FEATURE–ORIENTED GRID TOPOLOGY DESIGN FOR AEROSPACE CONFIGURATIONS

Y. Xia[a], J. Häuser[a], M. K. Patel[b], M. Cross[b]

[a]Department of High Performance Computing
Center of Logistics and Expert Systems
Salzgitter, Deutschland

[c]Centre for Numerical Modelling and Process Analysis
School of Computing and Mathematical Sciences
The University of Greenwich, London, UK

ABSTRACT The last few years have seen a substantial increase in the geometric complexity for 3D flow simulation. In this paper we describe the challenges in generating computational grids for 3D aerospace configurations and demonstrate the progress made to eventually achieve a push button technology from CAD to visualized flow. Special emphasis is given to the interfacing from the grid generator to the flow solver by semi–automatic generation of boundary conditions during the grid generation process. In this regard, once a grid has been generated, push button technology for most commercial flow solvers has been achieved. This will be demonstrated by the ad hoc simulation for the Hopper configuration.

1 GRAND CHALLENGE IN AERO–SPACE: A DIRECT PATH FROM CAD TO VISUALIZED FLOW

One of the grand challenges in aerospace computer simulation is easily stated: Once the CAD data has been generated, the design engineer wishes to see the flow around the configuration, without any further ado. Unfortunately, at present, there are several stages necessary to reach the final step, namely the visualization of the flow and its corresponding phenomena. The CAD data has to be converted into a surface decription for the grid generator, the actual grid has to be generated, the input for the flow solver has to be prepared, and the actual flow solution, including automatic grid adaptation, has to be carried out. Each step is a major undertaking by itself and does not work without a human interaction in the loop. In this paper we will address the topics of hexahedra grid generation for complex geometries and the interfacing to the flow solver, reporting about some of the progress that has been made in automating these two tasks.

In generating a computational grid, the CAD surface description is converted into a triangulated discrete surface. The triangulated surface needs to have certain features, such as closedness and orientability, and must retain the salient geometric features, such as sharp edges, tangent planes, and surface curvature. An evaluation of the geometric accuracy of the triangulated surface is possible only if these features do exist. To capture flow phenomena such as expansion waves or shocks, accurate shape preservation is a necessary condition. However, the complexity of mesh generation is substantially increased for these requirements. For instance, these geometric features are needed to model the trailing edge of a wing, a spacecraft body flap, or the sharp nose of a supersonic missile. The resulting grid should be singularity free and optimized (see Figs. 5 to 7). Producing such a grid quality may have a major influence on both the accuracy and the convergence speed of numerical solutions. Speedup factors of 3 to 10 have been reported for optimized grids generated by our grid generation code, compared to grids generated by standard techniques.

2 GENERAL STRATEGY FOR GRID DESIGN

There is a strict separation between grid topology, describing the connectivity of blocks in computational space, and the actual 3D grid in physical space using the physical coordinates of the body contour. In this way, parametric grid studies can be performed, using a fully automatic grid generation process. Topology is described by a wireframe model, visualized in physical space, i.e., wireframe edges are curved to reflect physical shape. In computational space, each block is represented by a cube, comprising 8 vertices, 12 edges, and 6 faces. Blocks are connected through faces only. Block boundaries overlap and C^1 continuity is automatically achieved by iteration. The wireframe model is constructed interactively, subdividing a complex configuration into logical components in form of cubes (see Fig. 3). The construction of the wireframe is supported by projection, duplication and grouping operations that means, for example, building a simple wireframe structure, this structure can be duplicated and automatically connected to the existing structure. The resulting, more complex, structure can be grouped and duplicated, or projected onto a surface etc. to build an even more complex structure. Set operators working on the wireframe points within a group can be used to modify the wireframe structure. The nesting of groups, is not limited. An arbitrary number of groups can be generated to describing the engineering (logical) parts of a configuration. In this regard, an object–oriented topology design is achieved. Before a grid computation is started, a comprehensive check of the topological validity is performed. This algorithm is a fairly complex piece of software. The grid itself, i.e., both surface and volume grids are generated fully automatic [1],[2].

3 RETAINING THE SALIENT GEOMETRIC FEATURES OF AN AERO–SPACE CONFIGURATION

In general, geometric features of 3D surfaces are characterized by the type of continuity they exhibit and, if applicable, by their local surface curvature. We do not consider grids that are discontinuous, because we feel that the redistribution of the fluxes across block boundaries is too cumbersome a process in 3D. However, we allow for merged grids, i.e. a multi–to–one mapping of neighboring blocks. Grids are normally generated with a one–to–one block mapping. For vectorized solvers (admittedly a rarity in these days) there is a merge tool that substantially reduces the number of blocks. Also, as mentioned before, completely unstructured grids can be generated.

C^0–continuity. This type of geometric feature represents intersection curves or sharp edges, or singular points (see Fig. 4). For example, this type of continuity may be generated by a deflected body flap or at a nozzle exit, e.g. modeling Ariane 5.

C^1–, and C^2–continuity. In general, the grid should ensure the smoothness of grid lines, rendering block boundaries invisble. The result of a simulation must not depend on the block structure. We have observed that in cases of heat load computation a grid that preserves curvature is mandatory for accurate results, i.e. automatic clustering is necessary. In practice, this means that there must be no wiggles in the discrete gradients computed along the streamwise direction for the direction off the body.

We need to determine sharp edges, because in the grid generation process points have to be projected back onto the original surface. At sharp edges or singular points (for example, pointed nose of a missile, see also Fig. 7) the surface is not orientable and special action has to be taken to retain these features in the surface grid. If not, sharp edges may be rounded and singular points may be lost. In automotive industry, for instance, this loss is not tolerable. Also, as our experience has shown, in simulating the ascent of a launcher, shocks may not be dtected or an incorrect prediction of shock strength may occur if these geometric faetures are not present anymore.

In general, it is sufficient for the search algorithm to check for a deviation in C^1–continuity to identify these geometric features. A C^0 curve is generally modeled by a block boundary. In this regard, the geometry influnces the block topology. If a solver accepts multiblock grids, parallelization is straightforward. Groups of blocks are formed so that each group has the same computational load. Naturally, this only results in static load balancing, but if recursive bisection is used as a more sophisticated load balancing algorithm, good results have been obtained. In the case of unstructued grid, the same technique is used to domain decompose the grid into a set of subdomains of equal size (i.e. number of cells).

4 SIMPLIFYING GEOMETRY

Before a complex configuration can be gridded, some thought has to be given to the level of detail of geometric modeling. In grid generation, preservation of geometric accuracy and grid generation efficiency may be conflicting issues. Retaining all geometric details increases the topological complexity. It is often unnecessary to mesh components, such as screws, bolts, rivets etc. Simplifying an existing geometry may be a time consuming process, because the CAD data has to be modified. For example, a CAD surface description for the surface of car may contain 800,000 patches, necessitating substantial preprocessing before the grid generation process can begin.

Geometric components. For most cases of flow simulation, a simple rule has shown good results in practice. Taking the reference length of the body (or major component) as 1, all parts whose length is less than 10^{-4} of the reference length, are considered to be micro–components, and will be removed. In addition, those geometric components whose influence on the flow simulation is considered negligible, will also be removed. For instance, in some cases of very high angle of attack, the vertical fin can be removed if only aerodynamic moments are computed [3].

Geometry impact on flow phenomena. Flow phenomena, such as expansion waves, shocks, contact discontinuities, jets, boundary layers, or flow separation and reattachement are fundamentally influenced by geometric features. In order to simulate these geometry–dependent flow phenomena, grids of high quality and geometric accuracy are required. Therefore a special optimization procedure for grid orthogonality, both at surfaces and in 3D space, grid smoothness (aspect ratio of neighboring cells), and minimization of grid skewness is implemented.

5 FEATURE–BASED BLOCK TOPOLOGY BUILDING

In devising a three–dimensional block topology a few simple but practical rules have been established. A 3D topology can be considered as an extension of a 2D one, advancing the wireframe from either body surfaces or from outer bounding surfaces into the solution domain. Which surface to select for the advancing wireframe front, depends on the geometric complexity of a surface. For external flow, the body surface is normally chosen. Next, the body surface is assumed of comprising a set of sub–surfaces. On each sub–surface, a block topology is generated in a 2D manner. In the next step, the resulting two–dimensional block topology is extended into the third dimension. As mentioned in Section 2, this is a semi–automatic process only.

In general, a body surface can be described in closed form by a triangulated, quadrilateral or mixed mesh (see Fig. 1). This mesh consists of a set of patches, because only at patch boundaries non–orientability of the surface is allowed. Thus, sharp edges etc. are preserved in the discrete surface description. It should be noted that the surface block topology in general is different from the patch topology of the surface description. However, block boundaries are required for sharp edges etc.

In addition, a configuration may comprise a set of mechanical components, such as fuselage, wing, and vertical fin. Regarding these engineering entities, the complete wireframe model is seen as an assembly of local wireframe models for these individual components. Identifying these components, local wireframe models can be generated at a much lower level of topological complexity.

6 AUTOMATIC INTERFACING OF GRIDS TO FLOW SOLVERS

Once a grid has been generated, push button technology for the solver is required. Grid output is either in form of a structured multiblock grid in NASA Plot3D format, or an unstructured hexahedra grid that directly fits into a solver like StarCD. In addition, the input of most commercial flow solvers can be generated. In other words, the output of the grid generator has to be the input of the flow solver. In the flow solver, only the free stream parameters and the CFD numbers should be set. In modern flow solvers an agent (sometimes called wizard) can be set up to simplify this task even further. The most important issue therefore is the provision of the proper boundary conditions for each patch of all of the bounding surfaces.

First, the grid generator has to have the capability to generate the output format needed by the flow solver. In our grid generator, formats for most commercial solvers are available along with the NASA Plot3D format. In addition, it is absolutely necessary that all boundary conditions for the flow solver are set during the grid generation stage. Surface grids may comprise several hundred patches, and, since each block has its own local coordinate system, the orientation of these blocks with respect to each other is unknown to the user. In the next section we will present the interfacing of the grid and flow solver, not only to our solver ParNSS (block structuerd grid), but also to the commercial flow solver CFD++ from Metacomp Technoloy, which requires an unstructured grid (see Fig. 6).

The setting of the boundary conditions is done interactively in the graphics user interface of the grid generator and takes only minutes for even the most complex grids, comprising several thousands of blocks. The user identifies one patch one a surface and assigns the proper boundary condition, e.g. wall. Each patch is assigned a center in form of a square. By graphically grouping those squares that have the same boundary conditions, this process is completed quickly. Additional set operators for forming groups of squares and activating or deactivating groups simplify this task further. Patches having the same boundary conditions have the same color and thus the result of setting the boundary conditions is easily visualized and corrected if necessary.

In this regard, a push button technology from grid to solver has been achieved. There is absolutely no interaction to prepare the input of the grid generator to the solver. This will be demonstrated by computing a grid for the German Hopper configuration, interfacing it to the CFD solver CFD++ and computing an Euler solution. The results are shown in the next section (see Fig.6).

7 3D EXAMPLES

In this section we show three examples of 3D configurations, demonstrating the process from the IGES data to the visualized flow solution. Most time consuming is the surface repair, needed to convert the IGES surface data into the proper triangulated or quadrilateral surface. Here, we are currently developing an IGES converter so that IGES data can be directly accepted by the grid generator. Once we have the discrete surface, see Hopper picture, Fig. 1, the topology is constructed using the GUI of the grid generator. With the topology in place, surface and volume grids are generated fully automatic including optimization, see Fig 2. In the next step, boundary conditions are set interactively. Finally, the format for the grid output is chosen. This output should serve as input for the flow solver, so that a Unix pipe can be used.

Hopper and EXTV configurations. Hopper is the German concept developed in the ASTRA program for a next generation launcher. Hopper is a configuration for a two stage to orbit vehicle (TSTO) where the first stage does not go into orbit but performs a suborbital hop only. A simialr concept, EXTV, was considered in the ESA FESTIP system study. The CAD data is in IGES format, consisiting of more than 200 entities. This data is converted to a quadrilateral mesh used for surface description, as shown in Fig. 1. Sharp edges after wing and vertical fin are considered as the main geometric features to be preserved. The complete vehicle is decomposed into three parts: fuselage, wing, and vertical fin. Their block topologies are built as separate objects. Sharp edges are generated by fixing grid points on the intersection curves of the additional geometric surfaces, called *internal surfaces.* Fig. 2 shows a coarse grid of the Hopper configuration. The grid consists of 908 blocks with some 200,000 hexahedra. The essential geometric features to be accurately gridded are marked by circles.

Generic missile. The requirement for this grid generation is that a singularity at the nose of the missile has to be avoided. In addition, the rotationally symmetric features must be accurately modeled. Using special control surfaces, the tip is fixed with a singularity free topology around it. This ensures the geometric accuracy of the missile nose. Fig. 7 shows the symmetry plane and surface grid of the missile.

8 CONCLUSIONS

In this paper we addressed the topics of feature oriented grid generation, i.e. generating grids for complex geometries with numerous components and the preservation of geometric features like sharp edges or singular points. A practical rule was stated for building 3D topologies. The automatic interfacing from grid generator to flow solver was outlined, namely the interactive setting of boundary conditions in the grid generator. Three 3D examples were shown to demonstrate the validity of the approach.

The grand challenge for aerospace simulation is still not solved. The direct usage of CAD data in the grid generator has not been achieved, a fully automatic topology building is not (yet) available, and fully solution adaptive grids are not available. Of course, it is straightforward to automate topology building by restricting the available topologies, and grid adaptation can also be implemented but at the cost of either randomly filling the space with additional cells or deteriorating the existing grid quality. The challenge is quality, i.e. optimzing topology. grid quality as well as grid density. All three topics have a major impact on computing time, solution accuracy, and convergence speed. This impact may be much more severe than any new or improved numerical algorithm as our computational experience has shown. The key to solving these problems lies in advanced algorithms and software engineering.

ACKNOWLEDGEMENT

This work has been funded by the Ministry of Science and Culture of the State of Lower Saxony, Germany, AGIP Project Nr. 1999.365.

NOTE

This paper presents part of the PhD thesis of the first author, pursued at the University of Greenwich, London, U.K. and the University of Applied Sciences, Braunschweig–Wolfenbüttel, Germany.

References

[1] J. Häuser, P. Eiseman, Y. Xia, Z.–M. Cheng: Parallel Multiblock Structure Grids, CRC Handbook of Grid Generation, CRC Press Inc., 1998.

[2] P. Eiseman, Z.–M. Cheng: GridPro/az3000, User's Guide and Reference Manuals, 2000. PDC, 300 Hamilton Ave Suite 409, White Plains, NY 10601.

[3] Y. Xia, J. Häuser, J. Muylaert, M. Spel, L. Walpot: A General Grid Generation Strategy for Complex Aerospace Geometries, Numerical Grid Generation in Computational Field Simulations, Proceedings of the 7th International Conference, September, 2000.

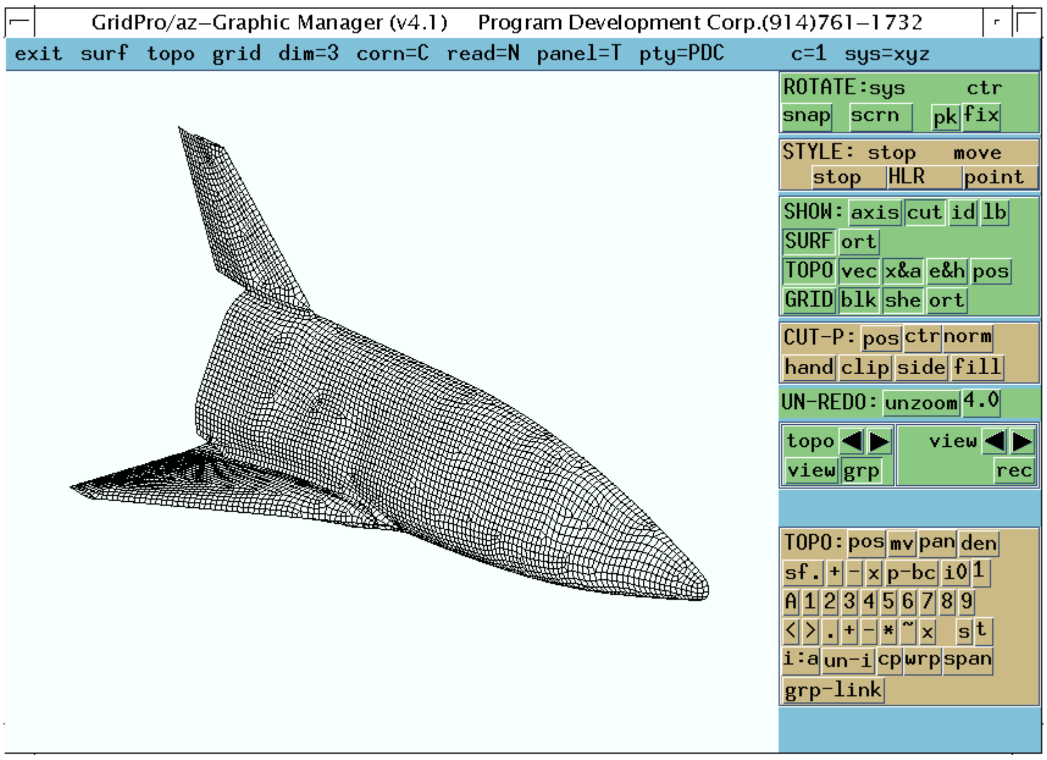


Figure 1: IGES geometry data can be converted into either triangular, quadraliteral or a mixed element surface description using the tool HyperMesh. The figure shows converted surface geometry data visualized by the grid generator GridPro.

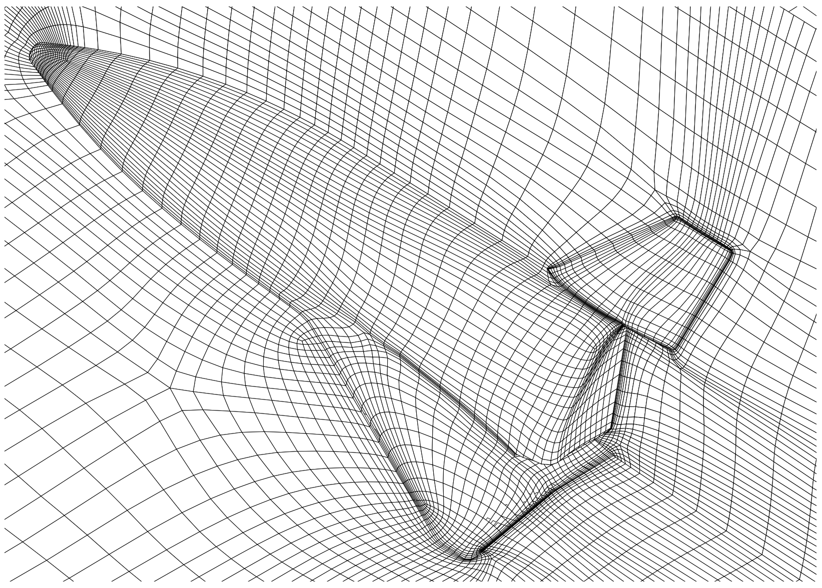


Figure 2: The figure shows the multiblock grid for the Hopper configuration. This grid consists of 908 blocks and some 200,000 grid points. Depicted are the symmetry plane and a horizontal plane to show both the grid line distribution as well as the block structure.

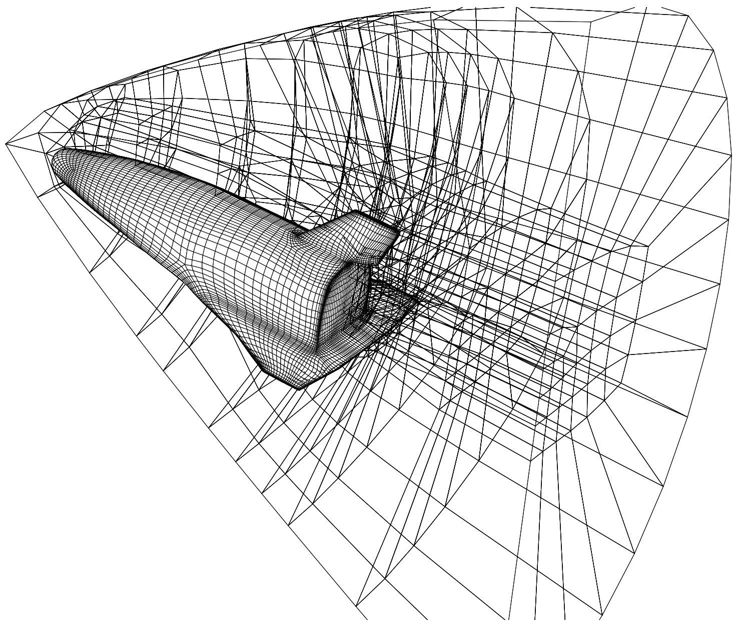


Figure 3: The figure depicts the wireframe of the EXTV configuration, consisting of 390 blocks. To obtain a grid topology, the complete EXTV geometry was decomposed into a set of components according to either engineering or geometric features. All sub–topologies were generated at this lower level.

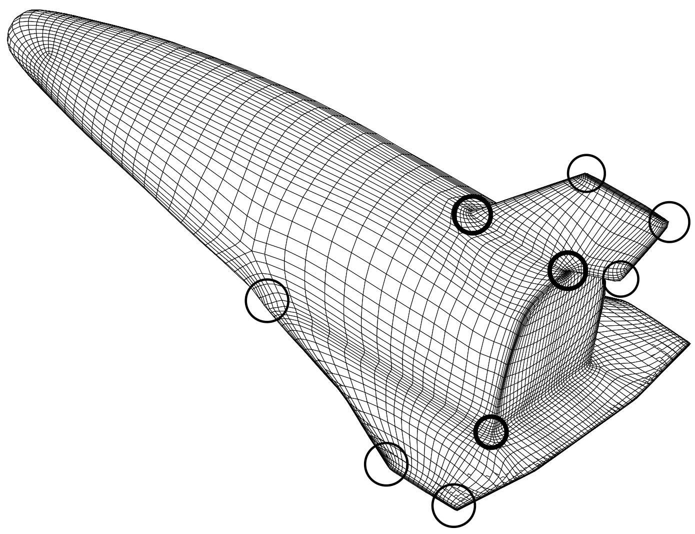


Figure 4: The regions marked by circles are chosen to evaluate grid quality, since the critical features are found in these regions. Grid qualitiy is measured by grid line distribution, smoothness, and cell aspect ratio. Clustering of grid lines is required for geometric accuracy to mesh surfaces with large curvature, while smoothness and cell aspect ratio are needed to improve numerical accuracy and computing speed.

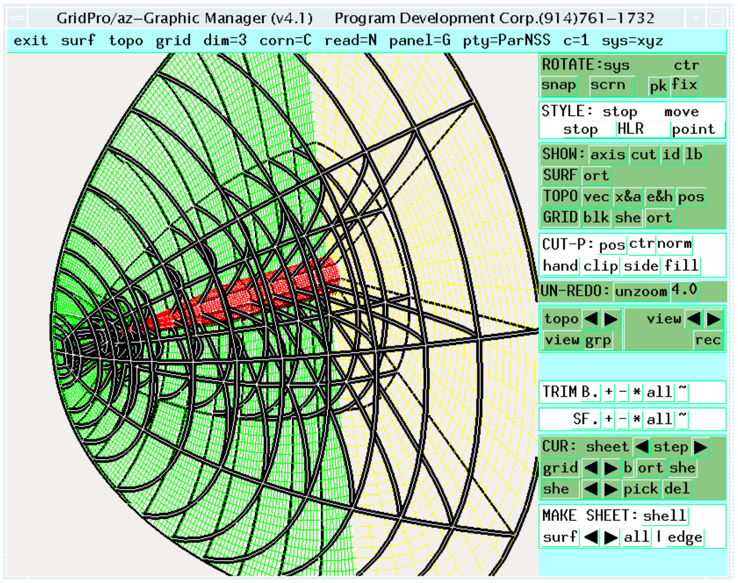
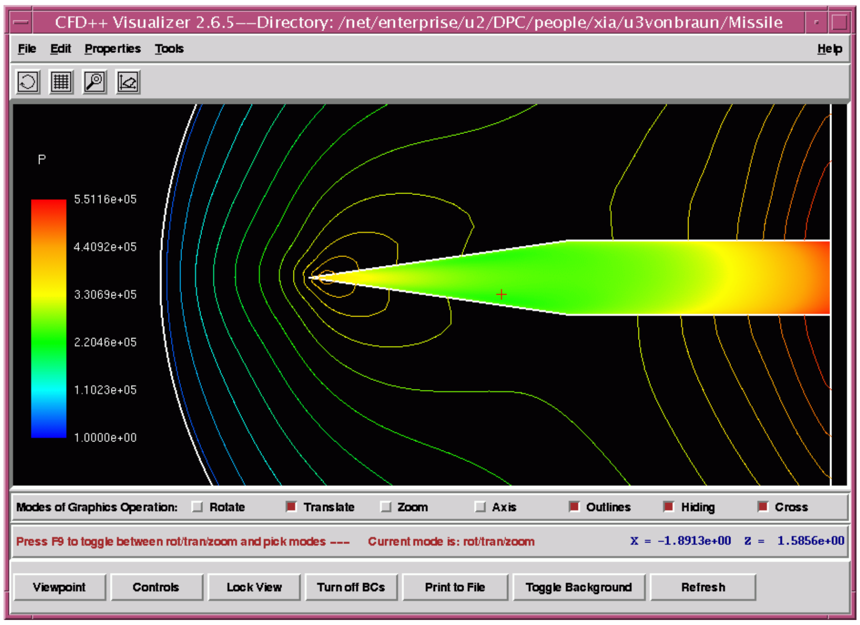


Figure 5: The nose of this generic missile ends in a singular point. The usual approach to meshing such a geometry is to generate an axis–symmetric grid. However, the resulting singularity at the nose substantially reduces the convergence speed, or necessitates special numerical treatment. A more sophisticated wireframe avoids the singularity, but exactly retains the geometry of the pointed nose. Colored surfaces indicate boundary conditions that were set interactively in the grid generator.



Press F9 to toggle between rot/tran/zoom and pick modes --- Current mode is: rot/tran/zoom X = -1.8913e+00 Z = 1.5856e+00

Figure 6: Having set flow boundary conditions interactively in the grid generator, the missile grid data that was converted into unstructured StarCD format is directly imported by the flow solver, for instance, CFD++. This provides CFD engineers with some kind of push button technology, once the grid has been generated.

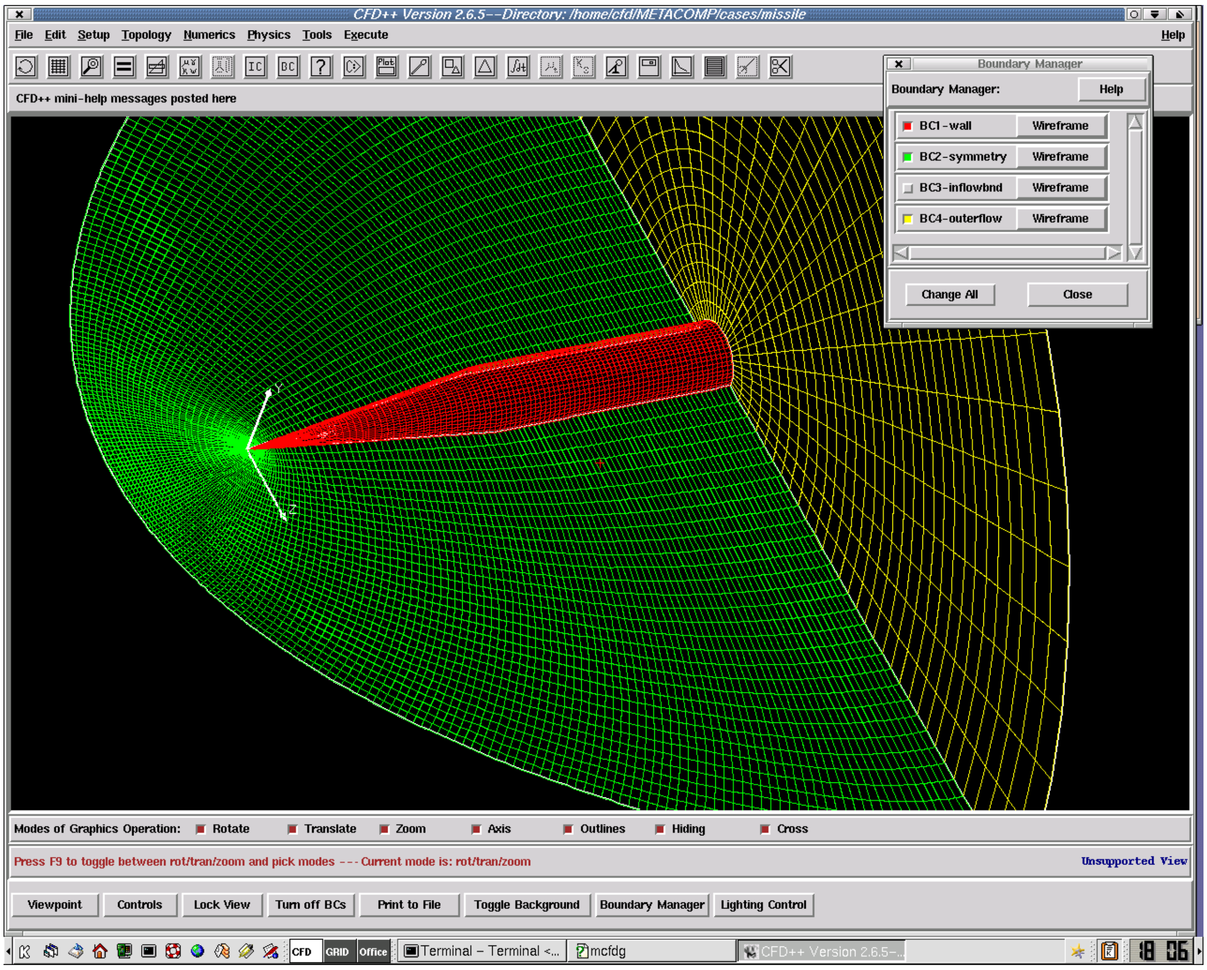


Figure 7: GridPro also provides interfaces to numerous commerical flow solvers. That means, having generated the grid, the user can interactively specify boundary conditions. In this example, a StarCD output was generated and directly read by the viewer tool of CFD++ as shown in the figure. The small window in the upper right depicts the boundary conditions, where each color indicates a different boundary condition whose type is listed next to the color squares. A comparison with Fig. 5 shows that the boundary conditions were exactly reproduced as generated in GridPro.