Boundary Interface Caching as a Method to Accelerate Solver Performance for Industrial Sliding Mesh Simulations on GPU

Siddhartha Gautham A V*
(Siemens Digital Industry Software, India);

Benjamin Ganis
(Siemens Digital Industry Software, USA);

Abstract

Due to the significant benefits being realised in time and cost to solution, the use of General Purpose Graphical Processing Units (GPGPUs) over traditional Central Processing Units (CPUs) is becoming ever greater in the industrial Computational Fluid Dynamics (CFD) world. As the processing architecture of a GPGPU is fundamentally different to a CPU, alongside arises a need to optimise and accelerate solver performance specifically on GPGPUs. While certain processes become less expensive as a result of GPU acceleration, certain other processes which were relatively quick on the CPU, may become dominant portions of the overall run time on GPU. For unsteady CFD problems involving Rigid Body Motion (RBM), the costs of boundary interface computations and the maintenance of a dynamic framework for sliding meshes can oftentimes become one such performance bottleneck on GPGPUs.

In this paper, Boundary Interface Caching (BIC) is presented as a method to accelerate overall solver performance by significantly cutting down on these over-head processing costs for sliding mesh / moving mesh CFD simulations. The fundamental methodology of BIC is firstly explained, following which the Simcenter STAR-CCM+ ® solver is used to demonstrate the benefits of boundary caching on two industrial use cases: (1) an external aerodynamics simulation of a car, and (2) an acoustic simulation of a HVAC (Heating, Ventilation, Air-conditioning) fan.

Both simulations are first run without BIC to establish a baseline. They are subsequently run with BIC, and this is done on both CPU and GPU architecture. The resulting convergence behaviour, accuracy of solution, and solver performance are presented, compared and contrasted in this paper.

For Simcenter STAR-CCM+ simulations with RBM, it is demonstrated that our proprietary boundary caching algorithm provides significant improvements in overall solver time on GPGPUs, whilst maintaining similar levels of accuracy as the traditional CPU based simulations.

Nomenclature

Acronyms

BIC	Boundary Interface Caching
CFD	Computational Fluid Dynamics
CFR	Cell to Face Ratio
C_D	Coefficient of Drag
C_L	Coefficient of Lift
CPU	Central Processing Unit
DES	Detached Eddy Simulation
DRM	Direct Rotating Motion
FV	Finite Volume
GPGPU	General Purpose Graphical Processing Unit
GUI	Graphical User Interface
HVAC	Heating, Ventilation, Air-conditioning
IT	Implicit Tree
LES	Large Eddy Simulation
OASPL	Over All Sound Pressure Level
PDE	Partial Differential Equation
RBM	Rigid Body Motion

1. Introduction

Sliding meshes are widely used in unsteady CFD simulations pertaining to the automotive and aerospace domains where rigid body motion needs to be modelled accurately. For details on how finite volume discretization is modified in the context of RBM, see [1] for a section on "moving grids". Within Simcenter STAR-CCM+ ®, a sliding mesh interface is an internal boundary interface which connects two regions rotating differently [2]. The interface is updated with each time-step to exchange data between the two regions. For an unsteady CFD simulation, the time consumed in a time-step typically consists of three parts: a 'pre-step' needed to, amongst other things, re-calculate the interface intersection due to the changed boundary configuration; a 'solver-step' needed to solve the physics (typically flow and energy) for that time-step; and a 'post-step' needed to handle any post processing set-up by the user.

If we are looking to reduce the overall run time for a typical industrial CFD simulation, there is limited scope for reducing the time spent on the 'solver-step' and 'post-step'. The time spent in the 'solver-step' depends on two factors; the complexity of the physics employed in the simulation (which is user-defined) and the efficiency of the physics solvers on the processing architecture (CPU or GPU). For general physics set-ups concerning RBM the solvers are already well optimised on CPU in most widely used commercial CFD software and are well on their way to be (if not already) optimised on GPU architecture with its ever-growing popularity [3,4]. The 'post step' part of the time-step is user-defined and typically is standard for an industrial CFD simulation and hence also unavoidable.

This brings us to the 'pre-step', which if optimised can deliver huge time savings in the overall run time of an unsteady CFD simulation. As an example, let us consider an unsteady external aerodynamic simulation of an automobile, where it is common practice (in the automotive industry) to prescribe up to ~10 internal iterations per time-step, and a minimum of ~360 time-steps per rotation of the rigid body (typically a wheel). This is to ensure that each degree of the rotation is associated with a time-step which generally results in the level of accuracy needed for industrial design and analysis. For acoustic CFD simulations however, the number of time-steps per rotation of the rigid body (say a fan) can go much higher (~2000) in pursuit of capturing the acoustic phenomena of interest with sufficient accuracy. Also, the typical external aerodynamic simulation in the automotive industry tends to target ~5 seconds of total physical time; this length of physical time generally corresponds to ~100 rotations of the rigid body (wheel). For acoustic simulations, this target tends to be an order of magnitude fewer rotations (of the rigid body).

Taking the above data into account, one can estimate that the total number of inner iterations in a typical industrial unsteady CFD simulation can range anywhere from ~200,000 to ~360,000. If the time spent in the 'pre-step' can be reduced even slightly for a single time-step of the simulation, this will result in a considerable reduction in overall run time.

Boundary Interface Caching (BIC) is one such way of reducing the 'pre-step' time significantly. The next section presents the methodology of BIC in detail.

2. Methodology of Boundary Interface Caching

The basic idea of the Boundary Interface Caching method is to form an inexpensive database of pre-computed boundary interface intersection solutions, to exploit the fact that the simulation will be set up to visit the same sequence of angular positions $\theta_1, \theta_2, \ldots, \theta_{n-1}$, on each revolution of a rotating body undergoing a constant time-step size and rotation rate. As long as the problem is set up to meet these requirements, the methodology of loading cached interface intersection solutions in lieu of computing new interface intersection results can drastically reduce the runtime bottleneck during the pre-step portion on each time-step, thereby accelerating transient industrial sliding mesh simulations. For the method to be optimal and efficient, the caching method should meet two characteristics: (1) it should have low memory requirements, and (2) it should be very cheap to store and load records.

In Simcenter STAR-CCM+, at the beginning of each new time-step of the simulation, rigid body motion (RBM) may be applied to some rotating bodies, resulting in a coordinate transformation on the rotating regions (subdomains). For simplicity, assume there are one or more boundary interfaces between stationary regions and rotating regions, such as the four rotating tires of an automobile. The volume meshes will slide relative to one another along a common boundary surface, such that the cells of each region will not overlap during motion, i.e. there is a sliding mesh. By choosing an arbitrary time-step size and rotation rate, the boundary interfaces between the rotating and stationary regions become non-conformal, without remeshing or other techniques. To compute the interface intersection mesh, the original faces are broken up into interface facets that are constructed to couple the cells between the rotating and the stationary regions. The intersected facets are then used to assemble the local contributions to couple the discretized mass and energy balance equations on adjacent cells between the stationary and rotating regions.

On each inner iteration of the time-steps, the assembly of the linear systems that arise from the Finite Volume (FV) discretizations of the physical equations do not require re-intersection until motion is applied at the beginning of a subsequent time-step, so an interface intersection is reused for all inner iterations. It is also notable in the FV method that the vector area and centroid of each facet are the sole metric quantities that are ultimately used by each facet, during the assembly of the linear system. More specifically, the geometry of the facets in terms of a vertex list and vertex coordinates, are first processed into abstract metric quantities before contributing to the linear system. In Simcenter STAR-CCM+, the technique of retaining the vector area and centroid of each facet is known as 'Metrics-Based' connectivity between sliding mesh regions. The metrics-based intersection technique meets the low memory requirement for the caching method. Even without BIC, the metrics-based intersector alleviates some of the "book-keeping" required by the Simcenter STAR-CCM+ framework to store a dynamic sliding mesh in a GPU environment.

Next, the problem is set up to be cheap to store and load records. Assuming the simulation will visit the same sequence of discrete angular values on each revolution of the rotating regions. On the first revolution, the metrics-based interface intersection is computed as usual, and the metric quantities of the facets are stored using the current angle of rotation as the record identifier. The interface boundary data for metric faces can be cheaply cloned into the boundary interface cache via a low-level copy. On subsequent revolutions, the current angle of rotation can be used to look up the record that contains the pre-computed metric quantities of the intersection facets.

Finally, in order to ensure that the simulation visits the same sequence of angles, we can use a simple calculation to help automate the procedure. Assume that the simulation has reached a point that the rotating regions have a given constant rotation rate, `$rotRateRPS`, in rotations per second. One free parameter is chosen by the user to give the desired temporal resolution in terms of the number of time-steps per revolution, `$nsteps`. The constant time-step size is then calculated from these other two parameters: `$dt=1.0/($nsteps*$rotRateRPS)`. This method will be advantageous when many revolutions are simulated. The maximum amount of computational savings will be the portion of the runtime that was spent computing interface intersection and related framework operations during the pre-step.

There are a few caveats to mention with the BIC method. The cache will only remain valid so long as the distributed simulation has not been repartitioned, unless the cache is also repartitioned appropriately. The cache is also valid as long as the original boundary meshes have remained unchanged, except for the application of a periodic RBM like a simple rotation. For example, dynamic mesh change events such as adaptive mesh refinement or morphing the regions would invalidate the boundary interface cache. Finally, if the sequence of angles visited upon subsequent revolutions has not been previously visited as described in the automation procedure, then the user would not obtain any cache hits or performance gains from this method.

3. Computational Hardware

Throughout this work, results from and the performance of Simcenter STAR-CCM+ on CPUs and GPUs are compared. The CPUs used are dual socket AMD EPYC 7532 ROME (32-Core, 2.4GHz) CPUs in a 64 physical cores per node configuration. The GPUs used are 40GB NVIDIA A100 SXM4 GPUs in a 4 GPU per node configuration [5]. The CPUs supporting the GPU architecture are dual socket AMD EPYC 7543 ROME (32-core, 2.4 GHz) CPUs. The Simcenter STAR-CCM+ framework utilizes GPGPU enabled kernels to assemble the discretized finite volume equations [6], as well as to obtain their solution using the AmgX linear solver from NVIDIA [7].

4. Results and Discussion

To demonstrate the impact of our proprietary Boundary Interface Caching (BIC) algorithm in Simcenter STAR-CCM+, two industrial cases simulating Rigid Body Motion (RBM) have been selected: Case-(1) an external aerodynamics simulation of the Maserati Ghibli car, and Case-(2) an acoustic simulation of a HVAC (Heating, Ventilation, Air-conditioning) fan [8].

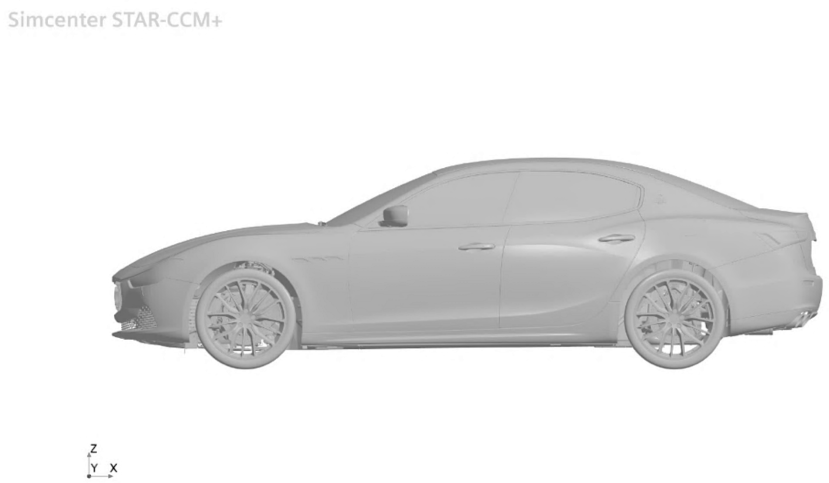

Figure 1: Geometry of Case-(1) - an external aerodynamic simulation of the Maserati Ghibli car

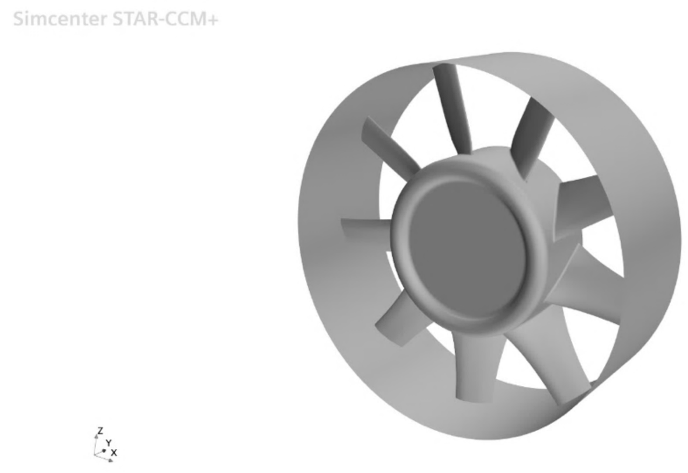

Figure 2: Geometry of Case-(2) - an acoustic simulation of a HVAC fan

a. Simulation Details

Table 1 below summarizes the salient simulation details for the Simcenter STAR-CCM+ simulations tested.

Table 1: Salient simulation details for the Simcenter STAR-CCM+ simulations tested

Simulation Name	Simulation Description	Mesh Size (No. of Cells)	Active Solvers
Case-(1)	External aerodynamics simulation of a car	~140 million	Implicit Unsteady (DES) Direct Rotating Motion (DRM) Segregated Flow Kw Turbulence IT Wall Distance (Frozen)
Case-(2)	Acoustic simulation of a HVAC fan	~26.5 million	Implicit Unsteady (LES) Rigid Body Motion (RBM) Segregated Flow Kw Turbulence PDE Wall Distance Perturbed Convective Wave

Case-(1) i.e. the external aero simulation has ~140 million cells in the core mesh and is the larger simulation amongst the two. Case-(2) i.e. the acoustics simulation contains only ~26.5 million cells in the core mesh. For simulating rotation of the rigid body, Case-(1) uses the DRM model within Simcenter STAR-CCM+ for the wheels of the car, whereas Case-(2) uses the RBM model to simulate the rotation of the HVAC fan. DRM is an alternative method to specify RBM, which allows for improved automation in the simulation setup.

Both simulations commonly deploy the following physics solvers in their respective set-ups: implicit unsteady, segregated flow and the k-ω turbulence model. Case-(1) uses a DES scheme for the implicit unsteady solver [9] while Case-(2) deploys an LES scheme [10,11]. The wall distance solver is kept frozen in Case-(1) and is set to the ‘PDE’ (Partial Differential Equation) in Case-(2). Furthermore Case-(2) being an acoustics simulation also deploys the ‘Perturbed Convective Wave’ solver alongside the remaining set of standard solvers for an industrial unsteady RBM simulation.

b. Test Set-up and Run Details

Both simulations are first run for a steady state solution without motion to establish the flow field. Using this steady state solution as a starting point, the unsteady CFD with RBM is then initiated. Case-(1) contains 4 sliding mesh interfaces, one for each wheel of the car while Case-(2) contains just 1 sliding mesh interface corresponding to the HVAC fan. As indicated in the methodology section (section 2) of the current paper, the sliding interfaces for both simulations are set-up to use the metrics-based intersector.

For the unsteady phase of the simulations, to ensure BIC works as intended, it is necessary that the same sequence of angles is visited in each rotation. This is achieved simply by tweaking the time-step size as described in the methodology section (section 2) of the current paper. Table 2 below lists the salient parameters set-up for both simulations in this context.

Table 2: Salient parameters set-up for BIC in the Simcenter STAR-CCM+ simulations tested

Simulation Name	Rotation Rate of Sliding Mesh Interface (RPS)	No. of Time-Steps Per Rotation	Max. Internal Iterations Per Time-Step	Time-Step Size Used (Secs) $dt=1.0/($nsteps*$rotRateRPS)
Case-(1)	17.51	228	5	2.504 E-4
Case-(2)	24.77	2000	5	2.019 E-5

The Graphical User Interface (GUI) for BIC within Simcenter STAR-CCM+ is set-up very intuitively, with a master control provided in the ‘Interfaces’ node of the simulation to switch ON/OFF BIC for the overall simulation [2]. A subsequent ‘Boundary Interface Caching’ node under ‘Physics Conditions’ of each individual sliding mesh interface provides specific control over the BIC properties used for that specific interface. Figure 3 below shows an example of the BIC GUI from Case-(2). Note that the default value for ‘Maximum Steps’ for BIC is 1000, but this has been changed to 2000 in Figure 3, based on the target number of time-steps per rotation used for Case-(2) (Refer Table 2 above).

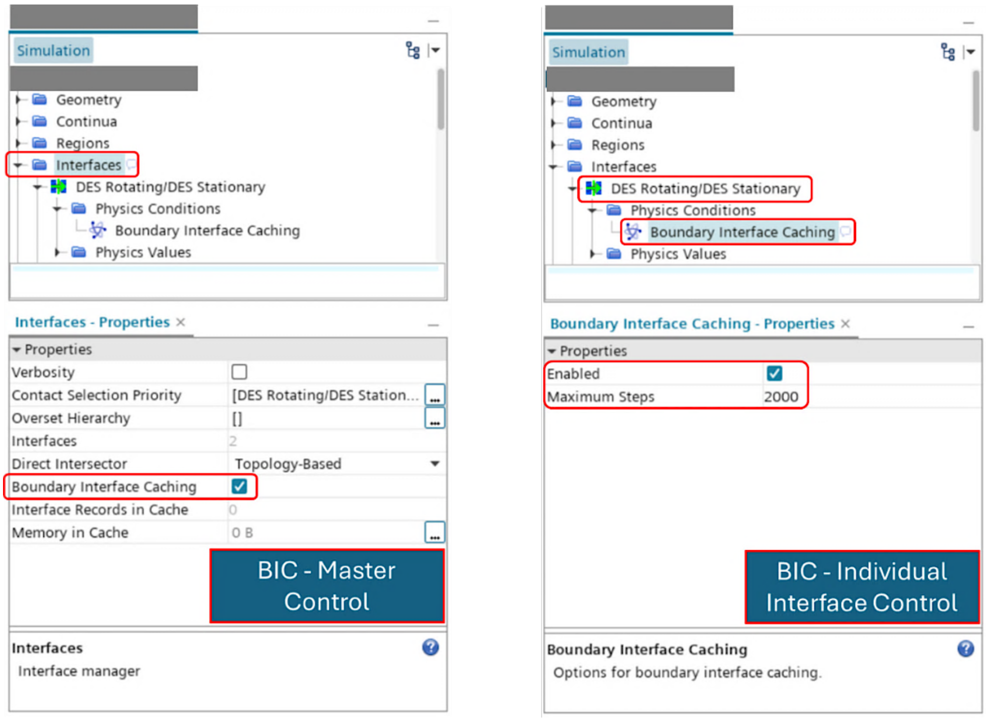

Figure 3: Boundary Interface Caching (BIC) GUI in Simcenter STAR-CCM+ (from Case-(2))

Coming to the run details, both simulations are run with and without BIC on CPUs and GPUs located on a Siemens high performance cluster in Ireland (see section 3 for full details of the computational hardware used). Table 3 below provides the run details for each of the 4 runs of both simulations.

Table 3: Run details for the Simcenter STAR-CCM+ simulations tested

Simulation Name	Run Configuration	Run Type	No. of Processors (NP)
Case-(1)	Without BIC	CPU	640
	With BIC		
	Without BIC	GPU	8
	With BIC		
Case-(2)	Without BIC	CPU	1024
	With BIC		
	Without BIC	GPU	8
	With BIC		

c. Simulation Results

The resulting solution from the GPU runs with and without BIC are compared in this section.

i. Case-(1)

Figure 4 below shows a comparison of the time averaged velocity field around the car in Case-(1), after ~5.5 seconds of physical time (~89 revolutions), which is very similar with and without BIC.

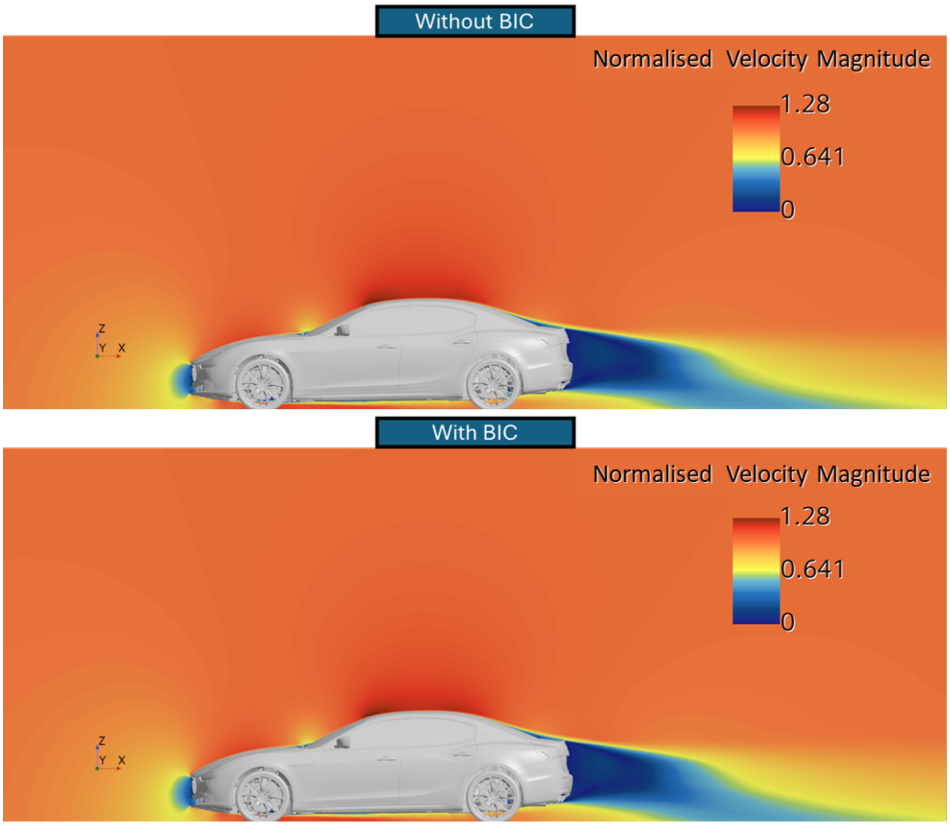

Figure 4: Normalised velocity contour comparison (Case-(1))

Figure 5 below shows a comparison of the instantaneous coefficient of drag (C_D) as a function of physical simulation time, with and without BIC for Case-(1). The difference between two major gridlines on the y-axis in Figure 5 is 10 counts (0.001 C_D). As expected, there is some variability seen in instantaneous C_D values between the two runs due to the transient nature of the simulation. However, we can see that the time averaged C_D values for both runs fall very close to each other (within 3 counts). Hence, for all practical purposes, the C_D predictions are very similar with and without BIC for Case-(1).

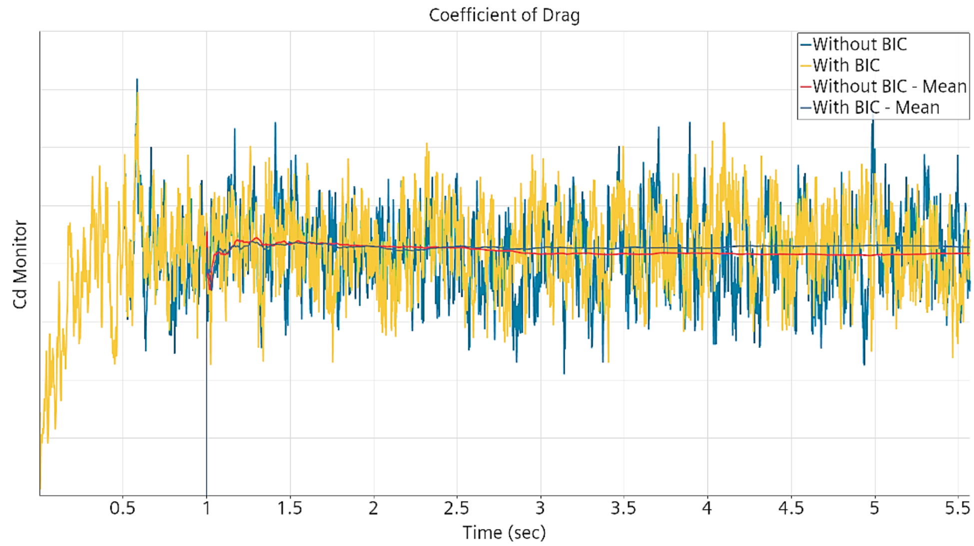

Figure 5: Coefficient of drag (Cd) comparison (Case-(1))

Figure 6 below shows a comparison of the instantaneous coefficient of lift (C_L) as a function of physical simulation time, with and without BIC for Case-(1). The difference between two major gridlines on the y-axis in Figure 6 is 10 counts (0.001 C_L). As expected, there is some variability seen in instantaneous C_L values between the two runs due to the transient nature of the simulation. However, we can see that the time averaged C_L values for both runs fall very close to each other (within 5 counts). Hence, for all practical purposes, the C_L predictions are very similar with and without BIC for Case-(1).

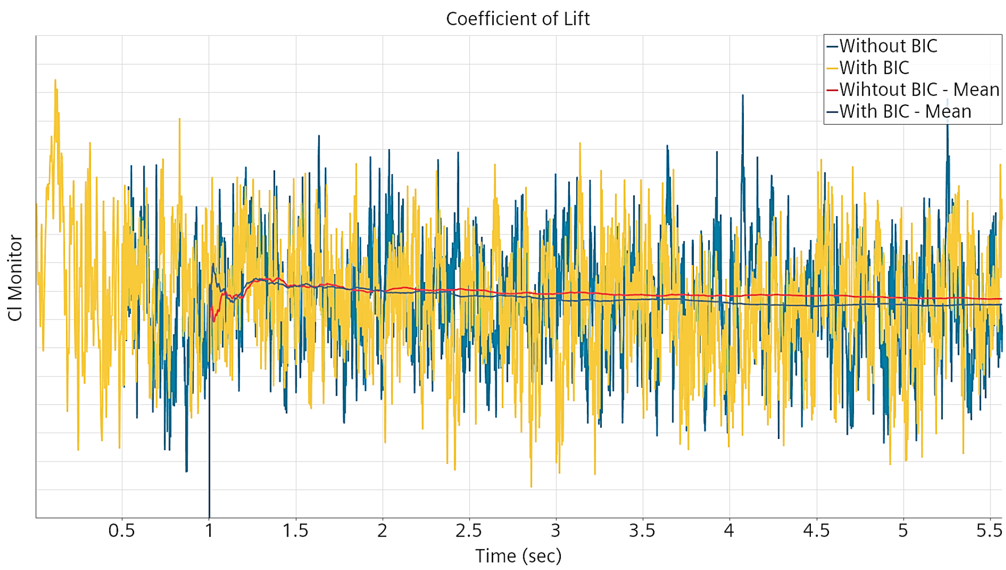

Figure 6: Coefficient of lift (Cl) comparison (Case-(1))

To sum up, from the above comparisons it is very clear the resulting external aerodynamic solution for Case-(1) is very similar with or without BIC on GPU. This is also true on CPU.

ii. Case-(2)

Figure 7 below shows a comparison of the instantaneous acoustic pressure field around the fan in Case-(2), which is very similar with and without BIC.

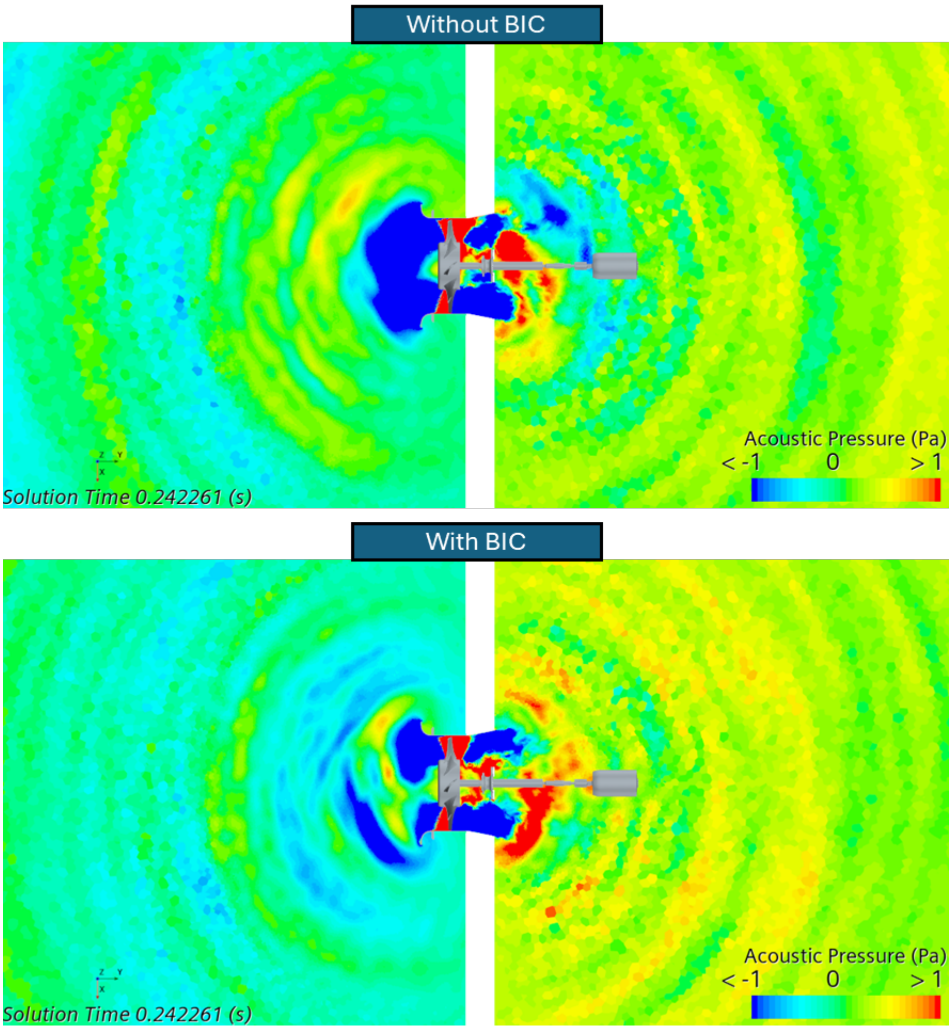

Figure 7: Instantaneous acoustic pressure field comparison (Case-(2))

Figure 8 below shows the position of a microphone (Mic 4) whose pressure perturbation data has been analysed and plotted in Figure 9.

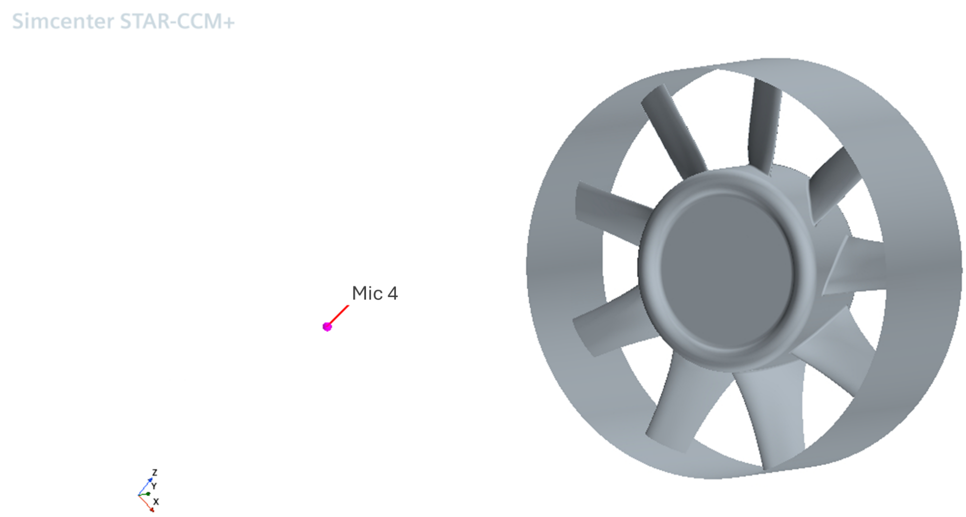

Figure 8: Mic 4 position (Case-(2))

Figure 9 below shows a comparison of the A-weighted sound pressure level spectrum from Mic 4 resulting from the runs with and without BIC. The test data from [8] for the same operating point is also plotted alongside for reference. As can be seen from Figure 9, for all practical purposes the sound pressure level spectrum with and without BIC is the same, barring some expected transient variability.

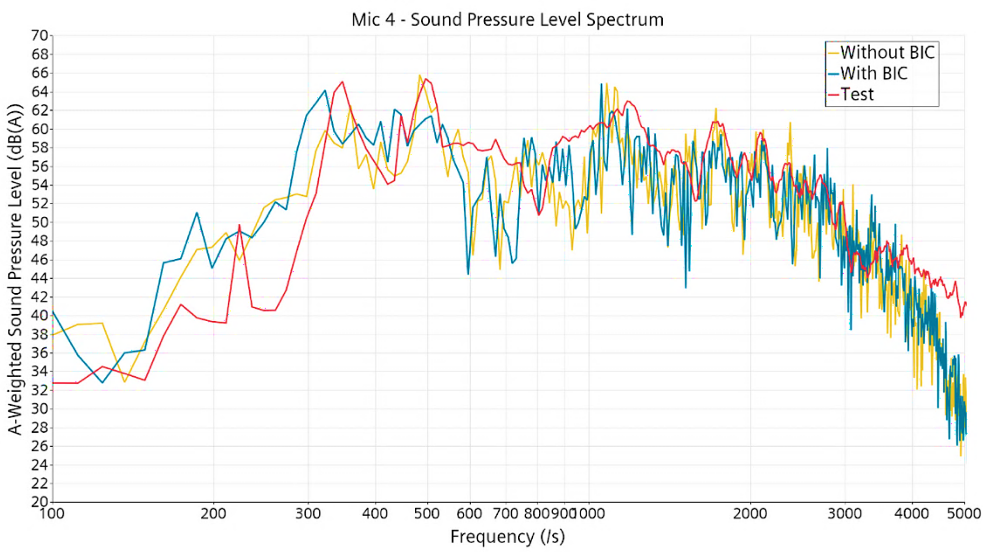

Figure 9: Comparison of sound pressure level spectrum from Mic 4 (Case-(2))

Table 4 below shows a comparison of the Over All Sound Pressure Level or OASPL (calculated as 20Log10((RMS^2)/2E-5) of Mic 4 integrated over the whole spectrum shown in Figure 9 above, for the runs with and without BIC.

The OASPL calculated from the Test data in [8], for the same operating point for Mic 4, is also provided in the Table 4 for reference. The difference between the OASPL values with and without BIC is quite small at ~0.9 dB. This difference falls well within the band of experimental variation for this set-up and is also much smaller than the difference between the simulation result (without BIC) and the actual test data itself (~2.4 dB).

Table 4: Overall sound pressure level comparison (Case-(2))

Simulation Name	Boundary Interface Caching	Mic 4 – Overall Sound Pressure Level (dB)
Case-(2)	Without BIC	81.6
	With BIC	82.5
Test Data [8]		84.0

To sum up, from the above comparisons it is very clear that the resulting acoustic solution for Case-(2) is very similar with or without BIC on GPU. This is also true on CPU.

d. Run Time Profiling and Memory Consumption

It is important to note that the first rotation of the rigid body in both Simcenter STAR-CCM+ simulations is used to store the boundary interface cache data, which is then loaded and used for all subsequent rotations. For the purposes of run time comparison, both simulations are profiled from the second rotation onwards for the breakdown of the pre-step times and the overall run times with and without BIC. Case-(1) i.e. the external aerodynamic simulation is sampled for ~10 rotations (after the first rotation) of the rigid bodies (wheels), while Case-(2) i.e. the acoustic simulation is sampled for ~5 rotations (after the first rotation) of the rigid body (HVAC fan). Figures 10 and 11 below summarize the run time profiling results for the CPU and GPU runs respectively, for both the Simcenter STAR-CCM+ simulations tested.

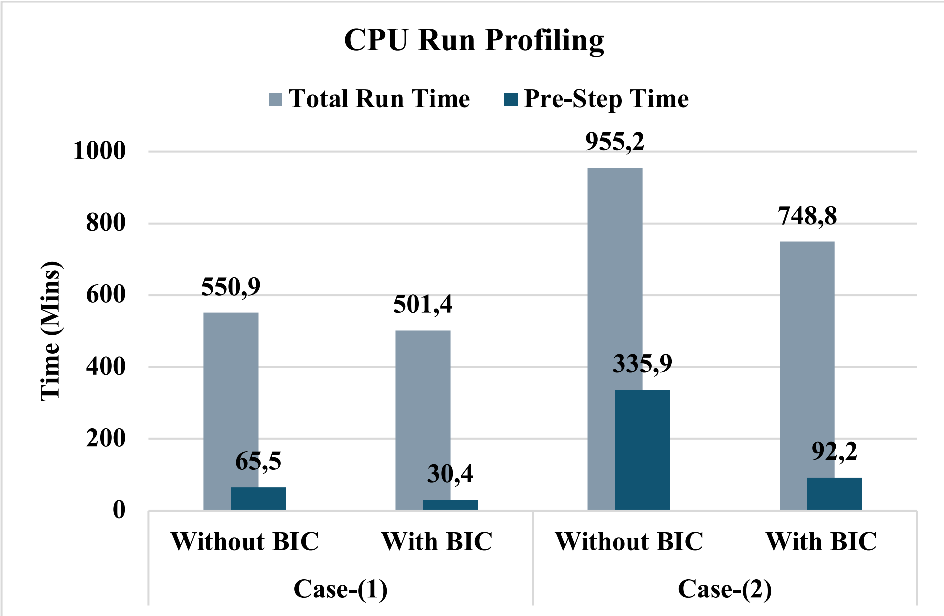

Figure 10: CPU run time profiling results for the Simcenter STAR-CCM+ simulations tested

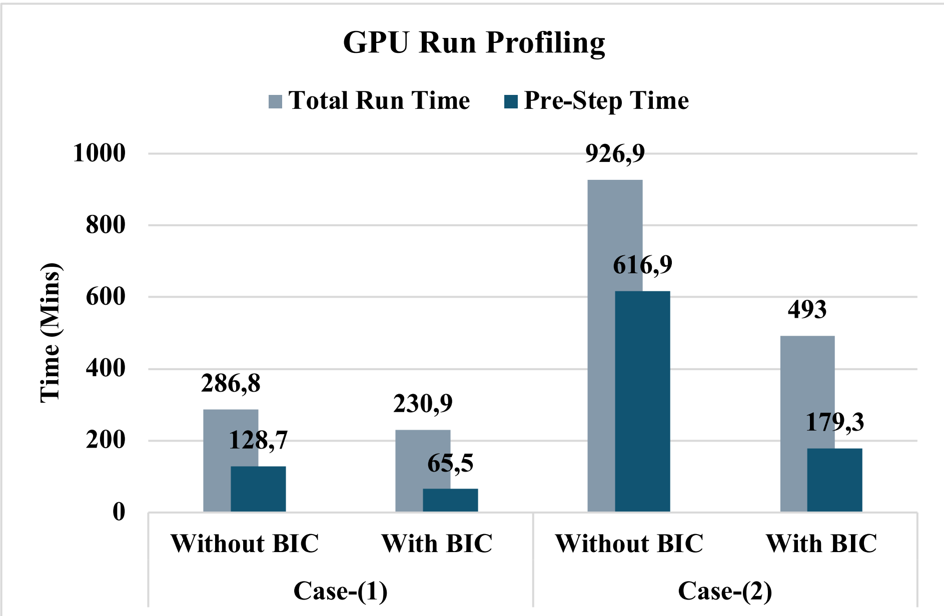

Figure 11: GPU run time profiling results for the Simcenter STAR-CCM+ simulations tested

Upon activating BIC, the time spent in boundary interface intersection at each time-step reduces to nearly zero, and instead some time is now spent on loading the boundary interface data from the cache. As the cache load times are significantly smaller than the intersection times, what results is a considerable reduction in the pre-step time at each time-step.

As can be seen from Figures 10 and 11, the pre-step times (and consequently the overall run times) for both Simcenter STAR-CCM+ simulations tested are significantly lower with BIC than without, both on CPU and GPU. The pre-step times without BIC on the GPU are generally a bigger portion of the overall run time than on the CPU, as can be seen from Figures 10 and 11. Consequently, BIC delivers a bigger reduction in pre-step time (and the overall run time) on GPU than on CPU.

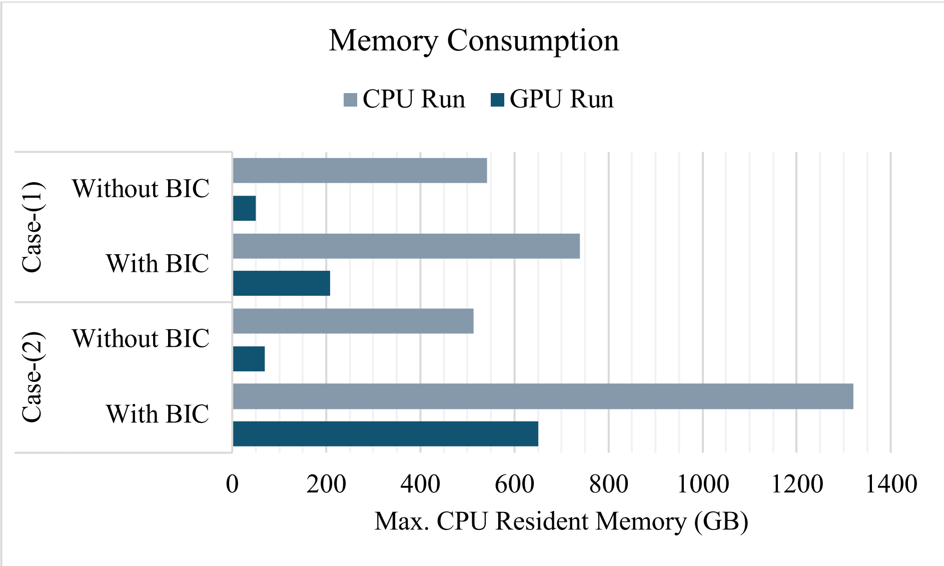

Figure 12: CPU memory consumption statistics for the Simcenter STAR-CCM+ simulations tested

Figure 12 above records the memory consumption statistics for all 4 runs of both the Simcenter STAR-CCM+ simulations tested. The maximum CPU memory consumption increases upon activating BIC for both simulations. This is true for both the CPU run as well as the GPU run, as the cache data is stored and retrieved from the CPU memory in both cases. In this regard, it should be noted that the maximum GPU memory consumption for the GPU runs for Case-(1) and Case-(2) remains at a steady ~210 GB and ~95 GB respectively, with or without BIC.

With BIC, the maximum CPU memory consumption increases by a bigger margin for the CPU run than the GPU run, primarily due to the higher partitioning and hence higher halo cell counts encountered in the CPU run (halo cells are cells shared between processors during parallel processing). However, the increase in the maximum CPU memory consumption was not found to hamper the CPU / GPU run progress in any way for either of the simulation tested on the computational hardware (see section 3 for details).

e. Performance Benefit

Figure 13 below summarises the benefits to overall run time, resulting from boundary caching, on CPU and GPU for both the Simcenter STAR-CCM+ simulations tested. The performance benefits from BIC, in terms of overall run time reduction, are much larger on the GPU (as compared to CPU) for both the simulations tested. With BIC, Case-(2)'s run time is nearly cut down in half (~47% reduction). The overall run time reduction resulting from BIC on CPU is roughly half that observed on GPU, and this is consistent for both the simulations tested.

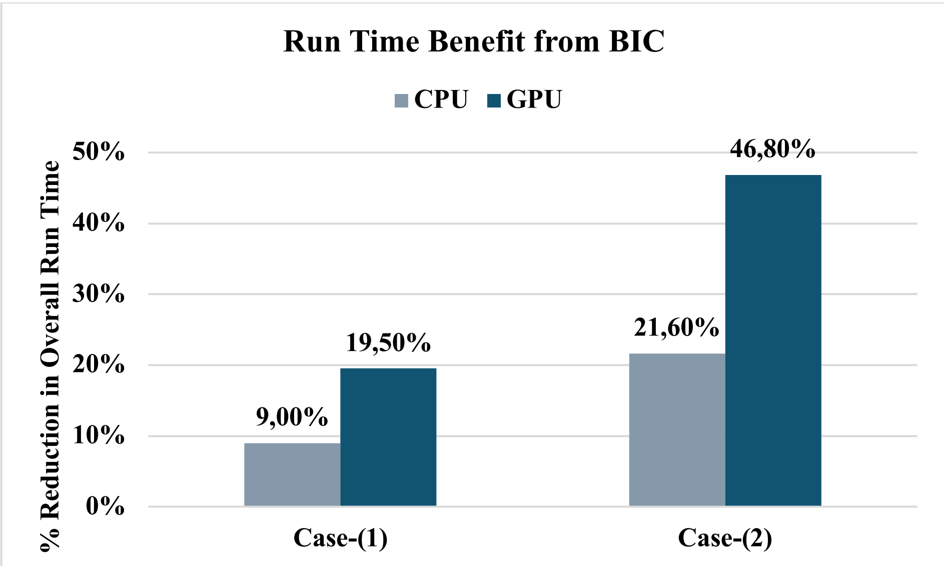

Figure 13: Run time benefit from BIC

Figure 14 below summarise the benefits in the CPU equivalence which result from boundary caching on CPU and GPU, for both the Simcenter STAR-CCM+ simulations tested. BIC helps increase the CPU equivalence as the overall run time benefits are much higher on GPU than on CPU. Case-(2)'s CPU equivalence jumps up by almost ~47.5% as a result of implementing BIC. It should be noted that, to be fair, the CPU equivalences for the runs with BIC are calculated independent to the runs without BIC, for both simulations.

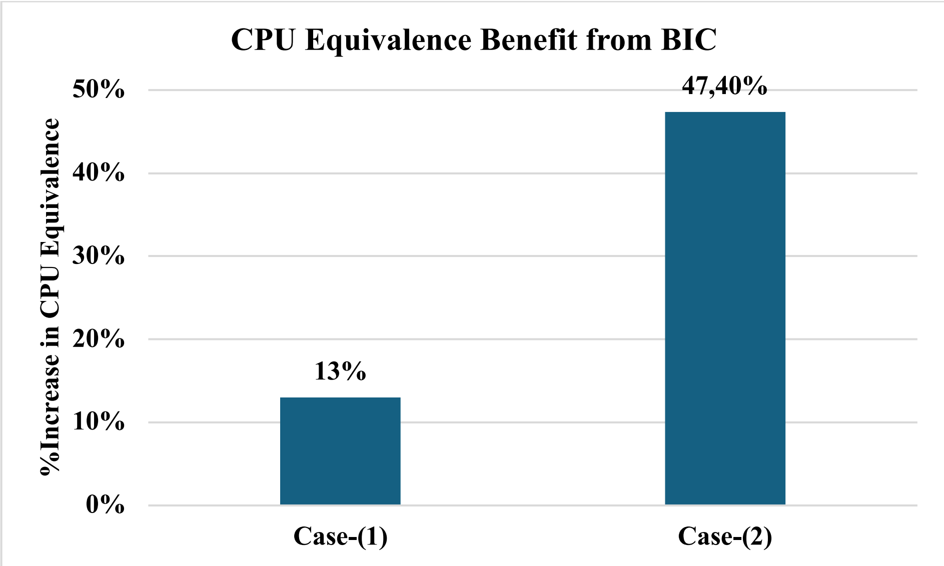

Figure 14: CPU equivalence benefit from BIC

Table 5 below records the Cell to Face Ratio or CFR for both the Simcenter STAR-CCM+ simulations tested. CFR is defined as the ratio of the number of cells in the core mesh to the number of faces on the sliding mesh interfaces of an unsteady CFD simulation with RBM.

Table 5: Cell to face ratio for the Simcenter STAR-CCM+ simulations tested

Simulation Name	Cell to Face Ratio (CFR)		
	No. of cells in core mesh	No. of faces on sliding mesh interfaces	Cell to Face Ratio
Case-(1)	140,161,300	2,171,172	65
Case-(2)	26,562,950	924,729	29

The performance benefits realised from BIC are inversely proportional to the CFR of the Simcenter STAR-CCM+ simulation. The higher the number of faces on the sliding mesh interface(s), in comparison to the total cells in the core mesh of the simulation, the higher is the time spent on boundary interface intersection in the pre-step, and hence higher is the time saved upon implementation of BIC. It is also good to note at this point that the benefits realised from BIC are also inversely proportional to the number of 'max. internal iterations per timestep' used in the stopping criteria of the unsteady Simcenter STAR-CCM+ simulation. This happens because the higher this number, the lesser is the proportion of the pre-step time in the overall run time.

And finally, while the % run time benefits from BIC are impressive, it is important to note that the absolute run time benefits from BIC increase with each new rotation of the rigid body. This would provide huge time savings for industrial sliding mesh simulations which typically run for many days (and many more rotations) at a time.

5. Conclusions

In this paper, Boundary Interface Caching (BIC) is presented as a method to accelerate overall solver performance by significantly cutting down on over-head processing costs for sliding mesh CFD simulations. The methodology of BIC is to effectively store boundary interface intersection information in the first rotation of a rigid body, and to load this information for all subsequent rotations, thereby cutting down the pre-step time of a simulation significantly. For BIC to work as intended it is mandatory that the simulation visits the same sequence of angles in every rotation, a condition that can be easily achieved by slightly tweaking the time-step size (see section 2 for details).

In order to demonstrate the impact of our proprietary BIC algorithm in Simcenter STAR-CCM+, two industrial cases simulating Rigid Body Motion (RBM) are selected: Case-(1) an external aerodynamics simulation of the Maserati Ghibli car, and Case-(2) an acoustic simulation of a HVAC (Heating, Ventilation, Air-conditioning) fan [8]. To start with, both simulations are run for a steady state solution without RBM. Subsequently, an unsteady CFD simulation with RBM is initiated, with the relevant BIC parameters set-up correctly. Both simulations are run with and without BIC on CPUs and GPUs located on a Siemens high performance cluster in Ireland (see section 3 for details).

The solutions resulting from the runs with and without BIC are compared. After analysing the relevant instantaneous and time-averaged contours and plots, it is concluded that the solutions resulting from the runs with and without BIC are very similar for both the Simcenter STAR-CCM+ simulations tested.

The GPU and CPU runs for both the simulations are also profiled to understand the impact of BIC on run time. While BIC results in a run time benefit on CPU as well as GPU, the benefits on GPU out-weigh the benefits on CPU considerably for both the Simcenter STAR-CCM+ simulations tested, owing to the larger pre-step times encountered on GPU (without BIC). BIC delivers a staggering ~47% reduction in overall run time on GPU for the acoustic simulation tested (Case-(2)) and a similar ~48% increase in CPU equivalence. The performance benefits from BIC are found to be inversely proportional to the Cell to Face Ratio (CFR) of a Simcenter STAR-CCM+ simulation (see section 4e for details).

Finally, it is concluded that BIC provides significant benefits in overall run times, especially on GPGPUs, whilst maintaining similar levels of accuracy as the traditional CPU based simulations.

6. Acknowledgements

The authors gratefully acknowledge (in alphabetical order) Alistair Brown, Eberhard Schreck, Jeremy Dahan, Lars Erbig, Liam McManus, Navid Hermidas, Ritesh Bafna and Simone Landi from Siemens Digital Industry Software for their invaluable guidance throughout this work and their timely review support for this paper.

7. Permission for Use

8. References

[1] J. H. Ferziger, M. Perić, and R. L. Street, *Computational Methods for Fluid Dynamics*. Cham: Springer International Publishing, 2020. doi: https://doi.org/10.1007/978-3-319-99693-6

[2] Siemens Digital Industries Software, "Simcenter STAR-CCM+ User Guide v. 2502", 2025. https://support.sw.siemens.com/en-US/product/226870983

[3] L. McManus, G. Goldin, Y. Zhang, I. Veljkovic, S. Sadasivuni, and K. Liu, "GPU Accelerated LES-FGM Modelling for Industrial Combustion Applications," *Volume 3A: Combustion, Fuels, and Emissions*, Jun. 2024, doi: https://doi.org/10.1115/gt2024-122041

[4] "The Computational Fluid Dynamics Revolution Driven by GPU Acceleration," *NVIDIA Technical Blog*, May 26, 2022. https://developer.nvidia.com/blog/computational-fluid-dynamics-revolution-driven-by-gpu-acceleration/

[5] "NVIDIA A100 GPUs Power the Modern Data Center," *NVIDIA*. https://www.nvidia.com/en-gb/data-center/a100/

[6] "HPC SDK | NVIDIA," *NVIDIA Developer*, Mar. 11, 2020. https://developer.nvidia.com/hpc-sdk

[7] M. Naumov *et al.*, "AmgX: A Library for GPU Accelerated Algebraic Multigrid and Preconditioned Iterative Methods," *SIAM Journal on Scientific Computing*, vol. 37, no. 5, pp. S602–S626, Jan. 2015, doi: https://doi.org/10.1137/140980260

[8] “Acoustics involving heterogeneous and moving fluids | TU Wien,” *www.tuwien.at*. https://www.tuwien.at/en/mwbw/mec/e325-03-research-unit-of-measurement-and-actuator-technology/eaa-benchmarks/benchmarks/acoustics-involving-heterogeneous-and-moving-fluids

[9] P. R. Spalart, “Detached-Eddy Simulation,” *Annual Review of Fluid Mechanics*, vol. 41, no. 1, pp. 181–202, Jan. 2009, doi: https://doi.org/10.1146/annurev.fluid.010908.165130

[10] J. SMAGORINSKY, “GENERAL CIRCULATION EXPERIMENTS WITH THE PRIMITIVE EQUATIONS,” *Monthly Weather Review*, vol. 91, no. 3, pp. 99–164, Mar. 1963, doi: https://doi.org/10.1175/1520-0493(1963)091%3C0099:gcewtp%3E2.3.co;2

[11] J. W. Deardorff, “A numerical study of three-dimensional turbulent channel flow at large Reynolds numbers,” *Journal of Fluid Mechanics*, vol. 41, no. 2, pp. 453–480, Apr. 1970, doi: https://doi.org/10.1017/s0022112070000691