IAR STORES CLEARANCE CFD APPROACH: FROM DEVELOPMENT TO AUTOMATED ENGINEERING TOOL

F. Fortin, A. Benmeddour and A. Tahi
Institute for Aerospace Research (IAR), National Research Council (NRC)
e-mail: francois.fortin@nrc.ca

Keywords: *store release, automation*

Abstract

*The IAR Quasi-Steady CFD approach for store release prediction is presented. First, the various modules of the approach are described. They consist of an inviscid flow solver module **FJ3SOLV**, a **6-DOF SSM** module and a mesh motion module (**MESH MOTION**). In order to produce an engineering tool, the approach, which required human intervention during the computation of the store trajectory, has now been automated. This was carried out by creating a GUI (Graphic User Interface) that allows the user to define interactively the various loadings acting on the releasing aircraft in the carriage position. A fully automatic procedure of the global remeshing of the computational domain, using ICEM CFD mesh generation software, was also implemented. Some results obtained in CFD challenges are presented. They compared well with those obtained by other countries.*

1 Introduction

Accurate predictions of the aerodynamic loads on a store released from an aircraft and the prediction of its trajectory are of primary importance in the store certification process. The accuracy is more critical during the first ½ second of the store release when the store is still near the carrier aircraft. These predictions were and still are very hard to carry out for most real configurations. Their evaluations are becoming difficult, as the aircraft geometry becomes more complex and the speed envelope of the store release is increased. The problem becomes more severe and more challenging, due to the complex nature of the aircraft flow field and the strong aerodynamic interference between the store and the aircraft. The aerodynamic interference can even cause the store to strike the parent aircraft if it is not released in a safe manner.

The two main traditional sources of data used in engineering analyses of store separation characteristics are ground and flight tests. However, obtaining these data is very expensive and time-consuming. To obtain sufficient carriage and trajectory data for a given store to be certified, 20 flight tests, plus one or two wind tunnel entries, are required. In the event of an improper trajectory, pylon and/or attachment points, modifications may have to be made, resulting in more flight and wind tunnel testing [1]. This process is required for each store/aircraft configuration.

The store carriage and separation community is now considering the inclusion of Computational Fluid Dynamics (CFD) in the process of store certification, to complement the wind tunnel tests and to reduce the requirements for flight tests of store separation events. This is due to time constraints, shrinking budgets, rising wind tunnel costs and advances in CFD [2].

To answer the needs of the Canadian Department of National Defence (DND), the Institute for Aerospace Research (IAR) at the National Research Council of Canada (NRC) undertook the initiative (in 1994) to develop a Canadian CFD capability for the numerical prediction of external stores carriage and

separation. This initiative has led to the development of an Euler Quasi-Steady approach that can be used to predict trajectories of stores released from an aircraft. The developed CFD approach has been validated through participation in international efforts organized to demonstrate the capability of CFD methods as an integral component to the overall store certification process [3-5].

In order to take into account the motion of the released store, IAR chose to use an unstructured moving mesh technique, in which the surrounding mesh is stretched to follow the store motion. It was coupled with other modules and could be run in batch mode. However, in the past, the user had to monitor the simulation at intervals to check whether the mesh motion technique had failed. In case of failure, the user had to move the store manually with ICEM CFD to its next position, generate a new mesh and restart the computational process.

Generating a mesh this way was subject to the user's experience and know-how, and could take anywhere between four to eight hours. If the mesh motion failure happened overnight, the trajectory calculation would not carry on until the next day. A whole night of CPU time was thus lost. Full automation of the IAR Quasi-Steady CFD approach, by adding the capability to carry out grid generation in batch mode, was thus crucial to eliminate the time-consuming tasks and non-desirable factors. This has been carried out, as explained in this paper.

Also, to ease the use of this approach for store release engineers who are not CFD or grid generation experts, a graphical user interface (GUI) has been developed, that allows the user to easily define the loading acting on the releasing aircraft, as well as the store release sequence, interactively. Once the loading is specified, a single batch file is launched which will compute the full store trajectory and the required new meshes, when required. The GUI will also be used as a post-processing tool to analyze the obtained results. It will display X-Y graphs, as well as a full animation of the store release process.

2 Quasi-Steady approach

The Quasi-Steady approach consists of three different modules:

a) a steady-state inviscid solver (**FJ3SOLV**),
b) a 6-DOF store separation model (**6-DOF SSM**),
c) a mesh motion module (**MESH MOTION**)

The approach has been implemented in a modular way, and each of the three modules can be used separately. They are briefly described in the next paragraphs.

2.1 Steady-state inviscid flow solver

The in-house unstructured inviscid flow solver, FJ3SOLV, is based on a finite volume formulation. The convective fluxes are computed using Jameson's cell-centered formulation with the standard explicit addition of second and fourth order artificial viscosity [6]. The steady-state solution is obtained using an explicit 4-stage scheme. Standard acceleration techniques, such as local time stepping, implicit residual smoothing and enthalpy damping, are used to speed up the convergence of the scheme. With all these various techniques, a Courant number (CFL) of about 5 can be used.

On the solid walls, a slip condition is applied with the normal velocity set to zero, while at the far field, a characteristic approach using the Riemann invariants is used.

At the engine inlets of the releasing aircraft, the engine mass-flow-rate ratio MFR is used to impose an average normal velocity V_n through the engine inlet face as:

$$\rho V_n = \mathrm{MFR}\ (\rho V)_\infty \qquad (1)$$

Using the characteristic theory, all the flow variables can then be evaluated by adequate extrapolation from the inside or outside of the computational domain [7]. The engine outlets are kept closed in the computations, as their influence on the solution is probably negligible when the store is released from the aircraft.

Various flags are used on the different boundaries of the bodies, allowing a breakdown of the contributions of the various components that comprise a store. The aerodynamic coefficients are output in global axes. The aerodynamic moments are evaluated, relative to the origin of the global axis system.

2.2 6-DOF store separation model

The 6-DOF store separation model used is a simplified version of a code developed by Bombardier [8]. The trajectory of the store is computed using the standard equations of motion, while the releasing aircraft is considered to be a single rigid point, without elastic deformation of any part of its structure.

It is assumed that the releasing plane is continuing a steady flight (level, climbing or diving) even after the store release. For a relatively heavy store, the aircraft would accelerate if its speed and angle of attack were unchanged, due to the weight change.

The ejector model was initially developed by Bombardier [8]. The ejector forces are imparted to the store, by assuming that there is a single point of application and permanent contact between the ejectors and the store.

Once the points of application have been estimated, the actual stroke of each ejector can be computed. A given ejector force, relative to the piston of the ejector, corresponds to this stroke. When the stroke of one of the ejectors is greater than the maximum ejector stroke length, the applied forces and moments due to this specific ejector are set equal to zero.

In the Quasi-Steady approach, the aerodynamic coefficients are known only at time n; thus, the trajectory is advanced by only one time step Δt. The aerodynamic coefficients must be re-evaluated for the new store position. Following computations on a similar test case [5], it was found that a time step of 0.02 sec. was sufficient to provide good engineering accuracy.

The store aerodynamic coefficients are computed using the steady-state Euler solver **FJ3SOLV** and provided as input to the **6-DOF SSM** code. The moment coefficients are first computed at the store gravity center (CG). Using a rotation matrix obtained from the store Euler angles, all the coefficients are then evaluated in the store reference axes. Using the store identifying flags, the resulting coefficients on the store are then summed up and a final renormalization with the adequate reference surface and length is performed. These final resulting loads (forces and moments) provide the right-hand side of the equations of motion, which are integrated for one time step Δt. As the approach used is quasi-steady, the effect of the aerodynamic damping coefficients (Clp, Cmq and Cnr), that appear due to the angular rates of the store, is added to the total moments using their estimated values.

2.3 Mesh motion module

When moving the store, the mesh should be moved with the store, while maintaining a constant connectivity. An initial mesh and the new position of the store are thus provided as input. The mesh motion is performed by assuming that the nodes are connected by a series of springs. The goal is to minimize the potential energy of the system, which is equivalent to solving:

$$\sum_{j=1,n} K_{ij} \ (x_i - x_j) = 0 \qquad (2)$$

where the summation is carried out on the n edges joining node i to the surrounding j nodes. K_{ij} is the spring stiffness of the edge connecting node i to node j, defined as inversely proportional to the length of the edge. For small edges, the degeneration of the mesh is postponed.

However, this formula has the undesirable feature of not retaining the initial mesh if there are no displacements. This can be alleviated by adding a source term, giving:

$$\sum_{j=1,n} K_{ij} \ (\Delta x_i - \Delta x_j) = 0 \qquad (3)$$

where the unknowns are now the node displacements. The linear system is solved using a Jacobi method with under-relaxation. This

approach is not failure proof, due to the overlapping of edges, especially for non-convex domains.

To minimize the computations required for the mesh motion and to allow larger displacements before overlapping, windows are used, as suggested by Singh [9]. A window can be created around each component by specifying a Euclidian distance. A search is then made to find the nodes that are inside the window. A rigid motion is imposed for these nodes, specified by the component to which they are related. This keeps the mesh in the window undistorted (Figure 1). An important feature to take into account is that if two windows related to different moving bodies intersect, then the nodes in the overlapped region are allowed to move, except if they are on the surface.

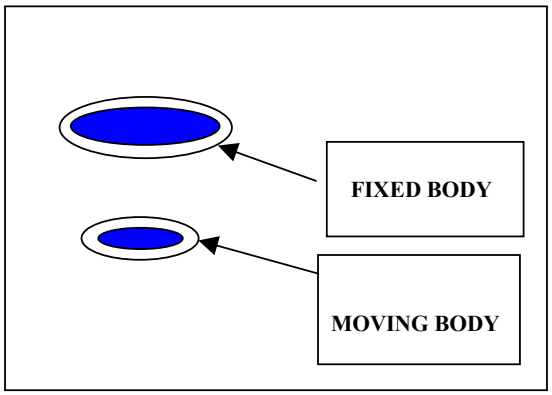

Figure 1 Window concept

When the mesh motion fails, a global remeshing has to be performed. This requires user intervention and thus slows down the process.

Work is ongoing in order to implement a series of local remeshing techniques [10-11] that would correct the degenerated cells and, at the same time, improve the mesh quality. The local remeshing would minimize the amount of numerical diffusion introduced by interpolating the solution into the new mesh. It would then be possible to make fully unsteady computations, as in [12].

2.4 Quasi-Steady approach

The Quasi-Steady approach used previously can thus be summarized in the following steps:

A) Compute, for a given store position, the steady inviscid flow-field solution with **FJ3SOLV**. Output the store aerodynamic coefficients in global axes.
B) Provide as input the store aerodynamic forces and moments to the **6-DOF SSM** code. Transfer them in store axes. Compute the new store CG position and orientation (Euler angles) for one time step Δt.
C) Move the store to this new position. Using windows around the moving store, move all the mesh nodes that are inside these windows, as a rigid body. Generate mesh motion on the other parts of the mesh using the spring analogy (**MESH MOTION**). Perform global remeshing, only if necessary, by user intervention. This stops the process, which must be restarted by the user.
D) If the new store position is far from the aircraft or if the store hits the aircraft (target), stop. Otherwise, go back to A using the previous solution as the initial estimate.

3 Towards automation

The approach must now be implemented as an engineering tool that can be used by store release engineers. Previously, all the inputs for the various modules had to be provided in ASCII files, which required a user's manual for successful execution. The process of mesh motion was also prone to failures, which then required human intervention to continue the process. It was not always obvious what parameters to set in order to restart the Quasi-Steady approach correctly. Ideally, the store release engineers should be able to use this approach, without having the burden of becoming experts in mesh generation or CFD.

To fulfill these needs, it was necessary to implement two additional automated procedures. The first one consists of a GUI that will ease the interactive creation of the appropriate input files for script procedures. The

second one is a full automation of the mesh generation process for any store position. In both of these procedures, the concept of fixed and moving objects is crucial. A fixed object is one that will be fixed relative to the releasing aircraft; a moving object is one that will be released from the aircraft. In the context of this paper, the moving object is represented by the store. This procedure, however, can be extended to handle a series of objects that are moving relative to each other (rotor-stator). The motions of these objects could be imposed, without requiring the **6-DOF SSM.** The **6-DOF SSM** module could be replaced also by a structural code that would compute the resulting displacements for a resultant loading on the structure. This would permit computations of aero-elastic problems.

The resulting scripts are intended to be executed on a large mainframe. It appears, however, that in future, clusters of high-end PC s will be more cost-effective.

3.1 GUI description

The platform chosen is a high-end PC, running the Linux environment. The programming language is Fortran 90. The various menus and dialogs have been implemented using the Winteracter software, which allows the use of API OpenGL programming.

The GUI consists of a series of menus that allows the user to prescribe the various loadings on the aircraft, as well as their release sequence in an interactive manner. Due to space limitation, only the basic dialogs for object manipulation will be provided. In the following, the options available in the dialogs, as well as the dialog names, will be highlighted in bold.

Figure 2 represents the main windows used when the GUI is started. The top window represents the various menus available. The middle window is an **OpenGL Graphics Window**, where the various 3D objects, as well as the releasing aircraft, will be drawn. The bottom window is a message window that describes the various processes being performed. **Geometry** is a sub-menu that can open either an **Aircraft** or a **Generic Object** environment.

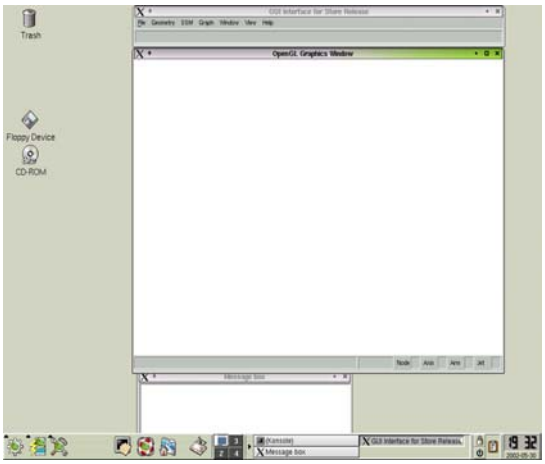

Figure 2 Main windows of the GUI

The **Aircraft** environment consists of two dialog windows. The first dialog, **List of objects (**Figure 3**)**, represents a database of various objects that can be used to build a new aircraft loading. It consists of different kinds of stores, pylons and fuel tanks that can be carried by the releasing aircraft. A short text **Description** is available to describe the object. It is possible to **View** any of these objects in another window. It is also possible to **Create** new **Object**s or **Aircraft** to add to the database.

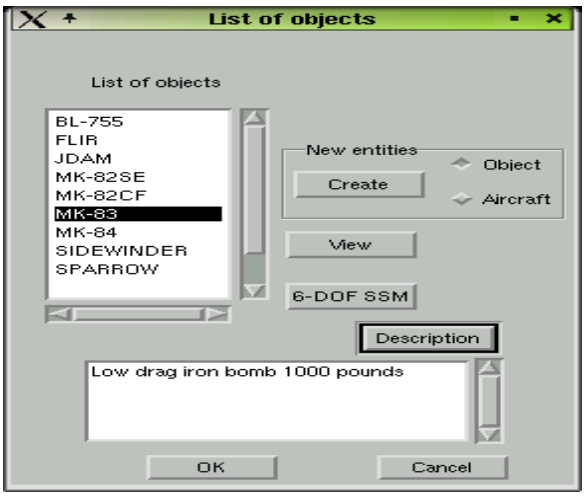

Figure 3 **List of Objects** dialog

It is also possible to scan through the characteristics of the objects for the **6-DOF SSM** (characteristic length **Diameter** and reference area **Sref** and **Mass** and **Inertia**, as well as their damping derivatives **Clp**, **Cmq** and **Cnr**) through a dialog (**Store characteristics**, Figure 4).

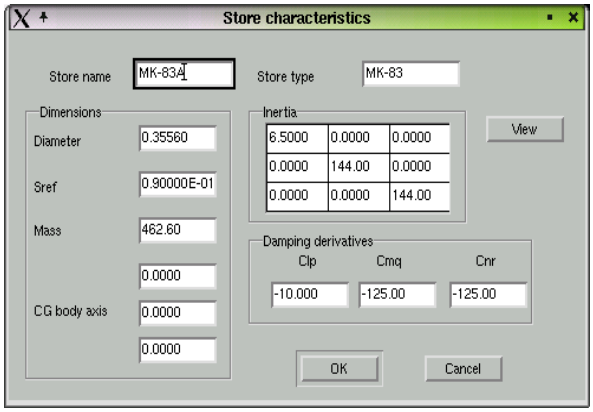

Figure **4 Store Characteristics** dialog

The second dialog represents the **Aircraft Loading** on the aircraft (Figure 5). At the beginning, no loading is defined. It is possible to reload an existing loading (**Specified Loading**) that defines all the parameters of the dialog. Elsewhere, the loading must be specified. The **Aircraft Type** can be selected, as well as the **Configuration** of the aircraft (full aircraft, port side or starboard side). Through the mouse, the user can sequentially select the objects that must be **Add**ed on the various **Station**s of the aircraft from the **List of objects**. Initially, the number of stores that can be added to each station is fixed at 4. It can be increased, however, by using the push-button **Update station**. As soon as an object is added, a faceted representation of the object is immediately drawn in the **OpenGL Graphics Window**. An object can be seen either in **Solid** or in **Wire frame** mode. A push-button (**Fix**) allows defining if the added object is a fixed or moving object. A S**ummary** of the number of fixed or moving objects, as well as the number of stores added to each station in **Station info**, is also available. The object characteristics can be modified for the **6-DOF SSM**, which brings up the **Store Characteristics** dialog, as well as the **Position** that it is occupying. The resulting loading can be **Save**d for later work. A **Delete** option allows some objects to be eliminated that are not present in the actual loading. This would happen if a **Specified loading** has been read that contains some objects that must be eliminated or replaced.

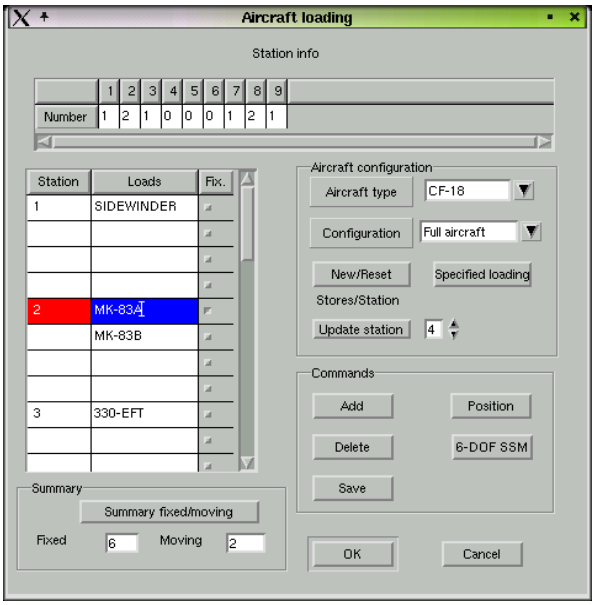

Figure 5 **Aircraft Loading** dialog

A surface mesh is included in the database of each object. When the object is moved to a new position and orientation by **Position**, the basic transformation matrices are then applied to provide the new coordinate values that correspond to the new position. The surface mesh is then redrawn at this new position in the **OpenGL Graphics Window**.

The releasing flight conditions are defined through the dialog **Flight Conditions** (Figure 6). In this dialog, the freestream flow conditions **(Mach number**, **Angle of attack** and **Sideslip angle)** can be defined in **Freestream**. The releasing modes are given in **Flight Mode**: the **Dive angle**, **Altitude**, and the **Load factor**, as well as the **Flight** regime (Straight and Level flight, Constant G Pull-up, Recorded flight-path and Banked turn).

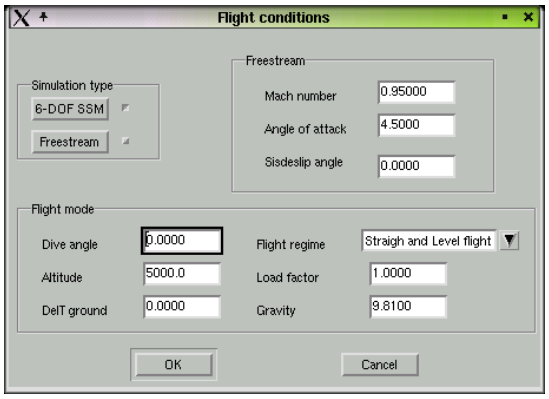

Figure 6 **Flight Conditions** dialog

For a **Generic object** case, two dialogs are present: **List of objects** and **Generic Object (**Figure 7**)**. For **Generic Object**, two columns of objects are available: one for **Fixed objects** and the other for **Moving objects**. The various options available are the same as for **Aircraft Loading**.

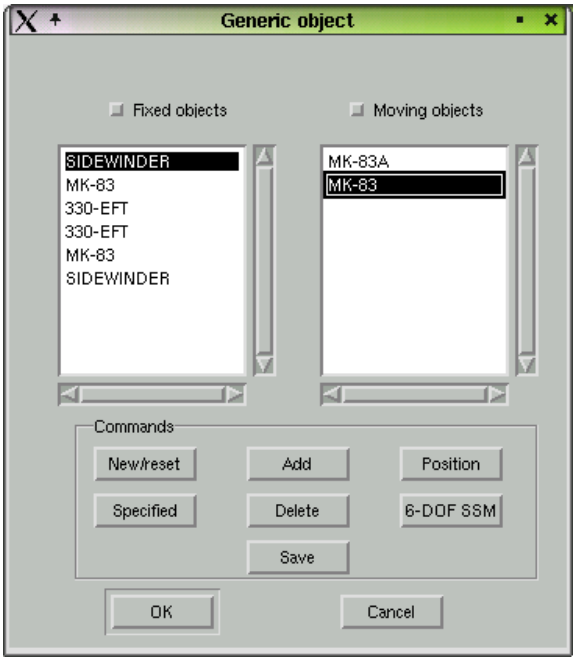

Figure 7 **Generic Object** dialog

Once all the objects have been loaded, the independent files representing them are concatenated into a single file that represents a specific loading. This loading will be **Saved** again in a database and could be used as a starting point for a new **Specified loading**. The loading file is included in a script file that is run with the mesh generation package ICEM CFD to get an initial mesh with the stores in carriage position. The concept of this technique has been presented in reference [13] and is discussed in the next paragraph.

3.2 Remeshing automation

The remeshing automation consists of integrating the mesh generation process with the trajectory calculation modules and running the whole procedure in batch mode. User intervention is required only at the end of the computation to post-process and analyze the data. The automatic procedure is summarized in the flow chart below.

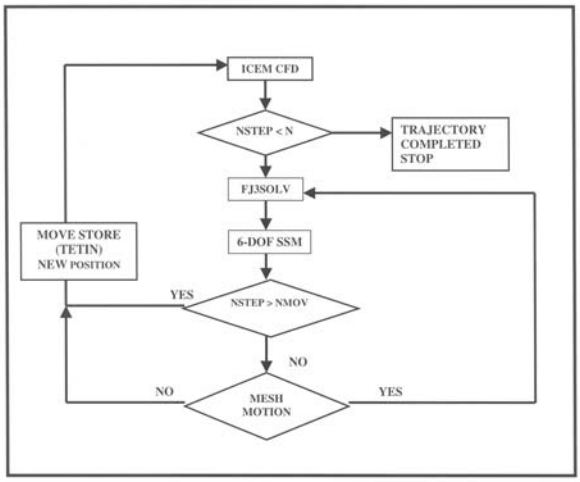

Figure 8 Flowchart

The automation was made possible by the fact that ICEM CFD can be run in batch mode (scripting) to generate unstructured meshes, based on modular geometry input files. To generate a tetrahedral mesh, ICEM Tetra mesher requires a geometric TETIN (TETra INput) input file. This file contains the geometric representation (b-spline curves, b-spline surfaces and prescribed points, if any), as well as all the mesh settings. The TETIN file is obtained using the DDN mesher interface (DDN-TETIN), where DDN is the CAD

software bundled with the ICEM CFD mesher. The TETIN file can also be generated using the appropriate ICEM-CAD interface if the geometry is built using CAD software, other than DDN.

Any aircraft/stores configuration can be broken down into different components, and each component can have its own TETIN file. The individual TETIN files can then be merged automatically in ICEM mesher, and a mesh over the full configuration can be generated. One of the advantages of this breakdown of the aircraft/stores configuration is that different transformations, translations and rotations, for instance, can be performed automatically on the store (moving object) without affecting the rest of the configuration (fixed object). The automatic manipulation of the store geometry (TETIN file) in isolation is an important feature of the automated process.

For any store carriage and release simulation, the automatic procedure starts with the generation of an initial mesh with the store in the carriage position. This is performed using the loading file generated by the GUI. The solver **FJ3SOLV** is then run to obtain the carriage loads. Once the carriage loads are computed, the next point (store CG displacements and orientations) of the store trajectory is determined for a small time increment, using the **6-DOF SSM**. If mesh motion is possible, the store and the surrounding mesh points are moved to the next store position, using the spring analogy technique. However, if the **MESH MOTION** module fails, the store is automatically moved to its new position, using the store's TETIN file, and ICEM Tetra mesher is called again to merge all the TETIN files and generate a new mesh for the whole aircraft/stores configuration. The flow solver **FJ3SOLV** and the **6-DOF SSM** are then run again, and the automatic process is continued for a certain number of time steps necessary for the computation of the whole trajectory.

As the procedure lacks an interpolation module to transfer solutions between the different meshes, the mesh motion is not performed for more than three time steps (NMOV), whenever possible. After three time steps, the mesh is considered to be inadequate (bad quality) for the calculations, and the store is automatically moved as if the mesh motion technique had failed. The computations then continue until the end of the trajectory simulation. An interpolation module is in development, as well as some built-in logic to better control the mesh generation process.

4 Results

Some results, previously obtained from international programs, are now demonstrated. The first validation exercise was carried out within an international TTCP program (The Technical Cooperation Program) on CFD techniques for store carriage and release on a generic Wing/Pylon/Finned-store (W-P/FS) [3]. The second validation was for a more realistic configuration, the CF-18/JDAM-store (Joint Direct Attack Munition) [4].

The W-P/FS is a generic test case, made using a clipped delta wing with a 45^{o} leading edge sweep carrying a generic finned-store with an ogive-cylinder-ogive body shape with four fins. The freestream conditions are given in Table 1. Figure 9a shows the Mach number distribution on the lower surface. Figures 10a and 11a give a comparison between the computed store positions and orientations (Euler angles), respectively, versus the experimental values.

The second validation case consists of a CF-18 aircraft with two pylons, a 330 US gallon External Fuel Tank (330 EFT) mounted on the inboard pylon, a JDAM store mounted on the outboard pylon and an equivalent body of revolution, approximating both the wing tip launcher and missile (AIM-9). The flight conditions of the releasing aircraft are again given in Table 1. Figure 9b shows the Mach number distribution on the CF-18 in the carriage position. Figure 10b and 11b give the comparison between the computed store positions and orientations, relative to the store in the carriage position, versus the experimental values.

The approach succeeded in capturing all of the major trajectory trends of both stores well, with results similar to those obtained by other countries.

	W-P/FS	CF-18/JDAM
Mach	0.95	0.962
AOA (deg)	0.0	0.46
Dive angle(deg)	0.0	43
Pressure altitude	26000.0	6332.0

Table 1 Release flow conditions.

Conclusions

A fully automated approach for store release predictions has been developed at IAR that does not require user intervention. Store engineers will now be able to concentrate fully on the physics of the stores clearance.

The results obtained using this approach compared very well to the prediction of other countries. Improvements are being implemented continuously to expand its capabilities.

IAR has been successful in reducing the CPU time required to get a solution. There is no more time lost during the store trajectory computations. The final objective, however, is to obtain a computed trajectory within a day or two. Work is now being performed to implement implicit GMRES matrix-free techniques into the code to improve its speed. The objective is to achieve a speed gain on CPU time of about 5. Investigations are also ongoing into the implementation of parallelisation, using the MPI library.

Acknowledgments

The authors would like to express their gratitude to the Canadian Department of National Defence (DND) for their financial support.

They would also like to thank Mr. Po-yang Jay Liu who, as a summer student, tested the basic graphical concepts of the GUI interface.

References

[1] Fox J H, Donogan T L and Nichols R H. Computed Euler Flow for a Transonic Aircraft with Store. *Journal of Aircraft*, Vol. 28, pp. 389-396, 1991.

[2] Cenko A and Lutton M. A CFD application to Store Separation–Status Report. *The Aeronautical Journal, Vol. 104,* pp. 459-466, October 2000.

[3] Benmeddour A, Fortin F and Jones D J. 3-D Euler Computations of the Flow About a Generic Wing/Pylon/Finned-Store Configuration Using Unstructured Tetrahedral Grids. *Proceedings of the 6th Annual Conference of the CFD Society of Canada*, Quebec, Canada, June 7-9, 1998.

[4] Fortin F, Benmeddour A and Jones D J. Application of the Canadian Code to the F/A-18C JDAM Separation. *AIAA Paper 99-0127*, 37th AIAA Aerospace Sciences Meeting & Exhibit, Reno, NV, USA, Jan 11-14, 1999.

[5] Benmeddour A, Fortin F and Jones D J. Validation of a Quasi-Steady Euler Approach for Store Separation Using Unstructured Tetrahedral Meshes. *Proceedings of the CASI 7th Aerodynamics Symposium*, Montreal, QC, Canada, May 2-5, 1999.

[6] Jameson A, Schmidt W and Turkel E. Numerical Solution of the Euler Equations by Finite Volume Methods Using Runge-Kutta Stepping Schemes. *AIAA 81-1259*.

[7] Fortin F, Sun Y, Benmeddour A and Jones D J. Unstructured Euler Solution for the CF-18 Aircraft. *Proceedings of the CASI 6th Aerodynamics Symposium*, Toronto, ON, Canada, April 28-30, 1997.

[8] Canadair SSM code, RAU-261-158, August 1995.

[9] Singh K P, Newman J C and Baysal, O. Dynamic Unstructured Method for Flows past Multiple Objects in Relative Motion. *AIAA Journal*, Vol. 33, No. 4, April 1995.

[10] Turcotte J, Dompierre J, Labbe P, Trepanier J Y, Fortin F, Benmeddour A. A Study of Repair Processes of Moving Meshes for Store Release Prediction. *CEAS Aerospace Aerodynamics Research Conference,* Cambridge, United Kingdom, 10-13 June 2002.

[11] Fortin F, Dompierre J, Labbe P and Trepanier J Y. Local Remeshing techniques around geometries for store release. *Proceedings of the 47th Annual Conference of the Canadian Aeronautics and Space Institute*, Ottawa, Ontario, Canada, pp 57-66, May 2000.

[12] Fortin F. Unsteady Euler Techniques for Moving Body Problems: A Comparative Study. *Proceedings of the CASI 48th Annual Conference,* Toronto, Ontario, Canada, April 29 May 2, 2001.

[13] Tahi A, Fortin F and Benmeddour A. Modular and Automated Unstructured Grid Generation Using ICEM CFD. *Proceedings of the CASI 48th Aerodynamics Symposium*, Toronto, Ontario, Canada, April 29-May 2, 2001.

a) W-P/FS

b) CF-18/JDAM-Store

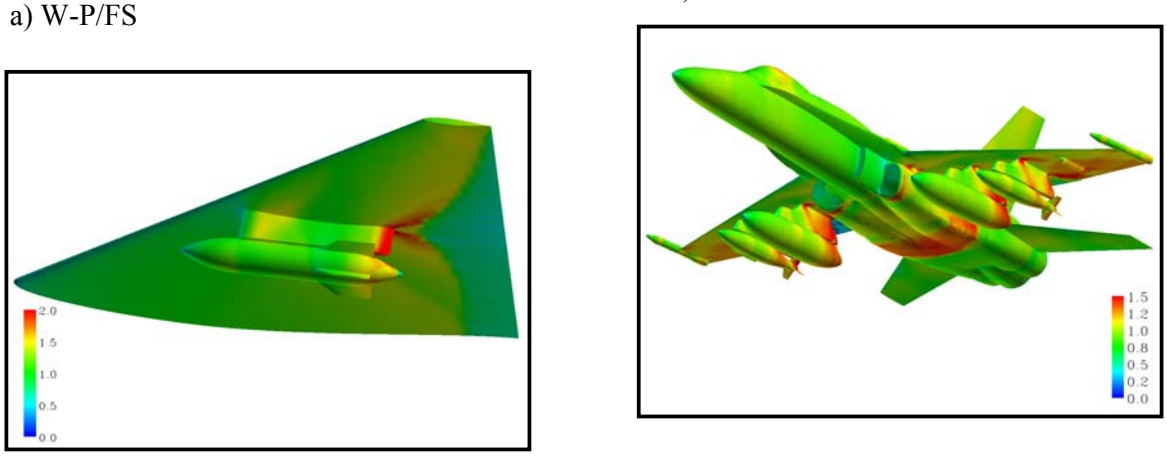

Figure 9 Mach number distribution

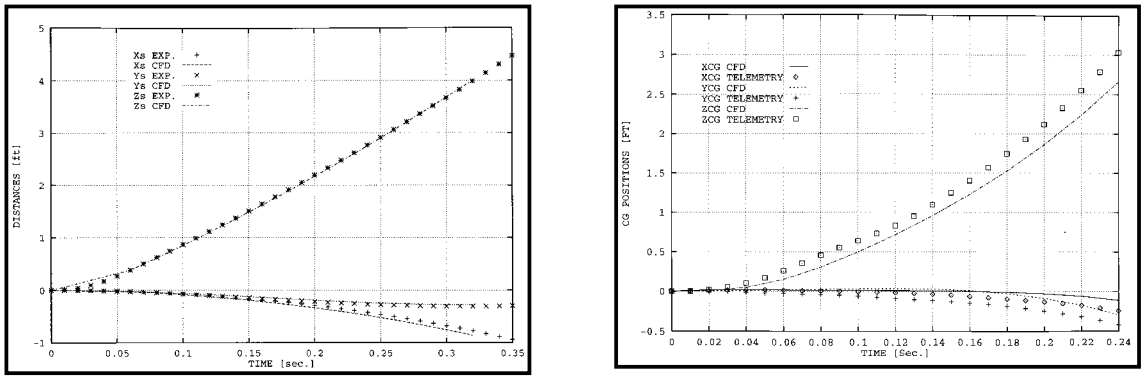

Figure 10 Store CG displacements

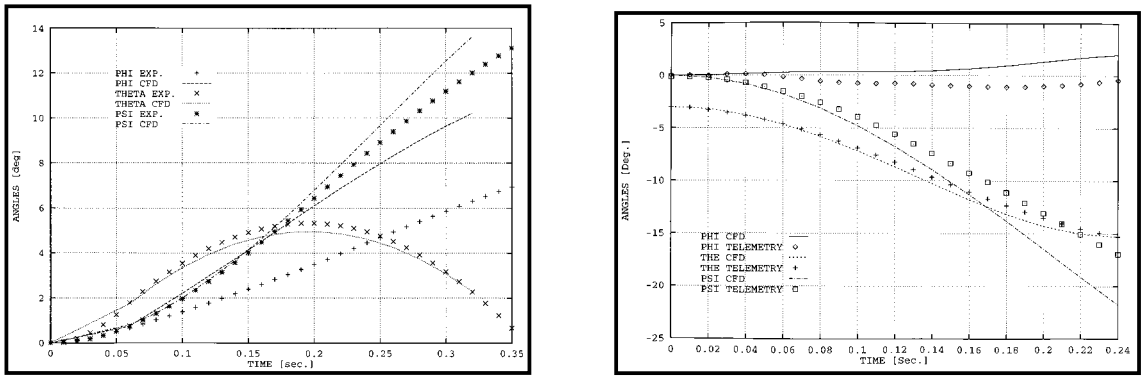

Figure 11 Store CG attitudes