

State of the art – Background

Due to the continuous advances in computer hardware technologies and in algorithmic performance, CFD has become a strategic design tool for the aeronautical industry. This leads to the current trend towards “High-Fidelity”; CFD whereby increasing complex geometries and physical effects are being simulated with up to hundreds of millions points, resulting in costly computations. Hence, there is a growing need towards further improvement of the current CFD computation chain in order to minimize computational costs and engineering time.

Many of the current industrial CFD solvers are based on explicit Runge-Kutta time integration schemes, associated with a low level of implicitness through a residual smoothing step. This kind of flow solvers have a very low CPU cost per iteration, but are only conditionally stable, leading to a very low time steps. On the other hand fully implicit solvers, based on techniques such as conjugate gradient and GMRES-type methods are applied with reasonable success, at the expense of additional complexity and prohibitive memory requirements for very large scale problems.

Hence, weighting the conflicting requirements on CFD algorithms, it is believed that there could be an optimum by combining explicit methods with simplified implicit steps, in the line of the implicit residual method, generally applied with explicit Runge-Kutta time integrations. This should be coupled to an effective multigrid method, which appears today as the most effective convergence acceleration technique. Therefore, a proper combination of explicit time integration coupled to effective, but simple implicit steps, might be optimal, both in terms of convergence efficiency, memory requirements and algorithmic simplicity and robustness.

A promising method to improve solver convergence by relaxing the CFL constraint has recently been reported in the literature by Rossow (2006)¹ and Swanson et al. (2007)². This method is based on the introduction of a strong implicitness into the Runge-Kutta time-marching scheme, replacing the existing strategy of residual smoothing. They reported promising cases for a range of steady cases on structured grids for Euler, laminar and turbulent flows. NUMECA has extended this concept to large scale industrial structured and unstructured hexahedral and hybrid finite-volume solvers with applications to large scale 3D test cases. The achieved

performance gain is significative. Convergence is obtained, at CFL=1000, in all cases, with a convergence to 5 or 6 orders of residual reduction in 50 multigrid cycles.

Another important component of the CAE analysis chain is the treatment of the CAD geometries and the subsequent grid generation.

For very complex geometries, the received CAD files often require cleaning and wrapping to fit the analysis problem under consideration. For CFD, it implies obtaining tight and continuous surfaces of the solid geometries, as well as the ability to remove geometrical features that are not necessary for the CFD objectives. This process can be very long, in terms of engineering and turn-around time even with the adequate software tools. Once a clean CAD definition is obtained, the meshing process starts and can be quite cumbersome for very complex geometries.

The top part of Figure 1 illustrates the current sequence of CAD cleaning and grid generation, incorporating CAD repair and wrapping, followed by the necessity to generate an initial surface grid, before creating the final volume grid. This process can easily take weeks, for very complex geometries, such as a complete car or an aircraft. On the other hand, the lower part of this figure illustrates the new approach with Hexpress™/Hybrid. This new grid generation system from NUMECA, Hexpress™/Hybrid, incorporates CAD cleaning and wrapping tools, as well as automatic, parallel generation of unstructured hexahedral dominant grids, with conforming hybrid additional cells, proceeding from volume to surface, as with Hexpress™, removing hereby the need for an initial surface grid. It operates in parallel on shared memory architectures for further gains in performance. This leads to a gain in turn-around time and engineering time of one order of magnitude.

But the “High Fidelity Simulation” needs by industrials must take into consideration multi-physic aspects. For instance, Fluid-structure interactions (FSI) are an essential component of most of the industrial systems in aeronautics, either at the level of static deformations, or at the level of aero-elastic interactions up to dynamic instabilities, such as flutter, making FSI simulations crucial for an efficient and safe design. FSI simulations concern the interaction between a flow and a deformable structure and combine the two disciplines Computational Fluid Dynamics (CFD) and Computational Structure Mechanics (CSM). During a FSI simulation the CFD

and CSM solvers are commonly performed separately through different GUIs, and have to share information about data at their common interface and the execution of the solvers has to be synchronized from a third interface such as MPCCI. The coordination of the solvers is handled by a dedicated software module. This process can be heavy as it requires an expertise with heterogeneous GUI's.

It obviously leads to the need for a higher level of integration close to a monolithic approach in order to raise FSI simulations to a higher level of efficiency. But other area need coupled computations. Amongst them, in the engineering systems involving convection phenomena, it can be crucial for the safety of the structure to control the heat exchange rate between the fluid and the solid structure. In particular, the turbine blades of the aircraft engine operate with high efficiency in the most hostile environment of the engine. The design of turbine cooling systems remains one of the most challenging processes in engine development. Modern high-pressure turbine cooling systems invariably combine internal convection cooling with external film cooling in complex flow systems whose individual features interact in complex ways. The heat transfer and cooling processes active are at the limit of current understanding and engine designers rely heavily on empirical tools and engineering judgment to produce new designs.

All put together, the pressure on the aeronautical industry to answer the current environmental challenges requires CAE and simulation tools to be able to take into account (1) Geometries including all details, leading to meshes up to the billion of points; (2) Interactions between the flow and the other physical phenomena, such as the fluid-structure interactions, aero-acoustics, heat transfer; (3) High fidelity simulation results leading to the use of LES unsteady simulations, in particular for acoustic broadband predictions and combustion. These objectives lead naturally to a fast increase of the complexity of the problems and require an efficient large data set management at all stages of the simulations.

As an example, the simulation of an entire car, or a complete aircraft in landing configuration, requires geometry defined by thousand of CAD surfaces and discretized with hundred millions of triangles in STL format. Therefore, simply loading or manipulating this type of geometry can quickly become a nightmare for the user due in particular to the prohibitive quantity of memory required to store the model, the limited capacity of the computer graphics cards or still the slowness of every interaction through the graphical interface. This problem can exist in most of the interactive modules of the simulation chain and is due to

using remote HPC resources by including parallel processing, client/server separation and render server/data server separation₆.

The present project provides a state of the art and a set of specifications and guidelines towards a new generation of software environment, responding to these objectives.

Finally, the use of graphics processing units (GPUs) is a relatively new approach in high performance computing. Nevertheless these many core co-processors (in opposition in the multi-core x86 processors) have shown to be very capable in accelerating a wide range of application codes by a factor up to 100 over a typical modern server CPU core. GPUs implement a many-core hardware architecture with hundreds of processing cores per GPU chip and a very high bandwidth memory interface. The latest generation of Nvidia "Fermi" GPU's₇ includes up to 512 compute cores per chip and a memory interface with up to 200 GB/s bandwidth.

Nvidia is the main player in the GPU computing field with the Nvidia CUDA₈ technology, a combination of hardware features and software tools, being the key to the development of GPU enabled applications. AMD, the other main supplier of graphics hardware, does not (yet) offers competitive tools for pure GPU compute applications. And the competition must count on other actors such as INTEL which has already announced the release of the MIC₉ architecture to enter into the competition of the hardware acceleration. In this race some unified and easy to use middleware technologies appears on the market. OpenCL₁₀, a unified programming model for all GPUs, is still in its infancy and does not provide competitive performance compared to Nvidia CUDA technology. Compilers and frameworks also exist to support GPU programming_{11, 12}. They permit to instrument the code in a hardware-independent way and offer the potential advantage of generating the best optimized code by taking into account the underlying hardware at compile time.

Especially workloads with high computational intensity, as they appear in CFD codes, are very well suited for hardware acceleration in general, and for GPUs in particular. Several groups in the scientific community have already shown the potential of GPU computing for CFD applications. Amongst them, a research CFD code currently under development at the IMSC at the University of Graz, for multi-physics Euler equations on unstructured grids already demonstrates a speedup up to 150 on a single GPU board compared with a typical modern CPU core, and a single GPU compute node with eight GPUs is approaching the 1000x speedup mark for the

Euler simulation, making a strong case for GPU computing in CFD.

Moreover, the amount of GPU development for research CFD is growing rapidly, as is evident from several important CFD conferences that feature sessions dedicated to GPU co-processing. These include the 48th AIAA Aerospace Sciences Meeting, from January 2010; a session of 8 papers on GPU computing accepted for presentation at Parallel CFD 2010 and a GPU-CFD Minisymposium featured at ECCOMAS CFD 2010 in June 2010. Several developments within the past year have demonstrated that CFD codes can benefit from GPU co-processing. One of the most encouraging trend refers the discontinuous Galerkin methods on GPUs by A. Klöckner et al. (2009)³, and unstructured grid CFD solvers on GPUs by A. Corrigan et al (2009)⁴, (2010)⁵.

The fast advancements on the GPU hardware side, especially driven by Nvidia corporation with their focus on the GPU computing, and realized in the latest generation "Fermi"-GPU's, will enable even higher performance levels. The generation of Nvidia Fermi-GPUs includes advanced hardware features that are important for commercial CFD applications, like ECC memory, L1/L2 cache hierarchy for improved application performance, 64bit address space for large onboard memory support, and full support for C++.

These advancements underline the high potential of GPU computing to significantly enhance the performance of real world CFD simulations. In this context, the structured flow solver FINETM/Turbo can be accelerated. The computation of the fluxes and the residuals, one of the most time consuming computation part, can be offloaded in order to gain an order of magnitude on the global computation time. It will open the door to further accelerations thanks to the instrumentation of other modules, but also thanks to the coming improvements of the co-processors.

1 Rossow, C.-C. (2006) "Convergence acceleration for solving the compressible Navier-Stokes equations", AIAA J., vol. 44, pp. 345-352.

2 Swanson, R.C., Turkel, E., Rossow, C.-C. (2007) "Convergence acceleration of Runge-Kutta schemes for solving the Navier-Stokes equations", J. Comp. Phys., vol. 224, pp. 365-388.

3 Klöckner, A.; Warburton, T.; Bridge, J.; Hesthaven, J. S., (2009). Nodal discontinuous Galerkin methods on graphics processors, Journal of Computational Physics, Volume 228, Issue 21, p. 7863-7882.

4 Andrew Corrigan, Fernando Camelli, Rainald Löhner, and John Wallin, (2009). Running Unstructured Grid CFD Solvers on Modern Graphics Hardware 19th AIAA Computational Fluid Dynamics Conference, San Antonio, TX, AIAA-2009-4001.

5 Andrew Corrigan, Fernando Camelli; Rainald Löhner, (2010). FEFLO: Porting of an Edge-Based CFD Solver to GPUs AIAA Paper-2010-0523, 48th AIAA Aerospace Sciences Meeting, Orlando, FL, USA

6 Andrej Cedilnik, Berk Geveci, Kenneth Moreland, James Ahrens, and Jean Favre (2006). "Remote Large Data Visualization in the ParaView Framework". A. Heirich, B. Raffin, L. P. Santos (eds.) editors, Eurographics Parallel Graphics and Visualization 2006, pages 162—170.

7 Fermi GPU compute architecture website, <http://www.nvidia.com/fermi>

8 Nvidia CUDA website, <http://www.nvidia.com/cuda>

9 Intel Many Integrated Core Architecture website, <http://www.intel.com/content/www/us/en/architecture-and-technology/many-integrated-core/intel-many-integrated-core-architecture.html>

10 OpenCL website, www.khronos.org/opencv

11 Portland group website, <http://www.pgroup.com/>

12 HMPP website, <http://www.caps-entreprise.com/index.php>

Objectives

The objectives are divided into 3 work packages. The first work package proposes to: (1) apply the convergence acceleration technique to GRA-LNC configurations, and extend its range to transition models and to unsteady flows; (2) extension and apply Hexpress™/Hybrid to the GRA-LNC configurations. It is expected to contribute at least one order of magnitude to the current objectives in performance gains.

The second work package provides a state of the art and a set of specifications and guidelines towards a new generation of software environment, responding to new objectives of the industrials in term of software integration and multi-physic simulations.

Finally, the thirds work package underlines the high potential of GPU computing to significantly enhance the performance of real world CFD simulations. The computation of the routines relative to the flux computation into the structured flow solver FINE™/Turbo are offloaded to demonstrate the capability of industrial codes to be accelerated through co-processors and many-thread architectures.

Description of work

WP1 - T1.1-6: Pre-processing adaptation of Hexpress™/Hybrid to the GRA-LNC configurations

An Euler mesh has been generated from the CAD file, provided by Fraunhofer, V2 CLEANSKY-GRA-NLG-Shortened_AiCATPart.igs. The mesh generation in the graphical user interface of HEXPRESS™/Hybrid, involves the following steps. Once the model is imported, as a CAD file, selections have to be defined. These selections are needed for both refinement purpose and boundary conditions definition. In the landing gear mesh generation, two selections have been considered. These selections are illustrated on Figure 2. A $4 \cdot 10^6$ cell mesh was generated in 38.51 minutes in sequential against 10.44 minutes with 8 threads. This mesh is mainly composed by hexahedral cells ($2.32 \cdot 10^6$ cells or 60% of the global mesh) with a minimum orthogonality of 11.5° (two cells only below 15°) and if the ground is removed from the mesh, the minimum orthogonality increases to 33° .

Boundary layers have been added, leading to a finer mesh, refined close to the wall in order to generate a Navier-Stokes mesh and insert the viscous layers. The result is shown in the figures 3 and 4. The generation of this $6.35 \cdot 10^6$ cells mesh has been achieved using eight processors in 20.44 minutes ($3.68 \cdot 10^6$ hexahedral cells, 60% of the global mesh). To check the ability of FINE™/Open

to run a calculation on this geometry, a calculation has been launched using standard boundary conditions. Only the convergence plots of the density and the energy are given on the figure 5 and visualization is shown on the figure 6.

The second configuration is a plane in landing condition. For this configuration, the flaps are fully extended. Navier-Stokes mesh has been generated from the CAD file, provided by Fraunhofer. The CAD geometry is illustrated on figure 6. Four refinements domains are defined with HEXPRESS™/Hybrid, the fuselage, the wing, the flaps and the sharp edges such as trailing edges of the flaps and wing (see figure 7) 8 processors were used to generate a $34 \cdot 10^6$ cells (75% of hexahedral cells) mesh in 1.49 hour. To check the ability of FINE™/Open to run a calculation on this geometry, a calculation has been launched (perfect gas, Spalart-Allmaras turbulent model, boundary condition fixed by Fraunhofer) using a multigrid method (4 grid levels). This computation was compared to the same one run with the CPUBooster. Results and comparison are shown on the figures 8 and 9: the CPU-Booster calculation needs 210 iterations to reach a convergent state on both lift and drag whereas 1500 iterations are required for the default technologies in FINE™/Open. This corresponds to a reduction of more than 7 times in the number of iterations and a global computation 3.85 times faster.

The improvement of engineering process time has been shown, using NUMECA chain software on GRA-LNC configuration provided by Fraunhofer. This study shows the ability of NUMECA chain software to deal with complex geometries from real CAD files. Innovative modules such as CPUBooster are then used, illustrating a real gain in terms of engineering process time.

WP1 - T1.1-12: Extension of the convergence acceleration modules to transition and unsteady flows

The boundary layer which develops on the surface of a solid body starts as a laminar layer but becomes turbulent over a relatively short distance known as the transition region. This Laminar-Turbulent transition is a complex and not yet fully understood phenomenon. Among the numerous parameters that affect the transition one can list: the free stream turbulence intensity, the pressure gradient, the Reynolds number, and the surface curvature. Furthermore, predicting the onset of turbulence is a critical component of many engineering flows (Mayle, 1991). It can have a tremendous impact on the overall drag, heat transfer, and performances especially for low-

Reynolds number applications. However, most of the standard turbulence models fail to predict the transition location. Numerous difficult models deal with transition in a turbulent model. Numerous works have detailed the way to deal with transition in a turbulence model, but they are generally difficult to handle in an industrial context. A much simpler approach has been retained hereafter.

It is proposed to include a transition model into the original Spalart-Allmaras [1992] turbulence model in order to take into account the transition onset at a certain chord distance on a blade pressure and suction sides. The original version of the Spalart-Allmaras turbulence model already includes the option to include additional terms in order to maintain the laminar region and to add "trip" terms to promote turbulence at transition location. The transition location (transition line in 3D) should be imposed either through a user input or via a separate guess. The latter is based on the empirical correlations from Abu-Ghannam and Shaw [1980] and derives from experimental data for transition on a flat plate with pressure gradients.

The implementation of this transition model into FineTM/Turbo CFD suite, have been conducted on a gas turbine nozzle, measured and computed heat transfer coefficients (HTC) along the nozzle airfoil have been compared (see figure 10). This case of a flow over a heated linear turbine airfoil at a Reynolds number of 230,000 and at low Mach number is documented in Radomsky and Thole [2000].). The Spalart-Allmaras turbulence model coupled to a transition model shows an excellent behaviour, with turbulence model activated downstream of $s/C=0.6$. This indicates that the heat transfer prediction is almost driven by laminar to turbulent transition. For all models the flow is laminar around the leading edge and the predicted HTC is in accordance with measured value. This approach is now widely used in the industry and several conference papers have been published using this method proving the ability of the method to properly handle laminar to turbulent transition (see for instance Johann and Swoboda 2007, Tartinville et al. 2007, Becker et al. 2009).

A new acceleration technique has been developed by NUMECA in FINETM/Turbo and FINETM/Open CFD suites in order to improve the convergence rate. This new method denoted CPUBooster, permits to increase the CFL number up to 1,000 on standard industrial configurations. The method is based on the introduction of more implicitness into multigrid explicit Runge-Kutta schemes. This leads to a drastic reduction of the total CPU time required to converge such a

simulation. This method has been adapted to unsteady simulations through the introduction of implicitness in the dual-time stepping loop. Furthermore, it has also been adapted to the transition model described above.

The subsonic Durham turbine cascade was selected in order to test the acceleration of the convergence for transition model. Two calculations have been defined, one fully turbulent and the second with transition onset along the blade based on Abu-Ghannam and Shaw criteria for both pressure and suction sides of the blade. Both simulations launched with the CPUBooster, converge to a residual of -5.8 in 50 iterations on the finest grid level (see figure11). The thick red line on the left panel corresponds to the location of the transition onset (intermittency equal to 1). One can see the difference in limit streamline on the suction side of this blade using either a fully turbulence boundary layer or a transitional boundary layer. Additionally, in order to compare the speed-up obtained by using the CPUBooster a reference simulation was conducted without this option activated but with the transition module. Both simulations have converged by more than 5 orders of magnitude. In other words, the CPUBooster simulation is 4.12 time faster than the reference one.

In order to test the adaptation of the CPUBooster to unsteady flow a series of simulations have been performed. One of these cases is devoted to study Shock Boundary Layer Interactions (SBLI). It corresponds to a 3D transonic channel flow at $M=1.3$ with a shock due to the thickening of the turbulent boundary layer along sidewalls (Bruce and Babinsky, 2008). Preliminary simulations have been performed in steady mode both with and without the CPUBooster option activated. The convergence and the final solution can be depicted on Figure 12, and one can note the CPUBooster offers an accelerating factor of 2.8.

Finally, to combine the two different adaptation of the CPUBooster to unsteady flow and to transitional flow, a low-pressure turbine cascade with separation along the suction side has been launched. Experimental data are available in both steady an unsteady mode (Stadtmüller and Fottner, 2001). Since the separation is quite important, a Dettached-Eddy-Simulation (DES) has been performed on this test case. The convergence history of a series of physical time steps is displayed on figure 13. This shows that the residual has decreased by about 2 orders of magnitude within each physical time-step.

According to all the tests that have been conducted so far, most of all the steady RANS fully turbulent simulations are converged in about 50 cycles when using the new acceleration technique (reduction in CPU time by a factor of

the order of 4). The adaptation of this method to transitional flow has been performed. The transition does not impact on the convergence rate and on the efficiency of the acceleration method.

WP2 – Inroad towards multi-physics aerodynamics simulation with reduced algorithmic complexity

The current evolution of industrial oriented CFD in the aeronautical field towards high-fidelity simulations, and the rapid development of massively parallel computing, calls indeed for a new approach to the complete CFD-multi-physics simulation chain. This goes from pre-processing, to very fast new generation of basic CFD tools and algorithms and to efficient, full parallel post processing. In this context, NUMECA Int. has defined and started a new strategic plan dedicated to the development of a new numerical simulation environment able to answer new ambitious objectives for years to come: (1) bring together all Numeca software into a unified environment, where the user navigates transparently across the different modules; (2) define a unified architecture and project management for all software components; (3) extend the project management to handle complex problems involving many geometrical components and complex physics; (4) build an open multi-physics environment; (5) thoroughly rethink the graphical user interfaces and the way the user interacts with the modules; (6) enable treatment of very large data set up to the billion of points; (6) enables direct link to CAD for parametric design loop; (7) simplify the user input through an expert System using the experience accumulated by the users; (8) massively parallel approach for all the steps of the simulation including multi-resolution and streaming methods in the pre/post processing phases; (9) enable co-processing and automatic reporting; (10) enable external computing resources usage through a web services approach. This work-package described the accumulated experience obtained during the current development of this new numerical simulation environment.

Unlike the current NUMECA software able to manage classical CFD user's workflow, the new environment proposes to handle complex simulations combining different areas and different physics like the study of the conjugated heat transfer inside the turbine combined with the external flow field (see figure 14). It also includes the mesh generators from both FINE™/Turbo and FINE™/Open to respectively mesh the rotating blade to blade domain and external domain, the solid body of the blade and

the cooling channels, and finally merge the domains before starting the simulation.

One of the common important requests of the user is to simplify the way the data are exchanged between different collaborators working on a common global project. For complex project, redundancy of data (copies of projects, meshes, etc) implies that the users must continuously check if their local versions are up-to-date and consistent with their collaborators. Hence, a client-server approach has been adopted. A client, an instance of OMNIS, runs on a specific machine and is controlled by a specific user: other available resources are seen in the cloud and can exchange data and perform sub tasks. The peer-to-peer technology implies all the clients can potentially interact together on common shared and distributed data. As a consequence, the client-server environment needs the following piece of component. (1) The project is the root object of the data structure managed by a client and points the current projects managed by a user, each of them pointing documents. The peer-to-peer approach is used to control where the documents are opened for edition by clients. (2) The command manager. Each document has a master client: all the requests are appended in a stack on the master, scheduled and re-distributed to the clients working on the same document.

The foundation of the GUI has been re-thing. In general, the human interface is imposed by the vendor. The new concept of document intends to change this common approach to give full flexibility to the user to customize the interface according to his specific needs. Each "document" (see figure 15) is composed by the simulation data (geometries, meshes, results, etc) and a list of pages (description of a layout) containing page elements (graphical view, widget to change object properties or to activate tools, test boxes, etc). From the Power Point like GUI, the user can create, select, edit and delete pages. In the current prototype, the page elements are created by selection through a search engine. Using this approach, the user can start a new document from a white page or load a predefined template document dedicated to a specific objective.

A complex data management is also mandatory to achieve the objectives. For instance, a component is a physical object defining a part of the problem (solid body) to be studied. The project implicitly contains several components like the compressor, the combustion chamber, the turbine and the case of the engine. Each component can be subdivided into several sub-components, i.e. the compressor can be divided in stages and each stage in rotor and stator (see figure 16). A (sub) component is similar to the assembly defined in

CAD systems, used by the user to: (1) set several cinematic laws called degrees of freedom (DOF), combined with the DOF of its parent; (2) define domains of simulation (space area around or inside a component, in which the simulations are performed) which can be either solid or fluid (see figure 17). Finally, each boundary can be characterized by a physical type, and even be connected to other domains from another physic. All put together, a project for further simulation may now contain several hierarchical components, geometries, domains, meshes and physics, interactions, and simulations. As a consequence, all these elements are organized in a Linux file-system like tree, allowing inheritance and links to minimize the memory footprint.

To handle large data and large simulations, modern data structures, partitioning tools for structured and unstructured topologies, resource management for both shared and distributed architecture, parallelized algorithms, cloud computing, etc, have been revisited to distribute the simulations, maximize the speed access, save the memory, optimize the resources and such, in all the steps of the process.

On top of this environment, as a front end, a new graphical engine based on OGRE (library running on top of OpenGL) and QML, respectively inherited from the video games and the mobile area, has been developed and optimized. The optimization has been done by grouping together the graphical objects having the same drawing properties. It intends to reduce at maximum the number of calls to the graphic card: the speed is no more linked to the number of objects plotted on the screen, but on the number of groups. Promising results has been obtained: complex geometry with up to 1^6 surfaces, > 1000 different materials and $30e^6$ triangles do not affect the user graphic manipulations.

Based on the pre-requirement, a first prototype named OMNIS has been developed. It finally includes: (1) a multi-client architecture enabling collaborative environments; (2) a new GUI based on the concept of document which allows full flexibility for customization and specialization for all the application fields; (3) complex tasks definition combined with a resource manager for massively parallel approach for all the steps of a simulation; (4) a power full graphics engine allowing visualization of large and complex data; (5) a new data structure for large data set and complex project management; (6) a new development framework to speed up the definition and the coding of new features.

WP3 – GPU accelerated routines for CFD in the test environment

An acceleration of NUMECAs structured flow solver FINE™/Turbo by means of GPUs (and more general many-core hardware) can be considered an advantage for the company. Such acceleration comes with major difficulties that have to be overcome. The basic design of FINE/Turbo was laid almost 30 years ago and its architecture does not meet modern software programming paradigms, i.e., does not implement a separation between data structures, logic and algorithms. As consequences, large parts of code need to be reorganized in FINE™/Turbo. It can be considered time consuming and comes with organizational challenges. Additionally, there are several restrictive general requirements stated by NUMECA: (1) only one code has to be maintained; (2) the current CPU performance must not be penalized by code reorganization; (3) the GPU acceleration can be performed in an iterative process with- out affecting the regular enhancements to the solver; (4) efforts invested in a GPU acceleration must be future proof; (5) bit compatibility of results.

By now, directive-based hardware accelerator is the only available technology meeting these requirements: HMPP Workbench has been identified as the only available directive-based compiler applicable to the acceleration of FINE™/Turbo. Such frameworks have reached some state of maturity but still are in an early stage of commercial distribution.

A module called GPU-MOD (see figure 18) has been developed to implement the following tasks: allocation/deallocation of GPUs (1 GPU per compute process); (2) enabling/disabling GPU acceleration; (3) allocation/deallocation of GPU memory and CPU-GPU data transfers of constant/non-constant data; (4) selection of the grid levels that are to be accelerated by a GPU; (5) GPU accelerated routines (mainly flux computations). The main focus relied on the acceleration of the flux and residual computation routines. They belong to the more compute-intense components of FINE™/Turbo and allow a prognosis of potential benefits from GPU acceleration. As leaves of the call tree of FINE™/Turbo, in theory, these arithmetic routines can be instrumented independently. The first instrumentation permitted to observe a speedup from a single routine ranging from 20 to 130 excluding data transfers, but the speedup was less impressive, even poor, including data transfers. Moreover, on smaller blocks the speedups are significantly smaller because of too long data transfer and a GPU not loaded enough. In regards to these first results, one could conclude these numbers are timings for a best-case scenario, the potential speedup strongly depends on the flux model and the among of

arithmetic executed on the GPU, and a sufficient fine grid level is mandatory for a satisfying GPU performance, knowing the data transfer is an heavy limiting factor, especially on small blocks because of a high latency of the GPU.

In a second time, more routines have been ported on GPU and some sequential tests were lead on a multi-block annular turbine case run in multigrid (see figure 19) where only the finest grid were transferred to the GPU. Since the instrumented routines take ~65% of the total run time, the maximal achievable speedup is bounded to a factor 3, but one could achieve an effective overall speedup of 1.66, including the acceleration due to the restructuration. In parallel, the overall GPU speedup drop to 1.29, with comparable acceleration when the data transfer is excluded from the monitoring.

As an outcome, it became clear a GPU acceleration of leaf arithmetic routines is not sufficient and the memory transfer strongly penalizes the GPU performance, which it also dependent on the hardware accelerator. Moreover, the design of FINE™/Turbo based on patterns chosen +20 years ago does not contain a strict separation between logic/program flow, algorithms and data structures, forcing a heavy re-architecturing of the solver with a bounded and not guaranteed acceleration factor. On another hand, HMPP Workbench is a very new piece of technology, continuously evolving and changing with some limitations due to its youngness.

These conclusions reflect the fact the GPU acceleration for industrial code is not mature enough, but it will offer an important room of acceleration in the coming years. In this context, (1) a methodology have been proposed to continuous the efforts to accelerate the FINE™/Turbo structured flow solver and, (2) guidelines using the FINE™/Open unstructured solver have been formulated based on a more modern design in C++. These guidelines must permit to meet the needed high standards for many-core friendly design of NUMECA's flow solvers based on directive-based technologies. While many-core hardware in academic and small scale codes is already well adopted, their application in large scale industrial software is challenging and still a matter of active research. These guidelines finally describes a "pseudo oriented object" design: template free functions using primitive type definitions must permit to accelerate the contained arithmetic and avoid redundant code thanks to the capability of HMPP to deal with function calling, generic programming and the specializations. A special attention is taken regarding the data transfer and thus, the data structure.

Expected results

a) Timeline & main milestones

This project was originally scheduled from February 2011 to February 2013. A 6 month shift was asked and the project was effectively executed from September 2011 to September 2013.

The different milestones and deliverables have been reached without delay or, the delay was explained and tasks were reorganized in respect of the constraints and the expected end of the project.

b) Environmental benefits

The main objectives of the E-CFD-GPU project are oriented at developing a new approach of a complete high fidelity CFD multi-physics computation chain, with a drastic reduction of the turnaround time by several orders of magnitude. Since the engineering requires to couple more and more physics, and the studies of entire models are more and more complex and involve more and more engineers, all the efforts made to reduce this turnaround time through a simplified unified environment or more efficient algorithms taking advantage of the modern hardware architectures, will contribute to make European companies more competitive and even leaders into the aerodynamic fields.

The present E-CFD-GPU project has provided a gain in CPU time of at least one order of magnitude on each phase of the traditional computation chain. On the other hand, all the experiences acquires during the last 20 years up to now, have been used to provides some paradigms to develop a unified multi-users multi-physics platform able to solved hexa-scale problems. All these progresses will assure a big step forward to the reduction of the turnaround time, the cost for CFD/CAA simulations, and thus offer a significant gain in productivity. There is no doubt that this will have a major impact on the aerodynamic field and the European competitiveness thanks to the CLEANSKY initiative. In addition, as several other subprograms, within CLEANSKY and current EU projects, are focused on this problematic, we can expect that the results of the E-CFD-GPU project will benefit other tasks and projects having similar objectives.

c) Maturity of works performed

All the tasks have been achieved in respect to the allocated time and resources, and they all lead to

either impressive results or to important conclusions to develop a “High Fidelity” simulation chain, especially for the CFD. Hexpress™/Hybrid, the convergence acceleration modules (CPUBooster) and its extension to transition and unsteady flows have shown impressive improvements on different stages of the computation chain. On one hand, Hexpress™/Hybrid incorporates CAD cleaning and wrapping tools, as well as automatic, parallel generation of unstructured hexahedral dominant grids, with conforming hybrid additional cells,

proceeding from volume to surface, removing hereby the need for an initial surface grid. This leads to a gain in turn-around time and engineering time of one order of magnitude. On the other hand, the flow solver also offers a non negligible gain thanks to the CPUBooster. This method is based on the introduction of a strong implicitness into the Runge-Kutta time-marching scheme. The achievement is quite significant at CFL=1000 with a convergence to 5 or 6 orders of residual reduction in 50 multigrid cycles.

Picture, Illustration

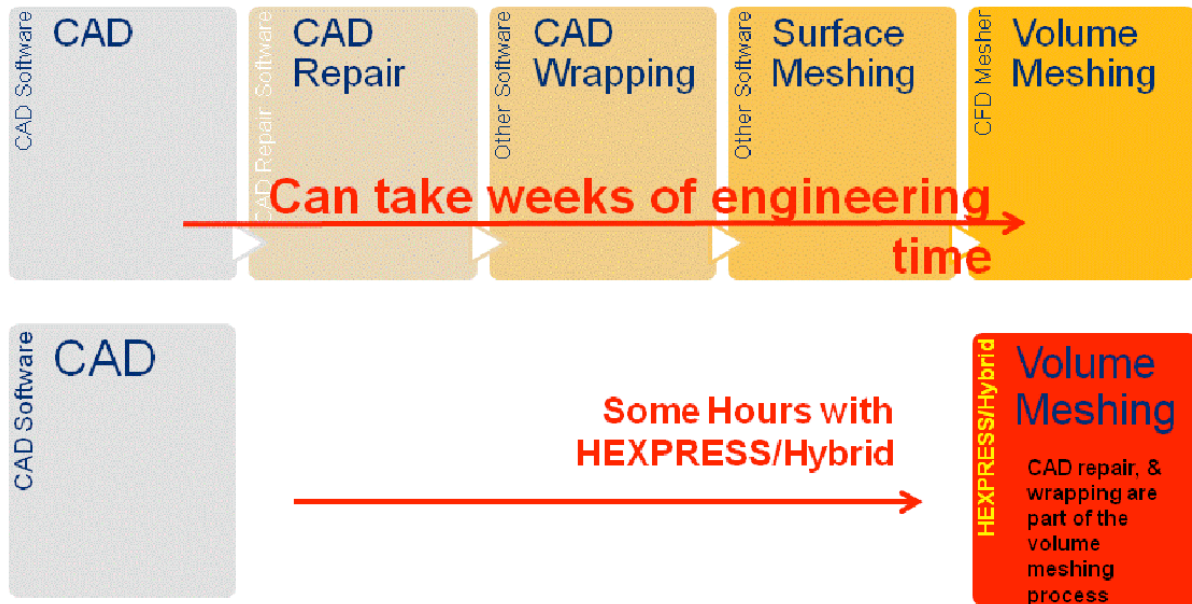


Figure1: Chart comparing the current pre-processing steps with the new approach with Hexpress™/Hybrid

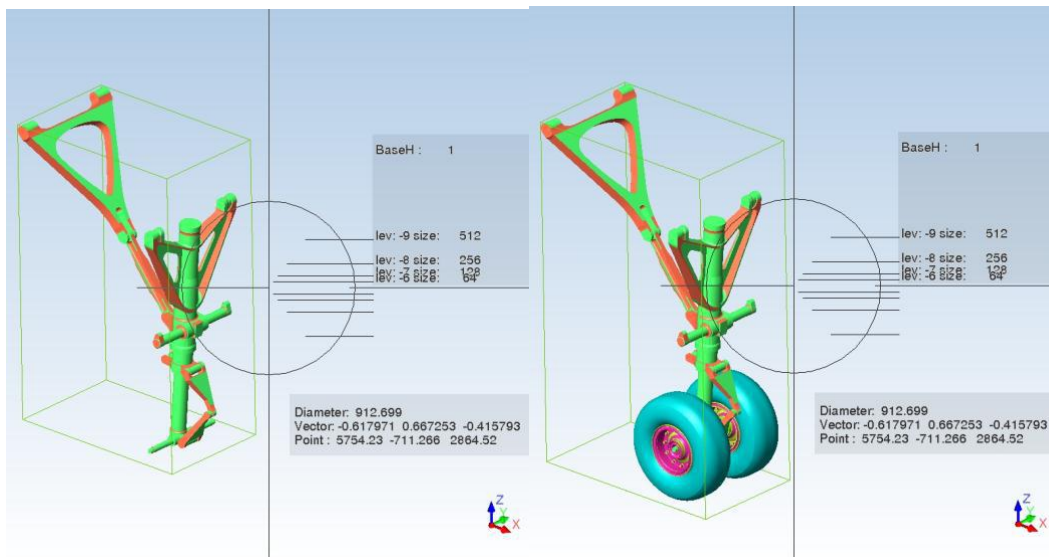


Figure 2: Two selections, the gear (left), the wheel added to the previous selection (right)

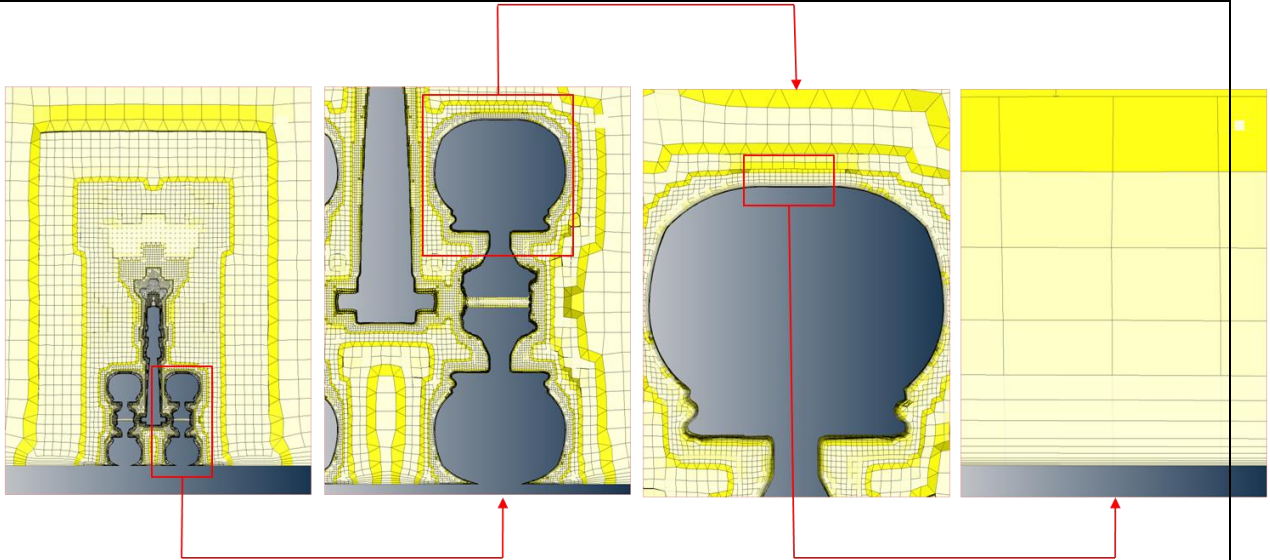


Figure 3: Slices of the configuration across the streamwise direction

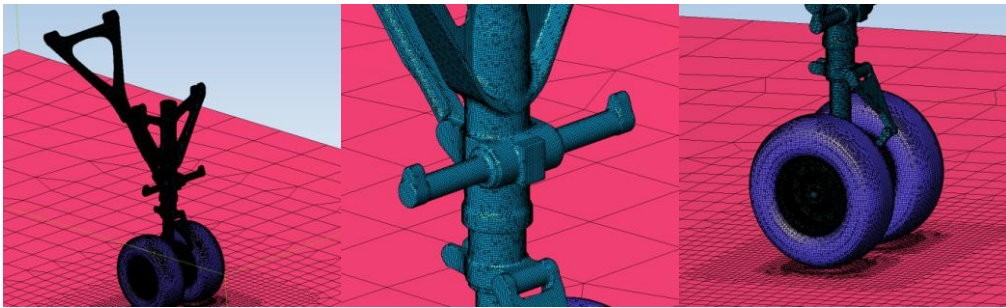


Figure 4: Final mesh of the wheel

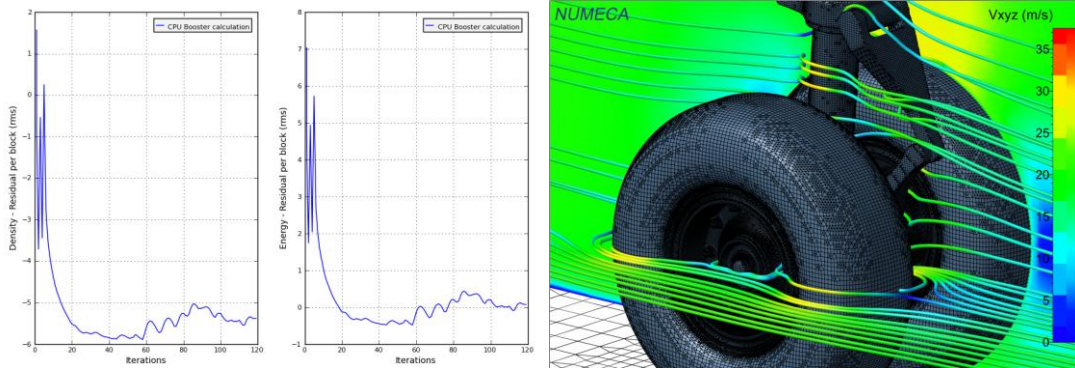


Figure 5: Convergence of the calculation; density residual (left), energy residual (middle) and streamlines colored by velocity magnitude (right)

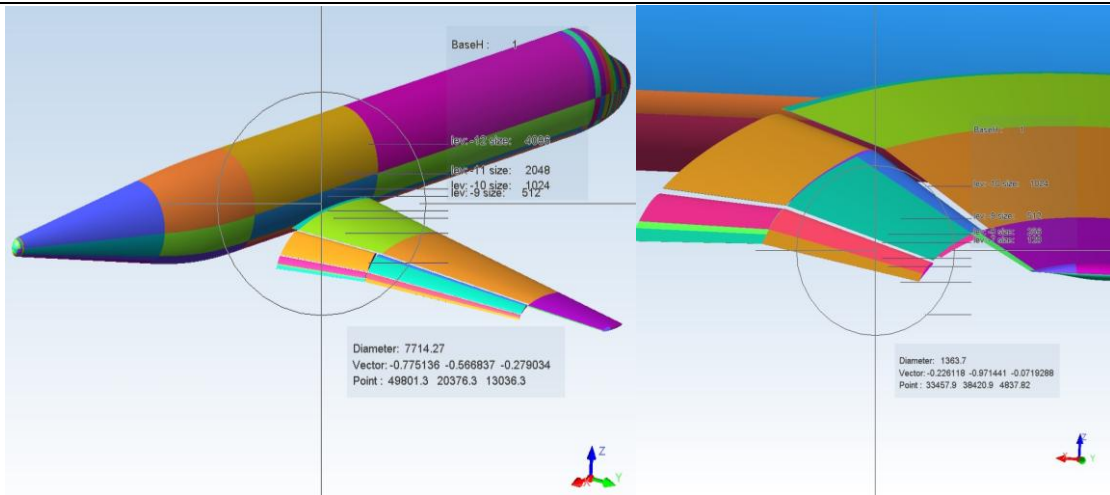


Figure 6: Picture of the CAD provided

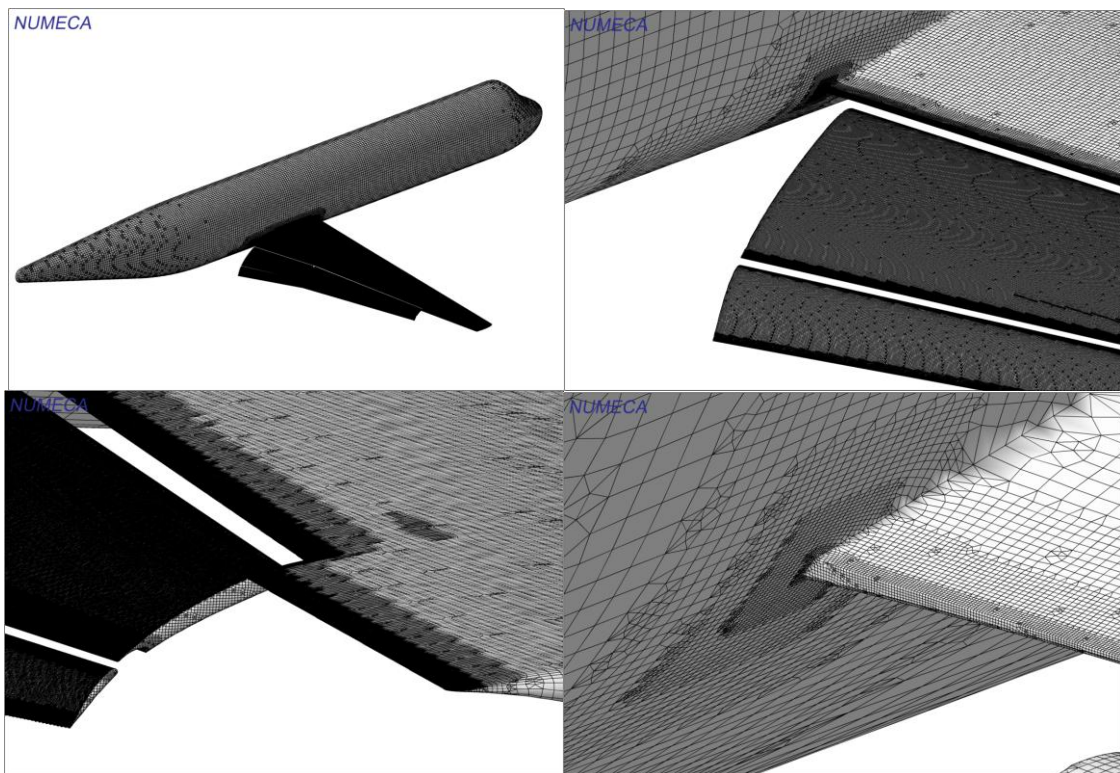


Figure 7: Illustration of the 3D mesh

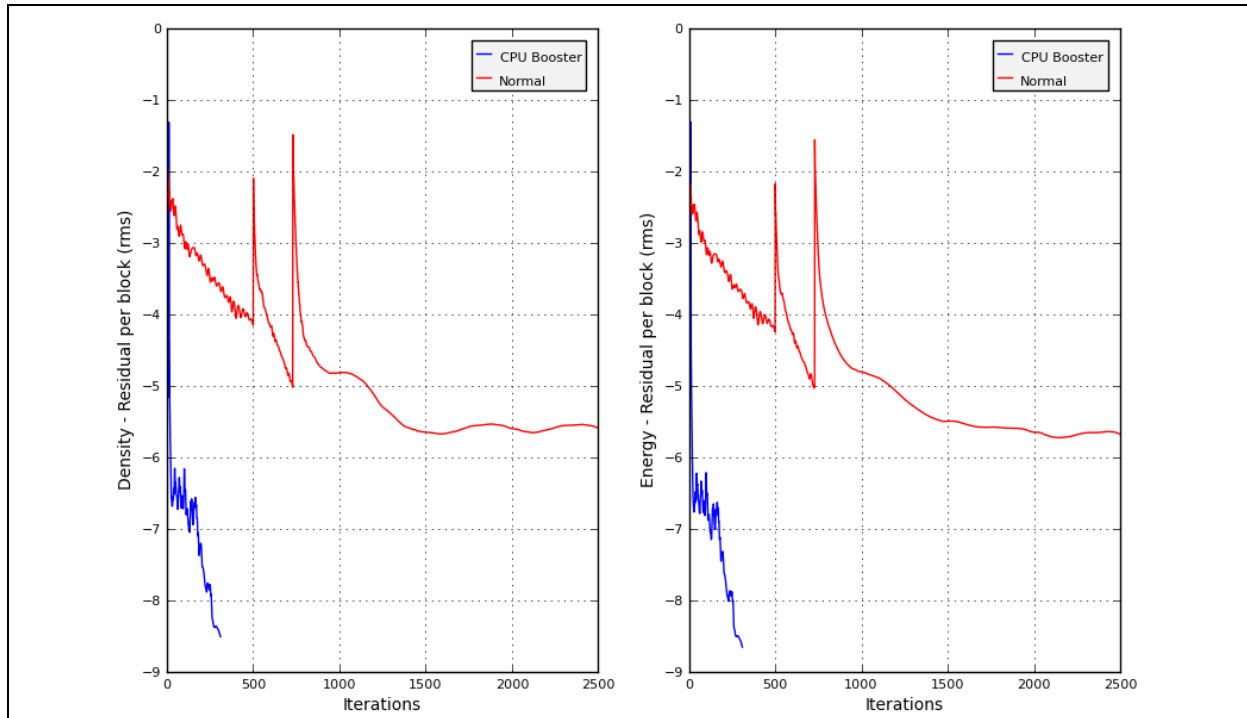


Figure 8: convergence plots in terms of residual for both density (left) and energy (right)

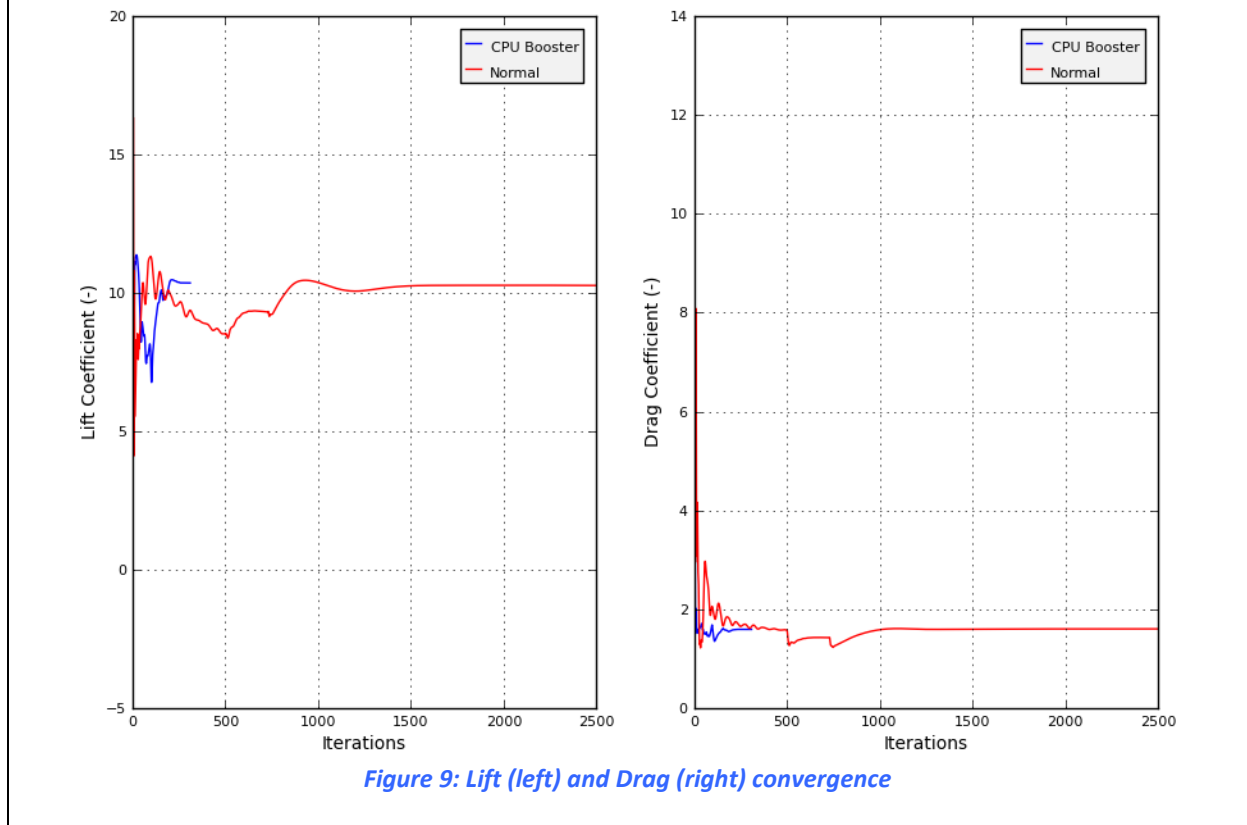


Figure 9: Lift (left) and Drag (right) convergence

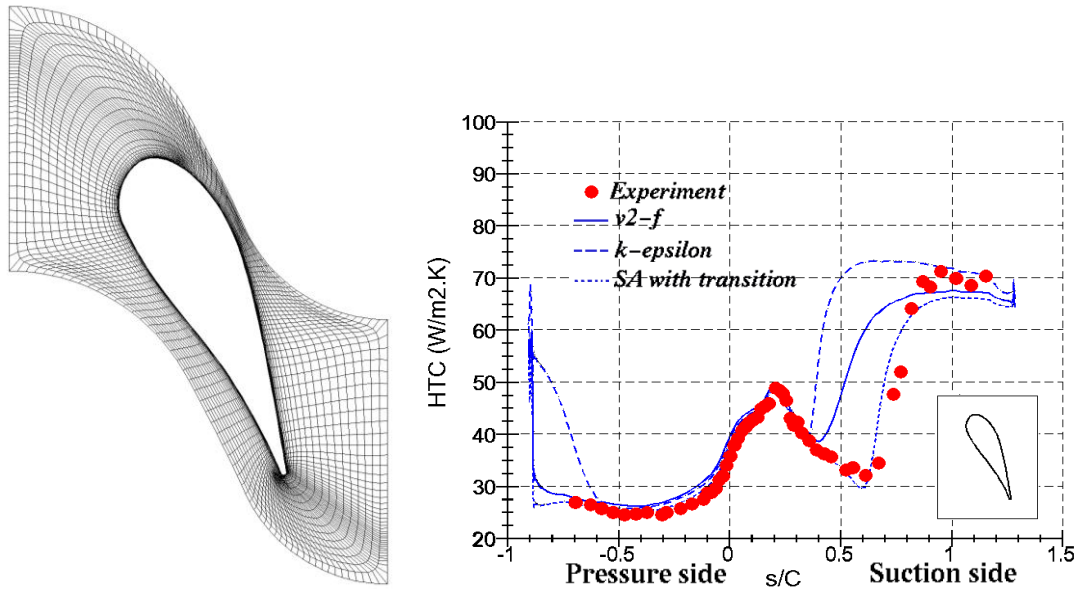


Figure 10: Bi-dimensional mesh for the gas turbine nozzle (right); Computed and measured distribution of heat transfer coefficient along the gas turbine nozzle (left).

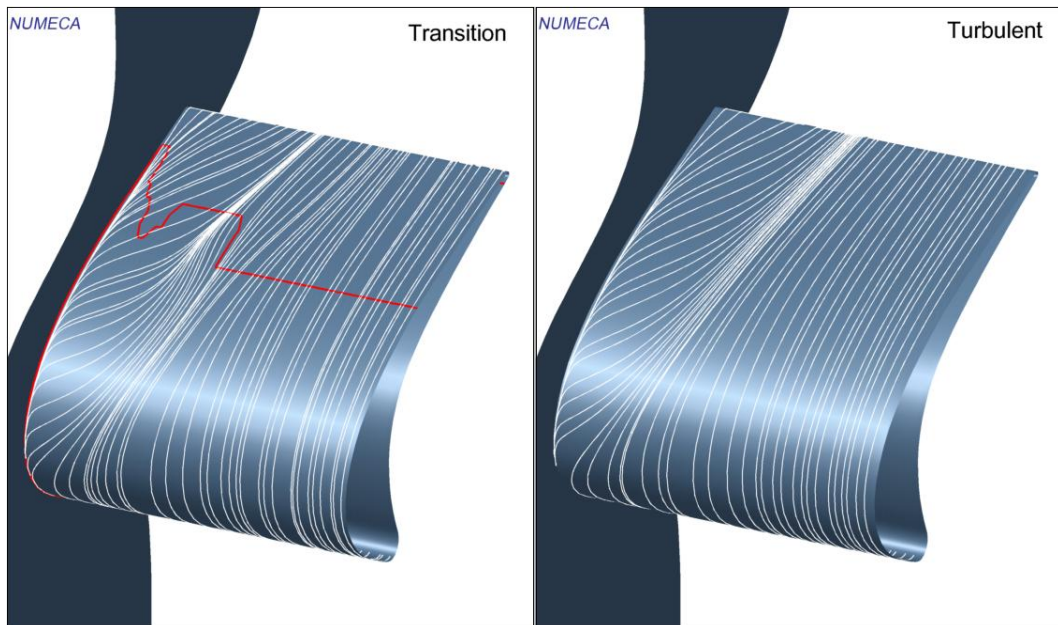


Figure 11: Limit streamline on the Durham turbine cascade suction side using a transition module (left panel) or a fully turbulent model (right panel).

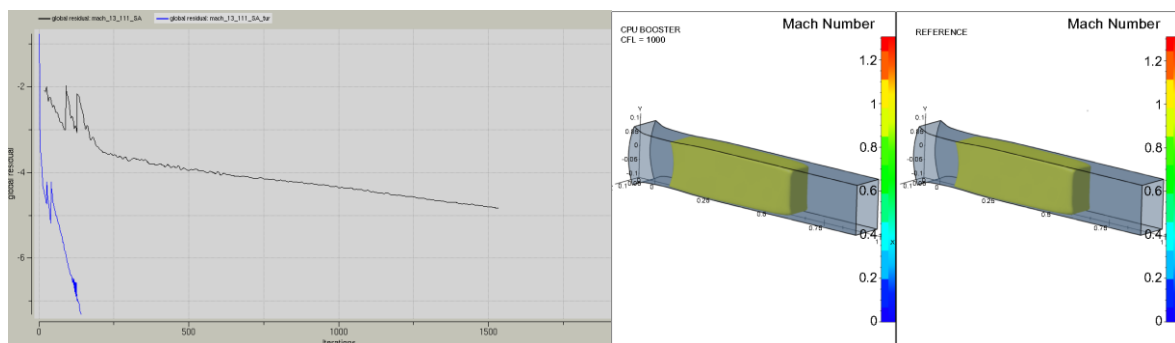


Figure 12: (left) Comparison of convergence history for steady simulations at Mach 1.3 (black for reference simulation and blue for simulation with CPUBooster); (right) Comparison of Mach=1.05 iso-surface for steady simulations

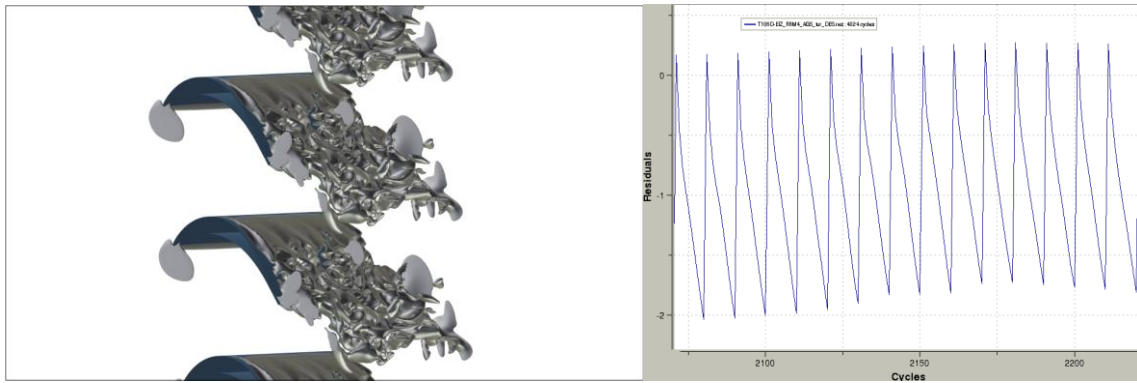


Figure 13: Iso-surface of negative axial velocity (left); example of convergence history (right)

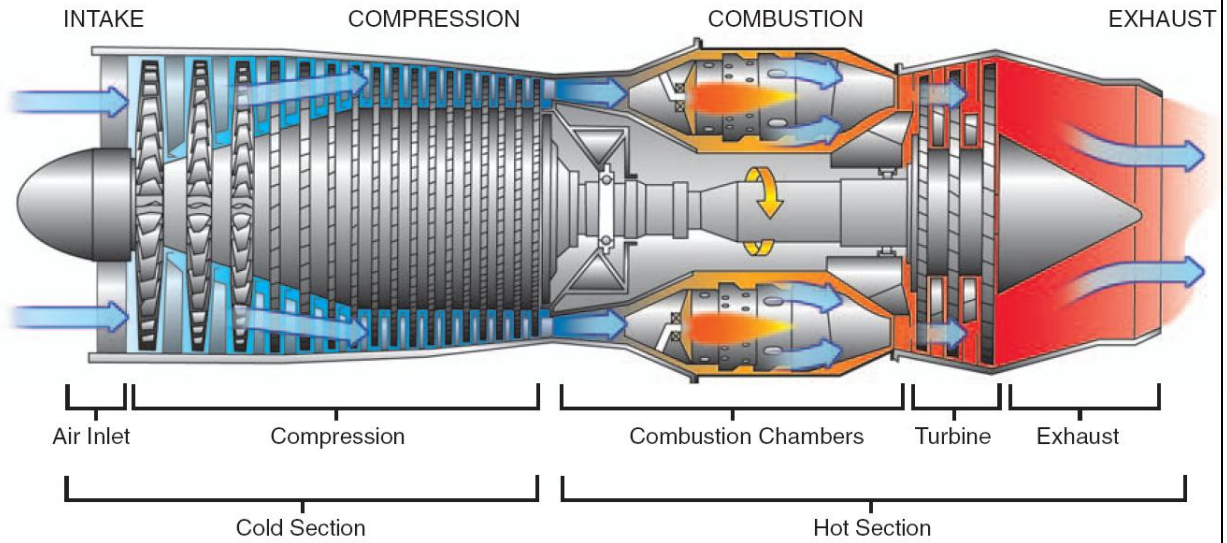


Figure 14: study of an aircraft engine and separation of the problem in different geometry parts

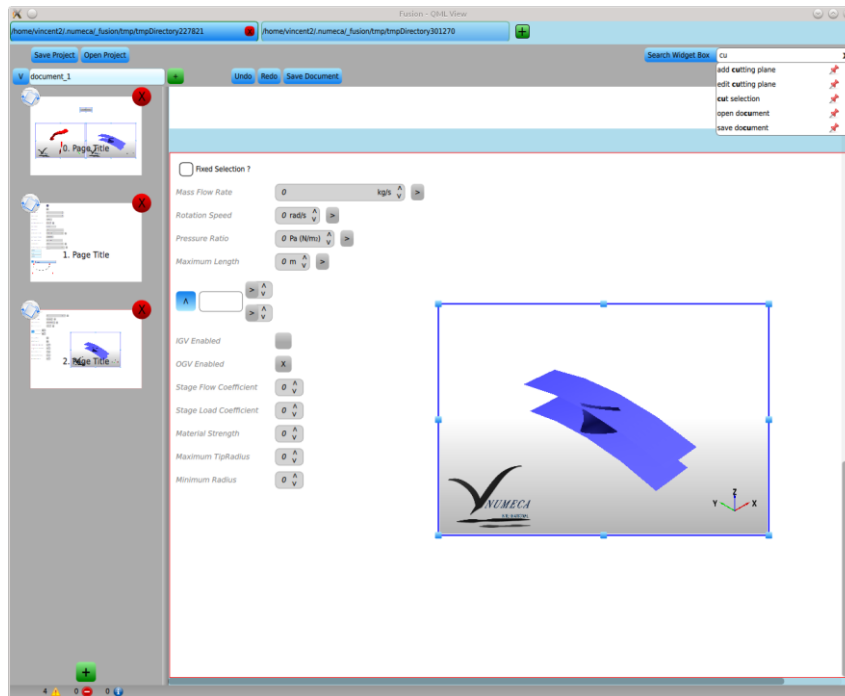


Figure 15: Document GUI prototype

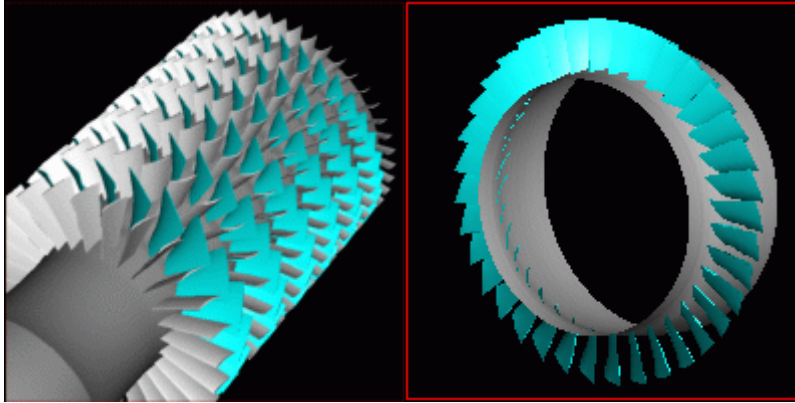


Figure 16: Decomposition of the compressor in stages, and rotor and stator



Figure 17: (from left to right) Solid body of the blade; volume between the blades; mesh of the fluid domain; physical properties and multi-physics interactions

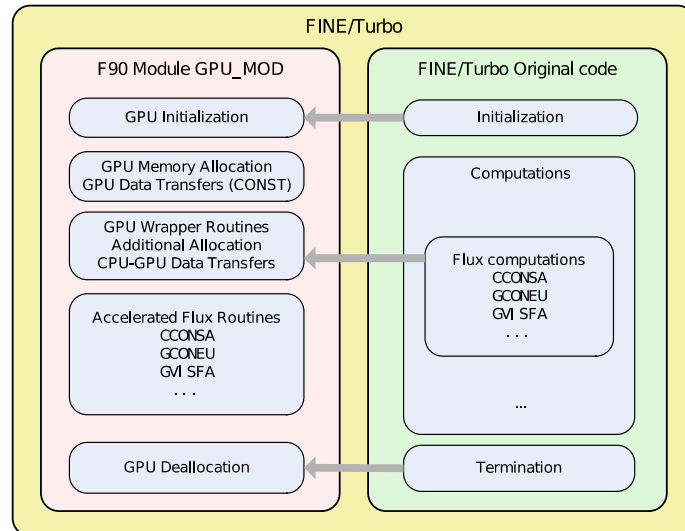


Figure 18: Fortran90 GPU module

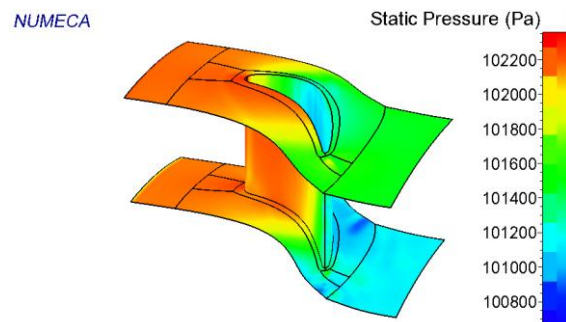


Figure 19: Annular turbine

Project Summary

Acronym: E-CFD-GPU

Name of proposal: Efficient CFD Multi-physics programming research

Technical domain: CS-RTD project

Involved ITD JTI-CS-2010-1-GRA-02-008 / SP1-JTI-CS-2010-1

Grant Agreement: CS-RTD

Instrument: Clean Sky

Total Cost:

Clean Sky contribution:

Call: JTI-CS-2010-1

Starting date: February 2011

Ending date: September 2013

Duration: 30 months

Coordinator contact details: Charles Hirsch

Project Officer: Andrzej Podsadowski

Andrzej.Podsadowski@cleansky.eu

Participating members NUMECA International

Institute for Mathematics and Scientific Computing, University of Graz

Fraunhofer