Parallel Computing for Trajectory Prediction of Aircraft

†Momoha Nishimura[1], Masashi Yamakawa[1], Ryuta Sakashita[1], Shinichi Asao[2] and Mitsuru Tanaka[1]

[1]Department of Mechanical and System Engineering, Kyoto Institute of Technology, JAPAN.
[2]Department of Mechanical Engineering, College of Industrial Technology, JAPAN.

†Corresponding author: d7821007@edu.kit.ac.jp

Abstract

GPU parallel computing was applied to a trajectory prediction of an aircraft. An aerobatic maneuver was simulated by a coupled method of 6-DOF motion and MCD method. Because the simulation contained a complex interaction of grid movement and flow dynamics and thus it was quite expensive, the acceleration by GPU was attempted to demonstrate its ability compared to CPU parallelization. The GPGPU code was constructed by OpenACC because of the directive-based programming. The GPU computing accomplished a remarkable speedup, which confirmed that GPGPU is useful for the acceleration of this system.

Keywords: CFD, Parallel Computing, GPGPU, OpenACC, Flight Simulation.

Introduction

The numerical flight simulation of an aircraft is an important problem in Computational Fluid Dynamics (CFD) for aerospace. With this system called the Digital Flight [1], aerodynamic coefficient of an airplane can be safely calculated even in a risky flight without real aircrafts. Moreover, a risk of stall can be predicted when the flight is simulated within a real time.

To achieve the flight simulation based on physics, the authors have proposed the Moving Computational Domain (MCD) method [2], which is one of the moving grid methods based on the Moving-grid Finite-Volume method [3]-[5]. In this method, the computational domain itself with an aircraft inside moves by following a movement of the airplane, therefore any restrictions of the computational domain for three-dimensional space can be removed [6][7]. Recently, we have integrated the six-degrees-of-freedom (6-DOF) motion to this CFD method [8][9], and simulated unconstrained motions of objects influenced by fluid and the motion. As applications of this coupled method to a flight simulation, various aerobatics of an airplane were computed. The rotation of the propeller and the moving control surface were installed in this system as well to simulate flight as if pilot operated the aircraft.

In this paper, parallel computing on graphics processing unit (GPU) is introduced towards the calculation within an actual time. Although OpenMP or MPI is generally adopted to shorten the calculation time [10], general-purpose computing on GPU (GPGPU) has been recently in the spotlight. GPU has a number of cores, thus GPUs could calculate much faster than CPU. Although major examples for GPGPU are NVIDIA's CUDA or OpenCL, much time and labor is required for code rewriting. Therefore, we employed OpenACC which can accelerate using compiler directives for our first coding on GPU to see whether GPGPU can help for faster computing compared to OpenMP.

Numerical Scheme

Flow Solver

The governing equations are the three-dimensional Euler equations for compressible flow written in the conservation form (1),(2) as follows:

$$\frac{\partial \mathbf{q}}{\partial t} + \frac{\partial \mathbf{E}}{\partial x} + \frac{\partial \mathbf{F}}{\partial y} + \frac{\partial \mathbf{G}}{\partial z} = 0, \tag{1}$$

$$\mathbf{q} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ e \end{bmatrix}, \mathbf{E} = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uw \\ u(e+p) \end{bmatrix}, \mathbf{F} = \begin{bmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ \rho vw \\ v(e+p) \end{bmatrix}, \mathbf{G} = \begin{bmatrix} \rho w \\ \rho uw \\ \rho vw \\ \rho w^2 + p \\ w(e+p) \end{bmatrix}. \tag{2}$$

The equations are discretized by the Moving-grid Finite-Volume method with four-dimensional control volume combined with time and space. The inviscid flux vectors are estimated by Roe's flux difference splitting [11] at the interfaces. MUSCL approach and Venkatakrishnan limiter [12] are employed to provide second order accuracy in space. The 2-stage rational Runge-Kutta method is applied as a time stepping scheme.

6-DOF Motion

Because the aircraft is treated as a rigid body in this paper, 6-DOF equations of motion govern the flight of the aircraft. The Newton's equation of motion (3) is applied to the translation of the mass center, and the Euler's rotation equation (4) is applied to the rotation in body axes. The rotational motion is calculated in the body-fixed axis subscripted with B. Here quaternion [13] is used to avoid gimbal-lock. 6-DOF equations are integrated into the inner iteration of flow solver as a strong coupling method [14].

$$\frac{d\mathbf{p}}{dt} = \mathbf{f} \tag{3}$$

$$\frac{d\mathbf{L}_B}{dt} + \boldsymbol{\omega}_B \times \mathbf{L}_B = \mathbf{N}_B \tag{4}$$

$$\mathbf{p} = \begin{bmatrix} m\dot{r}_x \\ m\dot{r}_y \\ m\dot{r}_z \end{bmatrix}, \mathbf{f} = \begin{bmatrix} f_x \\ f_y \\ f_z - mg \end{bmatrix}, \mathbf{L}_B = \begin{bmatrix} I_{x_B}\omega_{x_B} \\ I_{y_B}\omega_{y_B} \\ I_{z_B}\omega_{z_B} \end{bmatrix}, \boldsymbol{\omega}_B = \begin{bmatrix} \omega_{x_B} \\ \omega_{y_B} \\ \omega_{z_B} \end{bmatrix}, \mathbf{N}_B = \begin{bmatrix} N_{x_B} \\ N_{y_B} \\ N_{z_B} \end{bmatrix} \tag{5}$$

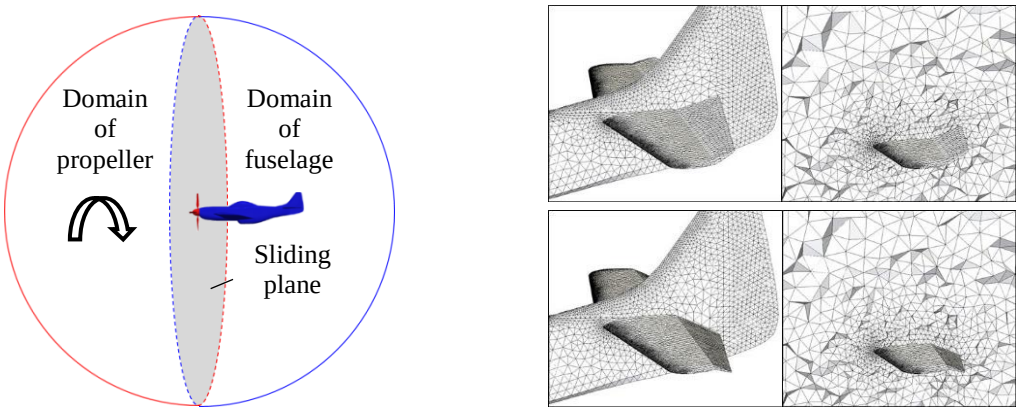
Grid Movement and Deformation

MCD Method

When it comes to the movement of the whole grid in a large area, Moving Computational Domain (MCD) method is applied. The computational domain itself with an object inside moves by following the moving aircraft.

Grid Deformation

The rotation of the propeller and the motion of the control surface are installed by sliding-mesh approach and tension-torsion spring analogy [15] (Fig. 1(a)). To apply the sliding mesh technique, computational domain is divided into two domains, one is the fuselage domain and the other is the propeller domain. The conservative quantities are interpolated at the interface where tetrahedral grids overlap each other. When the control surface is moved, grids are deformed by tension-torsion spring analogy (Fig. 1(b)).



(a) Sliding-mesh method **(b) Tension-torsion spring analogy**

Figure 1. Grid movement and deformation

Application

Calculating Condition

P-51 propeller aircraft model with 1,612,350 unstructured grid points are generated by MEGG3D [16][17] are shown in Fig. 2. The center of gravity is assumed to be located in 25%MAC (Mean Aerodynamic Chord), and the moment of inertia is generated by the engine, the fuel tank and the skin of the airplane.

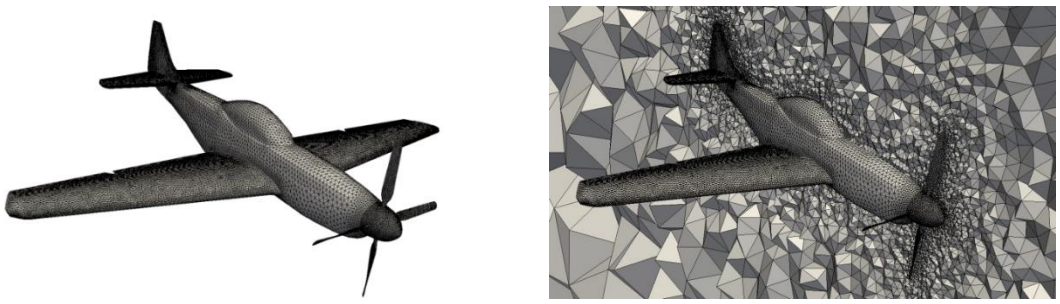
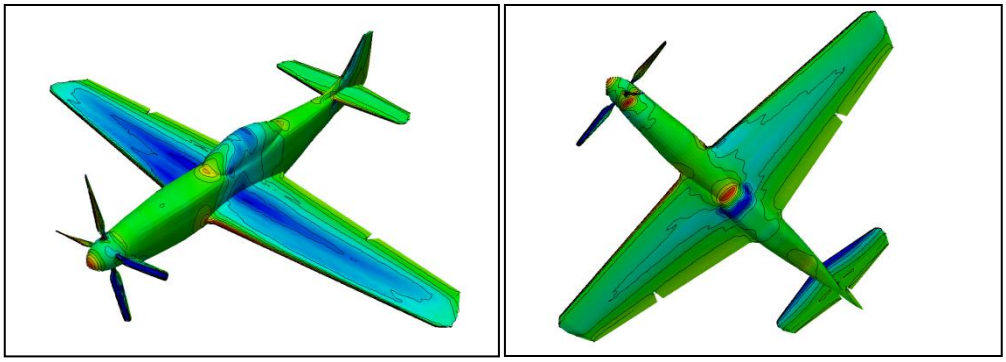


Figure 2. Computational grid

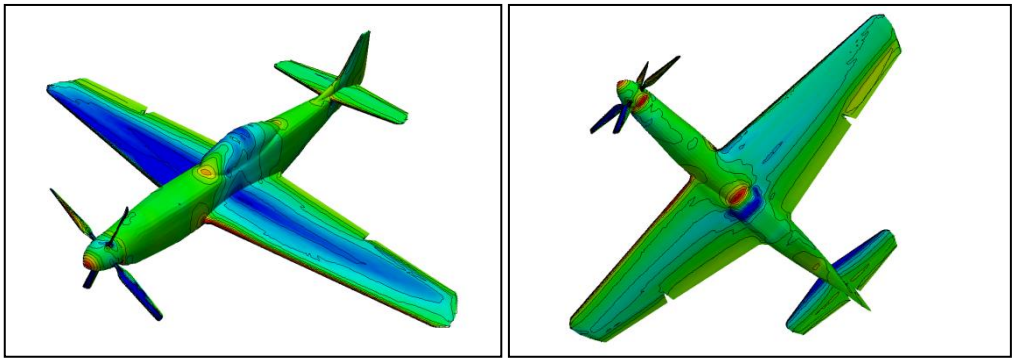
The straight flight, two clockwise and two counterclockwise aileron roll are simulated with this system. At first, the aircraft performs straight flight to avoid the initial turbulence at $V = 0.45$, where the speed of sound is 1.0. After that, to complete the clockwise rolling, the right aileron is manipulated up to 10 degrees, and left aileron is manipulated down to 10 degrees. The rudder is also controlled to avoid adverse yaw. In counter-clockwise rolling, the ailerons and rudder are operated oppositely.

Results

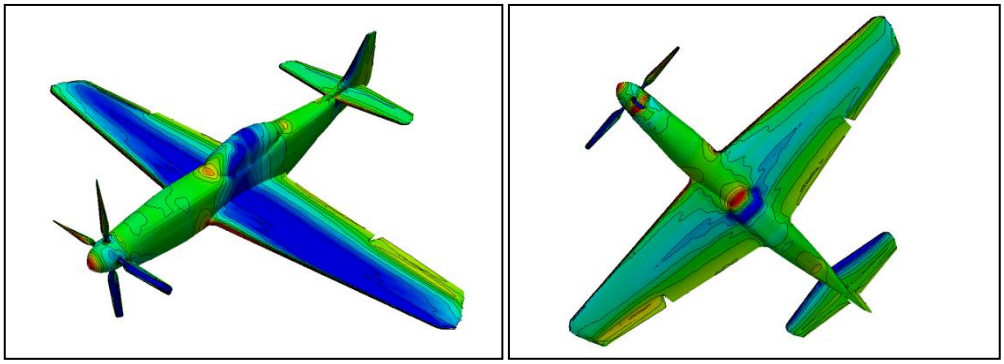
The surface pressure distributions at the forward straight flight, clockwise rolling and counter-clockwise rolling are shown in Fig. 3. It can be seen in Fig. 3(a) that the rotating propeller generated the vortex and pressure distribution on main wing became asymmetry. After ailerons are operated, the pressure distribution at the main wing changed significantly, as seen in Fig. 3(b) and (c). The pressure at the bottom side of the main wing is lower than the upper side, which yields the rolling moment to complete the aileron roll.



(a) Forward flight



(b) Clockwise flight



(c) Counterclockwise flight

Figure 3. Pressure contours in the aileron roll

GPU Parallel Computation

For high-speed calculations, we constructed the GPU calculation code of this system by OpenACC. The computer for GPU computing (PC1) has CPU of Intel® Core™ i7-3930K Processor and GPU of GIGABYTE6.0 V-NTITANBLKGHZ-6GD-B GEFORCE GTX TITAN BLACK. The operating system is Windows7 64bit and the compiler is PGI Accelerator Fortran Workstation. For comparison of the calculating speed, this system is calculated only with CPU (PC2). The computer, which has the highest performance for CPU we could use, has Intel® Core™ i7-3930K Processor. The operating system is Cent OS 6.3, and the compiler is Intel Composer XE 2013.

We constructed the FORTRAN code accelerated by OpenACC and measured the calculating speed of this system on 100 steps 10 times with PC1 and PC2. The speedup is defined as follows:

$$\text{Speedup} = \frac{\mathrm{T_{PC1}}}{\mathrm{T_{PC2}}} \tag{6}$$

where T represents the calculation process time. With the GPU parallelization, we achieved 13.6x speedup over the serial CPU solver. In contrast to the fact that approximately 2x speedup can be achieved with OpenMP, GPU computing accomplished the significant speedup. The flight simulation so far was carried out for some maneuvers only because of its long calculation time. However, with GPU computing, it is expected that flight simulation from take-off to landing can be calculated in future.

Conclusions

In this study, the coupled computational method of the unstructured MCD method and six-degrees-of-freedom flight dynamics was constructed, and GPU parallelization accelerated this system. In the simulation of aerobatic maneuver, the ailerons and rudder are operated, and then pressure distribution at main wing changed, which completes the aileron roll. Then, GPGPU by OpenACC was carried out to achieve the high-speed computation. GPU acceleration attained 13.6x speedup over the serial CPU computing. It is confirmed that GPU computing is effective for the acceleration of this system.

Acknowledgments

This publication was subsidized by JKA through its promotion funds from KEIRIN RACE and by JSPS KAKENHI Grant Number 16K06079.

References

[1] M. D. Salas, “Digital Flight: The Last CFD Aeronautical Grand Challenge”. Journal of Scientific Computing, Vol.28, Nos. 2/3, pp. 479-505, 2017.

[2] K. Watanabe and K. Matsuno, “Moving Computational Domain Method and Its Application to Flow Around a High-Speed Car Passing Through a Hairpin Curve”, Journal of Computational Science and Technology, Vol.3, No.2, pp. 449-459, 2009.

[3] K. Mihara, K. Matsuno and N. Satofuka, “An Iterative Finite−Volume Scheme on a Moving Grid (1st Report, The Fundamental Formulation and Validation)”, Transactions of the JSME (in Japanese), 65-637, No. 99-0060, pp. 2945-2953, 1999.

[4] M. Yamakawa and K. Matsuno: “An Iterative Finite−Volume Method on an Unstructured Moving Grid (1st Report, The Fundamental Formulation and Validation for Unsteady Compressible Flows)”, Transactions of the JSME (in Japanese), 69-683, No. 02-1277 pp. 1577-1582 ,2003.

[5] M. Yamakawa, D. Takekawa, K. Matsuno and S. Asao, “Numerical Simulation for a Flow around Body Ejection using an Axisymmetric Unstructured Moving Grid Method”, Computational Thermal Sciences, Vol.4, No.3, pp.217–223, 2012.
[6] S. Asao, K. Matsuno and M. Yamakawa, “Simulations of a Falling Sphere with Concentration in an Infinite Long Pipe Using a New Moving Mesh System”, Applied Thermal Engineering, Vol. 72, pp.29–33, 2014.
[7] S. Asao, K. Matsuno and M. Yamakawa, “Simulations of a Falling Sphere in a Long Bending Pipe with Trans-Mesh Method and Moving Computational Domain Method”, Journal of Computational Science and Technology, Vol.7, No.2, pp.297–305, 2013.
[8] S. Asao, K. Matsuno and M. Yamakawa, “Parallel Computations of Incompressible Fluid-Rigid Bodies Interaction Using Transmission Mesh Method”, Computers & Fluids, Vol.80, pp.178–183, 2013.
[9] S. Asao, K. Matsuno and M. Yamakawa, “Parallel Computations of Incompressible Flow Around Falling Spheres in a Long Pipe Using Moving Computational Domain Method”, Computers & Fluids, Vol.88, pp.850–856, 2013.
[10] M. Yamakawa, Y. Kita and K. Matsuno, “Domain decomposition method for unstructured meshes in an OpenMP computing environment” Computers & Fluids, Vol. 45, pp.168–171, 2011.
[11] Roe, P. L: “Approximate Riemann Solvers, Parameter Vectors, and Difference Schemes”, Journal of Computational Physics, 43 pp. 357–372, 1981.
[12] V. Venkatakrishnan: “On the Accuracy of Limiters and Convergence to Steady State Solutions”, 31st Aerospace Sciences Meeting and Exhibit, AIAA 93-0880, 1993.
[13] M. Yatabe: “Handy Note for Quaternions”, MSS Technical Report (in Japanese), Vol.18, pp. 29–34, 2007.
[14] K. Kawamura, Y. Ueno and Y. Nakamura: “Flight Simulation of Taketombo Based on Computational Fluid Dynamics and Computational Flight Dynamics”, Journal of the JSASS (in Japanese), Vol. 56, No. 654, pp. 324–330. 2008.
[15] M. Murayama, K. Nakahashi and K. Matsushima: “Unstructured Dynamic Mesh for Large Movement and Deformation”, 40th AIAA Aerospace Sciences Meeting & Exhibit, AIAA 2002-0122, 2002.
[16] Y. Ito and K. Nakahashi: “Surface Triangulation for Polygonal Models Based on CAD Data”, International Journal for Numerical Methods in Fluids, Vol. 39, Issue 1, pp. 75–96, 2002.
[17] Y. Ito: “Challenges in Unstructured Mesh Generation for Practical and Efficient Computational Fluid Dynamics Simulations”, Computers & Fluids, Vol. 85 pp. 47–52, 2013.