “killing” (setting to zero) of the normal component of velocity was attempted. The rotation of velocity vectors while maintaining the kinetic energy of the fluid particles was also implemented explicitly. The drawback is the decrease in the stability of the algorithm due to this explicit “killing” of the normal component. In order to maintain the stability of the algorithm, an implicit method is chosen here. Following Le Beau's structural outline for 2D slip boundary conditions using the Euler equations (LeBeau, 1990), recall that, in the finite element formulation, velocity within an element is the sum of the value of the shape functions for each node multiplied by the velocity value for the corresponding nodes. Thus, the velocity can be represented by

$$\mathbf{U} = \sum_{B=1}^{n} N^B \mathbf{U}^B, \tag{14}$$

Where U^B and N^B represent the velocity and shape function values at node B, respectively, and n represents the number of elemental nodes. For a 3D tetrahedral element, this can be expanded in vector form to show

$$\mathbf{U} = \begin{pmatrix} U_1^1 \\ U_1^2 \\ U_1^3 \\ U_1^4 \end{pmatrix} N^1 + \begin{pmatrix} U_2^1 \\ U_2^2 \\ U_2^3 \\ U_2^4 \end{pmatrix} N^2 + \begin{pmatrix} U_3^1 \\ U_3^2 \\ U_3^3 \\ U_3^4 \end{pmatrix} N^3 + \begin{pmatrix} U_4^1 \\ U_4^2 \\ U_4^3 \\ U_4^4 \end{pmatrix} N^4, \tag{15}$$

where the superscripts represent the degrees of freedom of the unknown. Those unknowns are the 3 components of velocity, u, v, and w, in the cartesian directions and pressure p.

When velocities or stresses are specified on the boundary (Dirichlet or Neumann boundary conditions), sometimes the shape of the boundary is arbitrary and it is not aligned with the cartesian coordinate frame. In such cases, a nodal vector is defined within the reference frame formed by normal and tangential vectors at that point such as

$$\mathbf{d} = \begin{pmatrix} u_\chi \\ u_\varphi \\ u_n \\ p \end{pmatrix}, \tag{16}$$

Where u represents velocity, n signifies the normal component at that node, and χ and φ are the orthogonally tangential components at the given node. Now that the coordinate system is in the local frame of the node, velocity values can be set directly for Dirichlet boundary conditions. In practice, the normal component, u_n, is typically assigned a value for such conditions.

The problem lies in having multiple nodes, each with a velocity vector in its own local coordinate system. Any nodes on slip boundaries with arbitrary shapes will all have vectors of unknowns in their local coordinate systems. In order to solve the global system of equations for the unknown velocity vectors, it is necessary to have all nodal vectors in the same coordinate frame. Thus there is a need to transform the nodal vectors in various local coordinate systems to the cartesian coordinate frame. Then the governing equations can be solved for the unknowns. Equation 15 can be written to show

$$\mathbf{U} = \begin{pmatrix} U_1^1 \\ U_1^2 \\ U_1^3 \\ U_1^4 \end{pmatrix} N^1 + \begin{pmatrix} U_2^1 \\ U_2^2 \\ U_2^3 \\ U_2^4 \end{pmatrix} N^2 + \mathbf{T}^3 \begin{pmatrix} d_3^1 \\ d_3^2 \\ d_3^3 \\ d_3^4 \end{pmatrix} N^3 + \mathbf{T}^4 \begin{pmatrix} d_4^1 \\ d_4^2 \\ d_4^3 \\ d_4^4 \end{pmatrix} N^4, \tag{17}$$

Where nodes 3 and 4 lie on the boundary and T^i represents a rotation matrix that rotates the local coordinate frame of node *i* to the Cartesian coordinate reference. In 3D, the rotation matrix T should be a matrix including rotations about multiple axes to align the local coordinate axes with the Cartesian axes. There will usually be multiple angles of rotation. Rewriting equation 14 for boundary nodes we find

$$\mathbf{U} = \sum_{B=1}^{n} N^B \mathbf{T}^B \mathbf{d}^B. \tag{18}$$

This rotation must also effect the test functions in the same manner, generally defining the test function, W, as

$$\mathbf{W} = \sum_{A=1}^{n} N^A \mathbf{c}^A. \tag{19}$$

For slip boundary nodes on arbitrary surfaces, the equation is modified to include the rotation matrix; thus,

$$\mathbf{W} = \sum_{A=1}^{n} N^A \mathbf{T}^A \mathbf{c}^A. \tag{20}$$

These new definitions in equations 18 and 20 can be generalized so that for nodes c and d are already in the cartesian coordinate reference frame. With that generalization that are not on slip surfaces, the matrix T represents the identity matrix and the vectors one can substitute those definitions in the DSD/SST finite element formulation found in equation 9. Ignoring the stabilization terms for now, we obtain

$$\begin{aligned}
&\int_{Q_n} \Bigg(\sum_{A=1}^{n} N^A \mathbf{T}^A \mathbf{c}^A \cdot \rho \sum_{B=1}^{n} \frac{\partial N^B}{\partial t} \mathbf{T}^B \mathbf{d}^B + \sum_{A=1}^{n} N^A \mathbf{T}^A \mathbf{c}^A \cdot \rho \left(\sum_{B=1}^{n} \frac{\partial N^B}{\partial k} \mathbf{T}^B \mathbf{d}_k^B \right) \sum_{B=1}^{n} N^B \mathbf{T}^B \mathbf{d}^B \\
&+ \sum_{A=1}^{n} N^A \mathbf{T}^A \mathbf{c}^A \cdot \mathbf{f}^h - \sum_{A=1}^{n} \frac{\partial N^A}{\partial x_i} \mathbf{T}^A \mathbf{c}_i^A \sum_{B=1}^{n} N^B p^B \\
&+ \mu \left(\sum_{A=1}^{n} \sum_{B=1}^{n} \left(\delta_{ij} \frac{\partial N^A}{\partial x_k} \mathbf{T}^A \mathbf{c}_i^A \cdot \frac{\partial N^B}{\partial x_k} \mathbf{T}^B \mathbf{d}_j^B + \frac{\partial N^A}{\partial x_j} \mathbf{T}^A \mathbf{c}_i^A \cdot \frac{\partial N^B}{\partial x_i} \mathbf{T}^B \mathbf{d}_j^B \right) \right) + q^h \sum_{B=1}^{n} \frac{\partial N^B}{\partial x_j} \mathbf{T}^B \mathbf{d}_j^B \Bigg) dQ \\
&+ \int_{\Omega_n} \left(\sum_{A=1}^{n} N^A \mathbf{T}^A \mathbf{c}^{A+} \cdot \rho \sum_{B=1}^{n} N^B \mathbf{T}^B \mathbf{d}^{B+} - \sum_{A=1}^{n} N^A \mathbf{T}^A \mathbf{c}^A \cdot (\mathbf{u}^h)_n^- \right) d\Omega = \int_{(P_n)_h} \sum_{A=1}^{n} N^A \mathbf{T}^A \mathbf{c}^A \cdot \mathbf{h}^h dP,
\end{aligned} \tag{21}$$

Where the superscript i, j, and k refer to degrees of freedom within each nodal block, while d^B and p^B represent the velocity and pressure components of the unknown vector, respectively. Since the c^A's are arbitrary, this can be rewritten as

$$\begin{aligned}
&\sum_{B=1}^{n} \int_{Q_n} \Bigg(N^A (\mathbf{T}^A)^T \mathbf{T}^B \rho \frac{\partial N^B}{\partial t} \mathbf{d}^B + \sum_{A=1}^{n} N^A (\mathbf{T}^A)^T \mathbf{T}^B \rho \left(\sum_{B=1}^{n} \frac{\partial N^B}{\partial k} \mathbf{T}^B \mathbf{d}_k^B \right) N^B \mathbf{d}^B \\
&+ N^A (\mathbf{T}^A)^T \mathbf{f}^h - \frac{\partial N^A}{\partial x_i} (\mathbf{T}^A)^T N^B p^B \\
&+ \mu \left(\delta_{ij} \frac{\partial N^A}{\partial x_k} (\mathbf{T}^A)^T \mathbf{T}^B \frac{\partial N^B}{\partial x_k} \mathbf{d}_j^B + \frac{\partial N^A}{\partial x_j} (\mathbf{T}^A)^T \mathbf{T}^B \frac{\partial N^B}{\partial x_i} \mathbf{d}_j^B \right) + q^h \frac{\partial N^B}{\partial x_j} \mathbf{T}^B \mathbf{d}_j^B \Bigg) dQ \\
&+ \int_{\Omega_n} \left(\sum_{A=1}^{n} N^A (\mathbf{T}^A)^T \mathbf{T}^B \rho N^B \mathbf{d}^{B+} - N^A (\mathbf{T}^A)^T (\mathbf{u}^h)_n^- \right) d\Omega = \int_{(P_n)_h} N^A (\mathbf{T}^A)^T \mathbf{h}^h dP.
\end{aligned} \tag{22}$$

The nonlinear advective term will be solved in separate sub-terms where one unknown is frozen and treated as known while the other is treated as a variable unknown. So in practice, the two summations would not be completed in the nonlinear advective term.

Interestingly, such rotations could considerably increase the computational costs, so reduction of rotational costs becomes important in such slip boundary problems. Since the many nodes in a given simulation are not slip boundary nodes, the rotations do not need to be employed for every element. Utilizing the rotations only when needed is the first cost-saving method. Secondly, the rotations need not occur until after the entire formation. The rotations can be moved outside the summations, outside the entire element level loop, to avoid rotating every term inside the elemental and inner shape

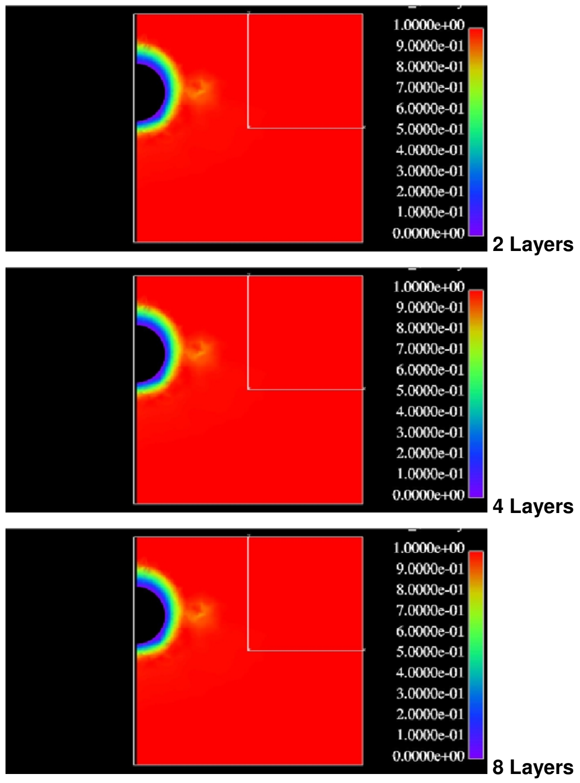

Figure 17. Velocity color plot: boundary layer around fuselage of cargo.

function loops. This saves time and also allows for the vectorization of the element level loops.

It should also be noted that another new implementation of the slip boundary condition using Navier-stokes equation for incompressible flow is in the calculation of the 3D point-normals. At a particular point that is the intersection of multiple triangles (sides of tetrahedra) in space, the normal is usually calculated by weighting the normals of the various touching triangles (faces). The weighting factor of a normal of a particular triangle is usually

$$w_f = \frac{A}{A_t}, \tag{23}$$

Where A is the area of the triangular side and A_t is the total cumulative area of all triangular sides at that point.

In this implementation, a different weighting factor based on the angle at that point is used. So regardless of the area of a triangle, if the angle of the triangle at a considered point is large, the angle increasingly affects the direction of the point-normal. This presents a new weighting factor equation,

$$w_f = \frac{\alpha}{\alpha_t}, \tag{24}$$

Where α is the angle made by the corner of the triangle in question at the specific point and α_t is the total sum of all angles made at that point by bordering triangles. On a slip surface, the normals at a particular point are then calculated by summing the weighted normals of the bordering triangles.

RESULTS AND DISCUSSION

For all methods, flow around a cargo aircraft was simulated in order to analyze the boundary layer and eventually model a paratrooper falling from the cargo aircraft.

In modeling the cargo aircraft, symmetry was assumed with respect to the plane passing through the middle of the aircraft. Only half of the aircraft was modeled for this simulation of a jumping paratrooper exiting the cargo aircraft.

Three-dimensional triangular surface meshes were then created from the models and then a 3D tetrahedral volume mesh was generated for the fluid dynamics solution using these surface meshes. The volume mesh consisted of 129,090 nodes and 728,902 tetrahedral elements. The modeling software, 3D surface mesh generator and automatic 3D mesh generator were all developed by Johnson (Johnson and Tezduyar, 1997).

Even boundary layers

The first variation of the boundary layer program created evenly sized layers of tetrahedra on the surface of the specified boundary. Of course, the costs increase with increased number of elements. It is infeasible to continually increase the layers of boundary elements without end, as the aircraft is a very large object with a very small refinement on its surface. Three different trials were attempted.

Those 3 trials were a 2-layer tetrahedral boundary layer, a 4-layer tetrahedral boundary layer and an 8-layer tetrahedral boundary layer. The resulting boundary layers are viewable in Figures 17 and 18. Upon first inspection, it seems that the boundary layer elements make little or no difference. There is a small decrease in the size of the boundary layer, but the decrease seems imperceptible. Decreasing the thickness of the numerical boundary layer around the cargo aircraft becomes a difficult task because as you move from the nose of the aircraft to the tail, the perimeter and effective diameter of the fuselage increase as extensions or protrusions to the fuselage are encountered. This creates areas of decreased velocity as the air flow must come to a stop or point of zero velocity at additional stagnation points. So the small decrease in the thickness of the numerical boundary layer is affected by the enlarging shape of the fuselage. Otherwise, boundary layer elements have been proven to effectively increase the resolution of the boundary layer when the

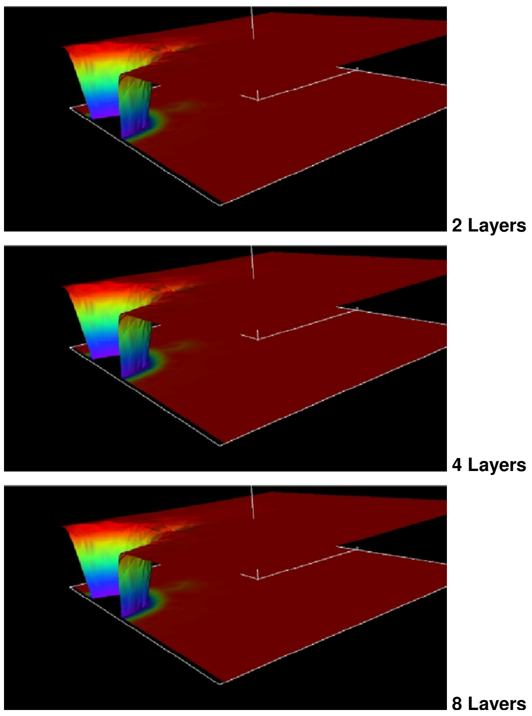

Figure 18. Velocity color 3D plot: Boundary layer around fuselage of cargo aircraft: Side view.

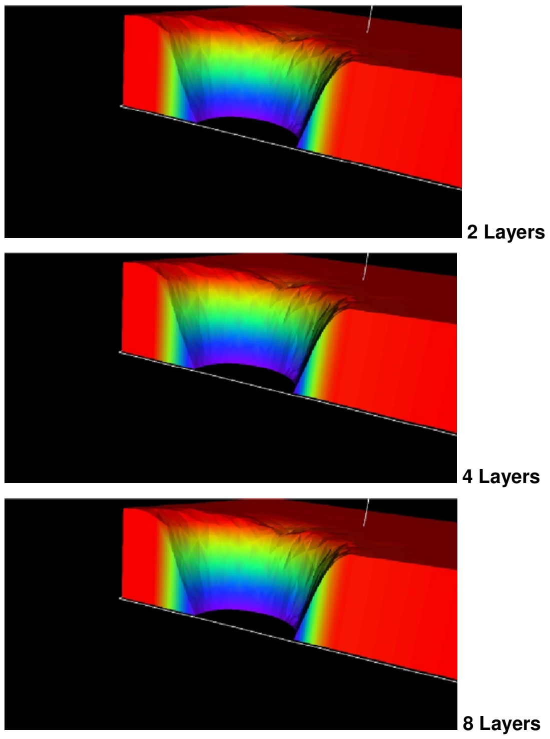

Figure 20. Boundary layer velocity surface around fuselage of cargo aircraft: Side view.

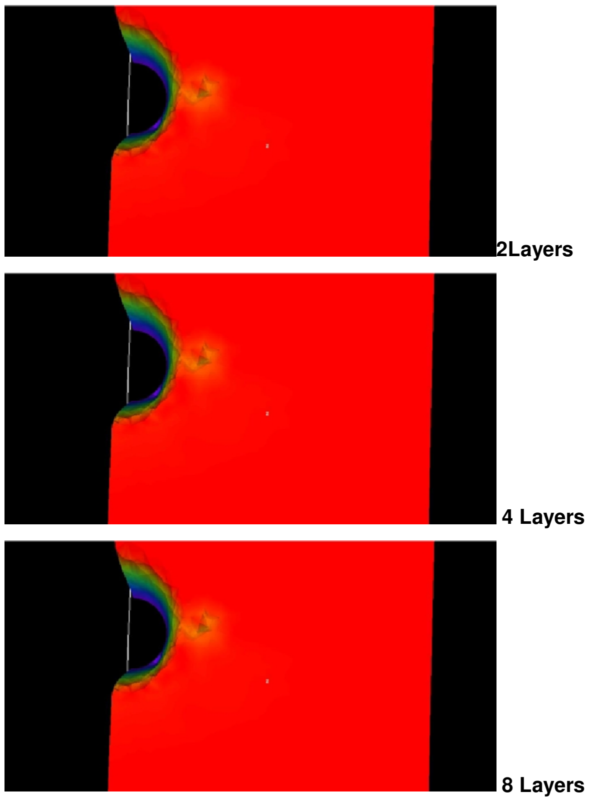

Figure 19. Boundary layer velocity surface around fuselage of cargo aircraft.

physical flow actually experiences only a very thin layer, (Schlichting, 1979; Johnson and Tezduyar, 1995). Additionally, the cargo aircraft is flying at an angle of attack of 8°, so separation occurs sooner on the top side of the cargo aircraft than if it were flying at a 0° angle of attack. This creates a slightly larger boundary layer on the top side of the fuselage than on the bottom side. Finally, with unlimited resources and refinement, the boundary layer could be reduced perceptibly. However our goal is to minimize it with reasonable refinement and limited resources.

Boundary layers with internal growth

The second program created layers of tetrahedra with a growth factor of 2 moving away from the aircraft into the fluid mesh towards the domain walls. If the first layer of boundary elements were attached to the surface of the cargo aircraft, then elements in the second layer of boundary elements had roughly twice the element length or size of those in the first layer of boundary elements.

In Figures 19 and 20, the same problems occur. A considerable decrease in the size of the numerical boundary layer is not observed. The same explanations apply, too. We expect boundary layers elements to resolve the boundary well on objects with constant diameter such as

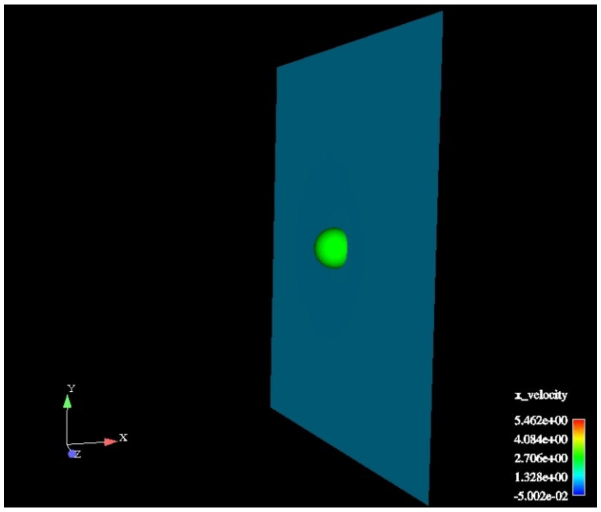

Figure 21. Velocity in free-stream direction about ball with slip conditions.

a 3D cylinder. The same difficulties arrive in the boundary layer of aircraft in which there is fuselage diameter growth.

Remember, with unlimited computer memory, the boundary layer can be resolved with increasingly higher numbers of elements. As long as the average element size on the surface of the aircraft is less than the thickness of the boundary layer, it can be resolved. Again, our goal is to resolve the boundary layer with limited elements and resources. In cases when the boundary layer is expected to be very thin, thinner than the numerical results show, and the Reynolds number is very high, one may approximate the flow with slip conditions on the specified object. This provides another option or means of reducing and possibly eliminating the numerical boundary layer.

Slip formulation

The effects of the 3D slip condition on an arbitrary boundary must be validated. Before observing its influence on the boundary layer of the cargo aircraft, we observed its effect on the flow around a ball. Using a slip boundary condition on the surface of the ball, Figure 21 clearly displays the boundary layer observed around the ball. It is nonexistent or, at least, imperceptible. This is what is expected. The accuracy was also be tested. Compared with an analytical solution for inviscid flow about a sphere, the norm of the error vector (vector containing the error for every node in the simulation) was 0.001. Ostensibly, the slip code approximated an inviscid simulation about a sphere very well. The observed error is simply due to imperfection of the modeling of the sphere and the lack of symmetry. If the refinement is increased, the error decreases. Repeating the same simulation with twice the refinement gave an error norm of 0.00001.

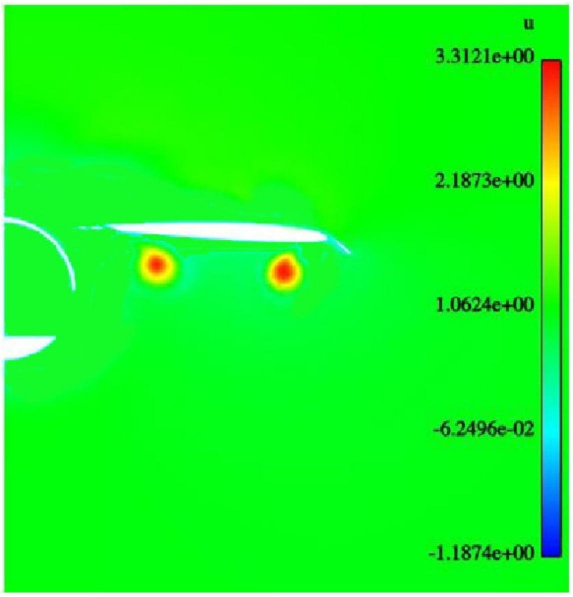

Figure 22. Velocity in free-stream direction about cargo aircraft with slip conditions.

This is an improvement that demonstrates the source of the error is as conjectured. Now that the slip boundary-condition code is validated, its effectiveness was tested on the cargo aircraft problem in order to see if it helped resolve the thick boundary layer. In Figure 22, there is no noticeable boundary layer, the boundary layer was resolved. Such a slip implementation can be used with objects separating from the aircraft to avoid the negative interaction with a false, numerically thick boundary layer.

Looking at Figure 23, we observe that the trajectory computed using the slip approximation was more drag-dominated versus the normal no-slip computation which is more gravity-dominated. Ostensibly, the slip approximation followed a more realistic trajectory. Remember, again, the goal is to model a realistic trajectory, so that, with this information, we can then design changes to the aircraft geometry and observe if the paratrooper path is beneficially affected. We now have a viable realistic trajectory with the slip approximation. Looking closely, it can be seen that the paratrooper was perhaps experiencing too much drag.

He appears to have been blown back excessively by the air. This is probably attributable to the lack of a spoiler door on this aircraft model. Such a cargo aircraft normally has a spoiler door that is hinged to one side of the opening through which the paratroopers jump. When the paratroopers prepare to jump, the spoiler door is opened to a position perpendicular, or normal, to the fuselage of the aircraft. It is important to note this spoiler door is opened to the upstream side of the opening. In Figure 24, it would be to the right hand side of the jumping paratrooper. The spoiler door contains holes or grooves that

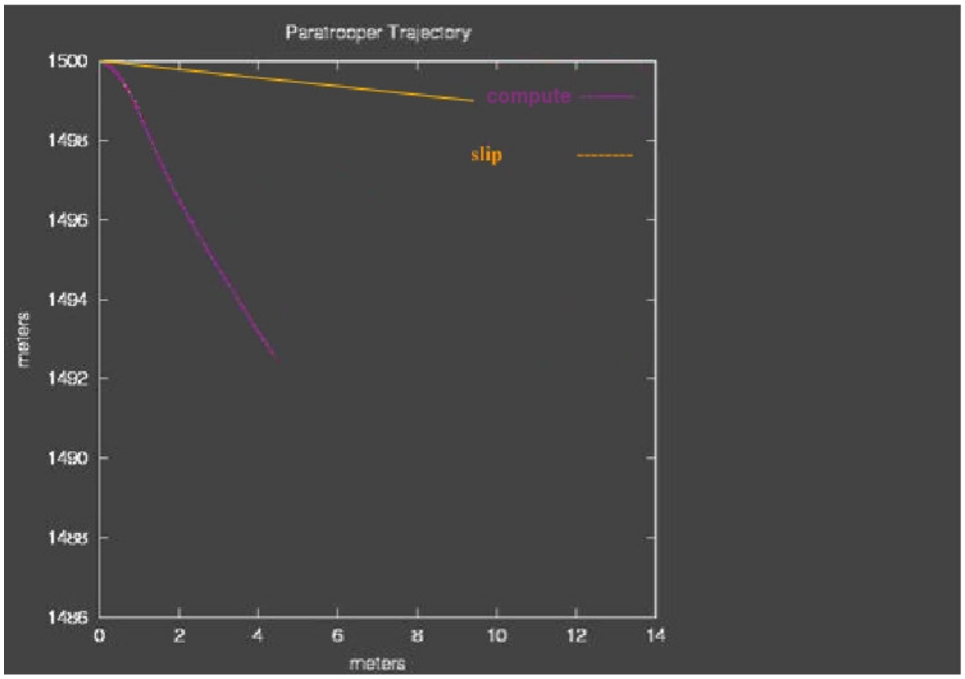

Figure 23. Paratrooper trajectory comparison: Numerical (no-slip) vs slip formulation.

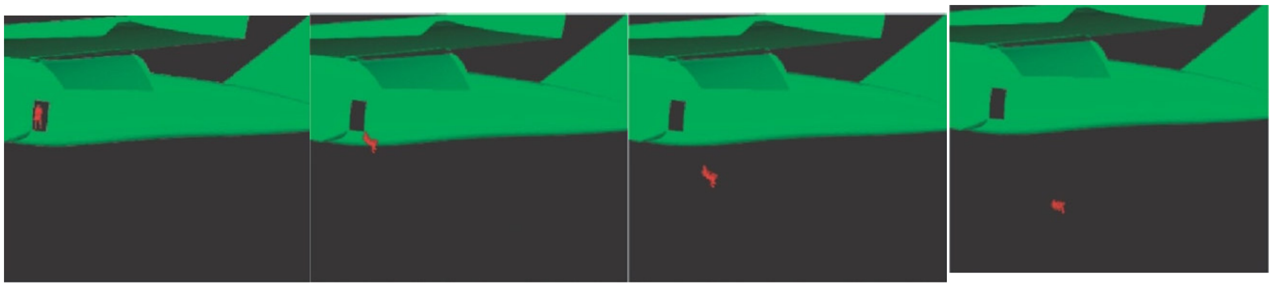

Figure 24. Paratrooper trajectory with no-slip conditions aircraft.

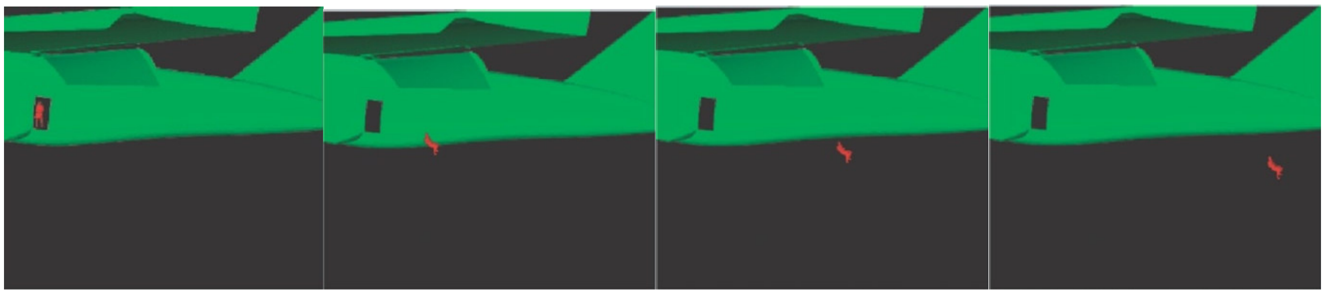

Figure 25. Paratrooper trajectory with slip conditions on aircraft.

direct or deflect the upstream air flow downward as it travels through the door. This effectuates a decreased drag force on the paratrooper directly downstream of the door. Once the paratrooper clears the door, the drag force he feels returns to full free-stream drag again. A spoiler door would change the trajectory of the jumping paratrooper so that he would not be blown back as much, experiencing so much drag. However without a spoiler door, the results were encouraging, and Figure 25 illuminated this.

Conclusions

Though tetrahedral boundary layers for arbitrary shapes were achieved, it was the slip formulation applied to the paratrooper problem that properly simulated the flow around the cargo aircraft. The slip formulation introduced here found the normals for arbitrary 3D shapes by a new weighting method based on the angles of the bordering 2D triangles. This formulation removed the thick boundary layer poorly approximating a very thin physical boundary layer. This increased the drag felt for separating objects. The paratrooper, for example, was clearly swept back more with a larger drag force. Geometry changes can now be made to the aircraft to beneficially affect the paratrooper trajectory. Much can be done for future directions. The biggest change can be made to the actual geometric model of the aircraft by adding a spoiler door and observing the affected change in the paratrooper path. Also the model of the paratrooper can of course be made flexible with rotating and moveable joints. Crosswind should also be added for more realistic effects.

ACKNOWLEDGMENTS

The author would like to thank Professor Richard Tapia at Rice University for his help. Thanks is also expressed to the US Army Natick Soldier Center, NASA Johnson Space Center, the Alliances for Graduate Education and the Professoriate, T*AFSM, NSF, NASA/Texas Space Grant Consortium, the US Air Force, the Roothbert Fund, and the Texas Association of Black Personnel in Higher Education for their support throughout the work.

Abbreviations: CFD, computational fluid dynamics; **FD**, fluid dynamics; **DSD/SST**, deforming spatial domain/stabilized space-time formulation; **FOI**, fluid object interactions; **FOIST**, fluid object interaction subcomputation technique.

REFERENCES

Aliabadi S, Tezduyar TE (1993). Space-time finite element computation of compressible flows involving moving boundaries and interfaces. Comput. Methods. Appl. Mech.Eng.107: 209-224.

Beetstra R, van der Hoef MA, Kuipers JAM (2007). Numerical study of segregation using a new drag force correlation for polydisperse systems derived from lattice-boltzmann simulations. Chem. Eng. Sci. 62: 246-255.

Behr M (1992). Stabilized finite element methods for incompressible flows with emphasis on moving boundaries and interfaces. PhD dissertation, Department of Aerospace Engineering and Mechanics, University of Minnesota, Minneapolis, Minnesota.

Behr M, Tezduyar TE (1994). Finite element solution strategies for large-scale flow simulations. Comput. Methods. Appl. Mech. Eng.112: 3-24.

Belytschko T, Kennedy JM (1978). Quasi-eulerian finite element formulation for fluid-structure interaction. Proceedings of Joint ASME/CSME Pressure Vessels and Piping Conference, number 78-PVP-60 in ASME paper, New York, 13.

Belytschko T, Liu WK (1985). Computer methods for transient fluid-structure analysis of nuclear reactors. J. Nucl. Saf. 26: 14-31.

Donea J, Fasoli-Stella P, Giuliani S (1977). Lagrangian and eulerian finite element techniques for techniques for transient fluid-structure interaction problems. Trans. 4th International Conference on Structural Mechanics in Reactor Technology, San Francisco, B1/2.

Franck RM, Lazarus RB (1964). Mixed eulerian-lagrangian method. Methods in Computational Physics, 3: Fundamental methods in Hydrodynamics pp. 47-67.

Gerdes K, Matache AM, Schwab C (1991). Analysis of membrane locking for *hp* fem for a cylindrical shell. Seminar for Applied Mathematics. ETH Zurich, Zurich, Switzerland.

Hirt CW, Amsden AA, Cook JL (1974). An arbitrary lagrangian-eulerian computing method for all flow speeds. J. Comput. Phys.14: 227-253.

Hughes TJR, Liu WK, Zimmermann TK (1981). Eulerian finite element formulation for incompressible viscous flows. Comput. Methods Appl. Mech. Eng. 29: 329-349.

Jensen HH, Smith H, Wolfle P, Nagai K, Bisgaard TM (1980). Boundary effects in fluid flow. Application to quantum liquids. J. Low. Temp. Phys. 41:473-519.

Johnson AA, Tezduyar TE (1994). Mesh update strategies in parallel finite element computations of flow problems with moving boundaries and interfaces. Comput. Methods. Appl. Mech. Eng. 119: 73-94.

Johnson AA (1995). Mesh generation and update strategies for parallel computation of flow problems with moving boundaries and interfaces. PhD dissertation, Department of Aerospace Engineering and Mechanics, University of Minnesota, Minneapolis, Minnesota.

Johnson AA, Tezduyar TE (1995). Mesh generation and update strategies for parallel computation of 3D flow problems. Computational Mechanics ’95, Proceedings of International Conference on Computational Engineering Science, Mauna, Lani, Hawaii.

Johnson AA, Tezduyar TE (1997). Parallel computation of incompressible flows with complex geometries. Int. J. Numer. Methods. Fluids. 24:1321-1340.

Korpus R (2005). Reynolds-averaged navier-stokes in an integrated design environment. Simposio Internacional de diseno y produccion de Yates de motor y vela, Madrid, Espana.

Le Beau GJ (1990). The finite element computation of compressible flows. Master’s dissertation, Department of Aerospace Engineering and Mechanics, University of Minnesota, Minneapolis, Minnesota.

Mittal S (1992). Stabilized space-time finite element formulations for unsteady incompressible flows involving fluid-body interactions. PhD dissertation, Department of Aerospace Engineering and Mechanics, University of Minnesota, Minneapolis, Minnesota.

Noack BR, Eckelmann H (1991). A low-dimensional galerkin method for the three-dimensional flow around a circular cylinder. Phys. Fluids. 6:124-143.

Noh WF (1964). Cel:a time-dependent two-space dimensional coupled eulerian-lagrangian code. Methods.Comput. Phys. 3:117-179.

Schlichting H (1979). Boundary-Layer Theory. McGraw-Hill, New York, 7th edition.

Smagorinsky J (1963). General circulation experiments with the primitive equations. Mon. Weather. Rev. 91(3): 99-165.

Soinne E (2000). Aerodynamic and flight dynamic simulations of aileron characteristics. Department of Aeronautics, Royal Institute of -, Report 2000-12, 70 p + 6 app.

Tezduyar TE (1991). Stabilized finite element formulations for incompressible flow computations. Adv. Appl. Mech. 28:1-44.

Tezduyar TE, Behr M, Liou J (1992a). A new strategy for finite element computations involving moving boundaries and interfaces-the deforming-spatial-domain/space-time procedure: I. The concept and the preliminary tests. Comput. Methods. Appl. Mech. Eng. 94(3):339-351.

Tezduyar TE, Behr M, Mittal S, Liou J (1992b). A new strategy for finite element computations involving moving boundaries and interfaces-the deforming-spatial-domain/space-time procedure: II. Computation of free-surface flows, two-liquid flows, and flows with drifting culinders. Comput. Methods. Appl. Mech. Eng. 94(3): 339-351.

Trulio JG (1966). Theory and structure of the afton codes. Technical Report AFWL-TR-66-19, Air Force Weapons Laboratory: Kirtland Air Force Base.

Udoewa V (2005). Computational Techniques for Aerodynamic Interactions between Multiple Objects Emphasizing Paratrooper-Aircraft Separation. PhD dissertation, Department of Mechanical Engineering and Materials Science, Rice University, Houston, TX.

Udoewa V (2009). FOIST: Fluid Object Interaction Subcomputation Technique. Communications in Numerical Methods in Engineering, DOI: 10.1002/cnm:1219.

Wooley NH, Hatton AP (1973). Viscous flow in radial turbomachine blade passages. Conference on Heat and Fluid Flow in Steam and Gas Turbine Plant, Coventry, England, United Kingdom.