Direct Numeric Simulation of Sheared Convective Boundary Layer Entrainment with GPUs

***Nicholas J. Stewart[1] , David W. Holmes[1], Wenxian Lin[1], Steven W. Armfield[2] and Michael P. Kirkpatrick[2]**

[1]School of Engineering and Physical Sciences, James Cook University, QLD 4811, Australia

[2]School of Aerospace, Mechanical and Mechatronic Engineering, The University of Sydney, NSW 2006, Australia

*Corresponding author: nicholas.stewart@my.jcu.edu.au

Abstract

Sheared convective boundary layers (SCBL) are a frequently observed boundary layer in nature and industry. This paper presents work conducted to validate a numerical fluid model of sheared convective boundary layers implemented in Nvidia's CUDA programming language for graphical processing units. The code is based on finite difference implementation of the SIMPLE algorithm using the Boussinesq approximation to couple the energy equation through the buoyancy term. Work presented shows validation of the model on simpler test cases that are more thoroughly understood, and the model shows agreement with physical phenomena.

Keywords: high performance computing, GPGPU, CUDA, boundary layer

Introduction

Sheared convective boundary layers (SCBL) are turbulent boundary layers that are prevalent in environmental systems such as the Earth's atmosphere and engineered systems including air-conditioning and natural ventilation of building spaces. As the name suggests, a SCBL is a boundary layer that is driven by both shear and convective motion of the fluid. Currently the flow behaviours of SCBLs are not well understood and parameterising entrainment and other flow characteristics is unresolved. Numerical modelling and laboratory scale experiments are required to give a better understanding of SCBLs [1]. Both techniques have been employed previously to develop understanding of a similar problem, convective boundary layers. The work presented in this paper will focus on validating the numerical model against simplified cases including a lid driven cavity, a Blasius flat plate and a stratified shear layer.

Due to the turbulent nature of SCBLs, high resolution numerical meshes are required to resolve the smallest scales possible.This requires significant computational power that is typically derived from either distributed (cluster) computing or through multi-core processing. In recent years, GPUs, or more commonly known as graphics cards, have been developed to perform general purpose processing tasks and are now similar in capabilities to CPUs, but offer significantly higher performance due to GPU's massively parallel architecture. To unlock this performance, heavy modification of traditional algorithms is usually required, leading to the field of programming known as GPGPU, or General Purpose computing on GPUs.

This paper will validate the numerical model developed on Nvidia's GPGPU programming language, CUDA, using several simplified cases of SCBLs. The numerical model

is based on the semi-implicit SIMPLE [2] algorithm and solves the incompressible Navier-Stokes equations while using a buoyancy term to couple the energy equation through the Boussinesq approximation.

Numerical Model

The Navier-Stokes and energy equations are needed to be solved to model a SCBL system. The model was simplified by assuming that the system is incompressible and the Boussinesq approximation is valid for the buoyancy force. The incompressible form of the governing equations can be written as follows

$$\nabla \cdot \mathbf{u} = 0, \tag{1}$$

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = -\frac{1}{\rho} \nabla P + \nu \nabla^2 \mathbf{u} - \frac{\rho}{\rho_0} \mathbf{g}, \tag{2}$$

$$\frac{\partial T}{\partial t} + \mathbf{u} \cdot \nabla T = \sigma \nabla^2 T, \tag{3}$$

where $\mathbf{u}$ is the velocity vector, t is time, ρ is density of fluid at temperature T, ρ_0 is the density of fluid at a reference temperature T_0, P is pressure, $\mathbf{g}$ is acceleration due to gravity, and ν and σ are the kinematic viscosity and thermal diffusivity of fluid, respectively. The Navier-Stokes equation (2) is coupled with the energy equation (3) through the buoyancy term $\frac{\rho}{\rho_0}\mathbf{g}$.

The SIMPLE algorithm is used to solve the governing equations by using a cyclic system of guess and correct steps. The velocity components are first determined by solving the momentum equations using an approximated pressure field, coupled with the energy equation, then the pressure and velocity fields are corrected to satisfy continuity. This process continues until the system converges. The accuracy of the model is largely based on how the individual terms are discretised. The model was implemented with first order upwind advection terms and second order central diffusion terms.

GPGPU Implementation

The numerical algorithms were implemented in Nvidia's CUDA programming language. The development process started with modifying an existing code and implementing it in a naive way on GPU (unrolling loop structures into GPU kernels) and followed with optimizing the algorithm for GPU. The code was benchmarked during all stages of development to determine how much performance is gained by which stages of implementation. This is helpful to gauge the viability of projects such as OpenACC, which provide compiler directives for automatic generation of GPU kernels, when applied to non-trivial parallel problems.

The performance of the algorithm was submitted as another conference paper [10], however a brief of those results are shown here which demonstrate that the implementation on GPU significantly improved the performance, as shown in Fig. 1, which shows the solution time of different processing units as the model node count is increase. Both the CPU and GPU solution time increases linearly as node number increases, but the GPU model performs consistently much better than the CPU model. Fig. 2 shows the speed

up obtained with the semi-implicit algorithm (as a multiplier of the CPU performance), demonstrating it steadily growing as the model size increases. The performance difference between the two GPUs is largely explained by the different clock speed on the units.

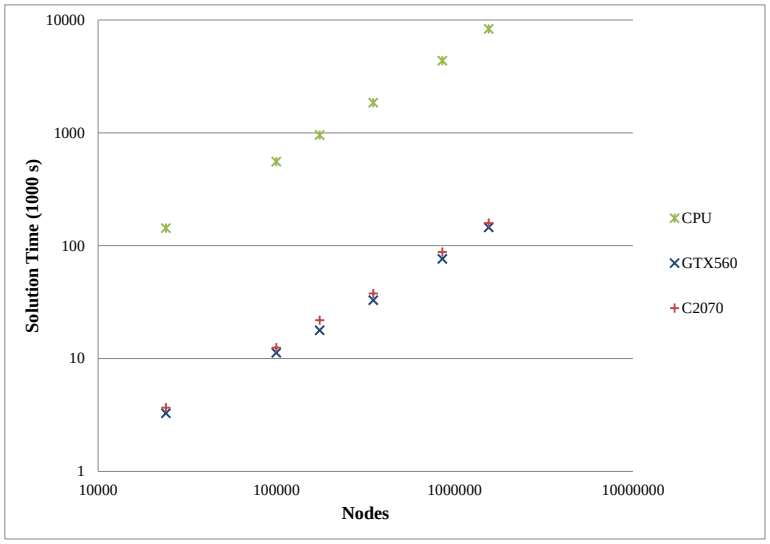

Fig. 1: Semi-implicit model performance as node number is increased.

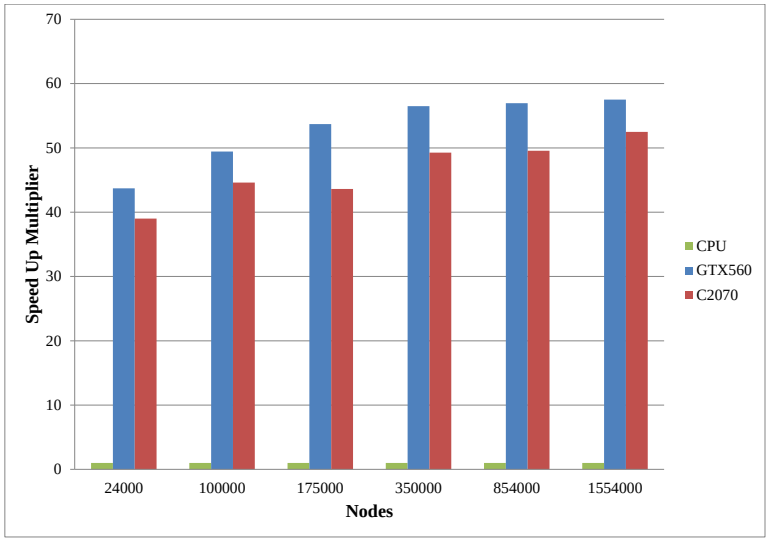

Fig. 2: Semi-implicit model performance improving when implemented on GPU.

Validation

Simplified test cases were used to ensure the correctness of the numerical model. Three main cases were investigated: a lid driven cavity flow, uniform flow over a flat plate, and a stratified shear layer. These cases were compared with literature to ensure consistent results.

Lid Driven Cavity.

The lid driven cavity is a common validation case with simple geometry and boundary conditions. The case is set up by having a square field of fluid, with non-slip conditions on the sides and floor, and a constant velocity boundary on the lid of the cavity as shown in Fig. 3.

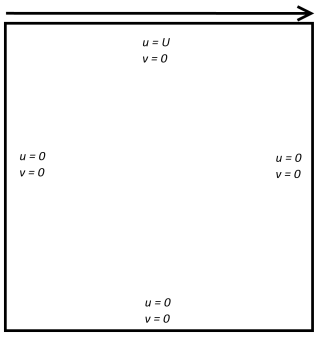

Fig. 3: Numerical setup of lid driven cavity flow.

The work performed by Ghia, Ghis and Shin [7] is frequently used as a benchmark, as it provides both images to compare, and tables of values. Fig. 4 shows an agreement of velocity profiles between the developed model and results of previous work. Fig. 5 shows the streamlines of the lid driven cavity, similar to those shown in other work.

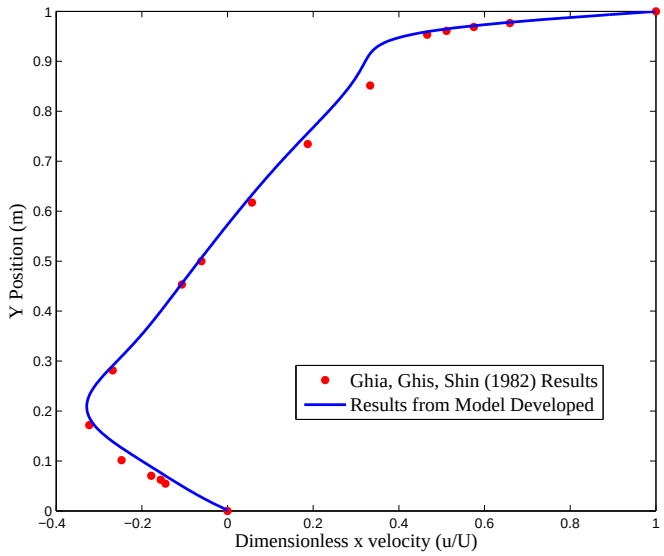

Fig. 4: Horizontal velocity profile over the midline of lid driven cavity with a Reynolds number of 1000

Flat Plate.

In the flat plate example a uniform velocity field is applied over a non-slip flat plate, on which a boundary layer develops. Fig. 6 shows a uniform flow field being applied to a flat plate, with the dashed line showing the boundary layer thickness (defined as where the velocity is less than 99% of the free field velocity).

The Blasius solution [8] gives the expected velocity profile and thickness of the boundary layer developed over a flat plate. Fig. 7 shows the profile determined by the CUDA algorithm matches closely with the Blasius solution. The main difference between the solutions is the bump occurring at the edge of the boundary layer, caused by the uniform inlet being applied at the start of the non-slip plate.

Stratified Shear Layer.

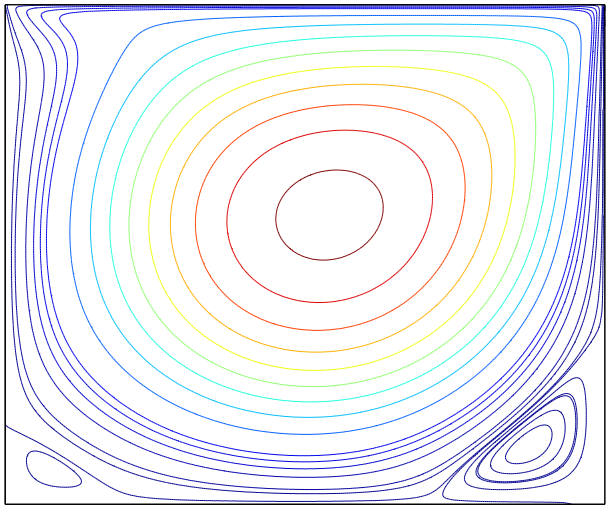

Fig. 5: Streamlines of lid driven cavity with a Reynolds number of 1000

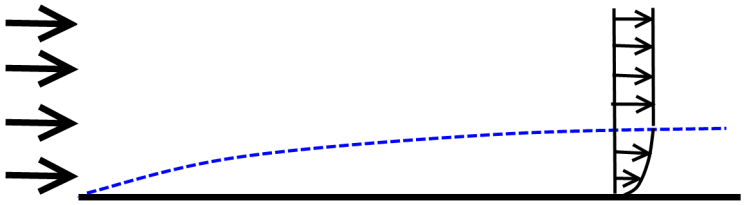

Fig. 6: The flat plate example showing the development of the boundary layer and the resultant velocity profile.

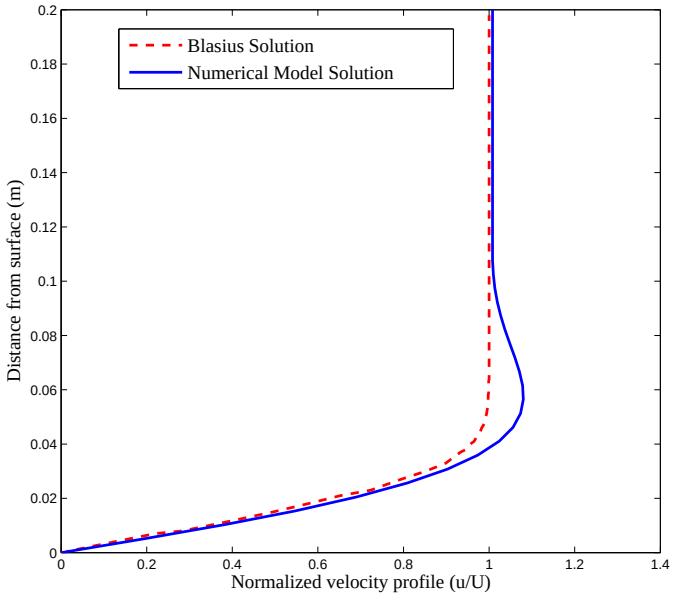

Fig. 7: Comparison of velocity profile calculated by the Blasius solution and the developed model.

In the stratified shear layer case two statically stable layers of fluid are moving at different velocities causing a shearing effect at the interface. A model can be set up as shown in Fig. 8 to investigate the effects of shear and buoyancy forces on the system. Characteristics of the system are largely determined by the Reynolds number and the Richardson number. The Reynolds number is given by

$$Re = \frac{\Delta U d}{\nu}, \tag{4}$$

where ΔU is the velocity difference between the two layers, and d is the interface thickness. The Richardson number is given by

$$Ri_g = \frac{N^2}{(\frac{\partial u}{\partial z})^2}, \tag{5}$$

where $N^2 = \frac{-g}{\rho_1}\frac{\partial \rho_0}{\partial z}$ is the buoyancy frequency, $\frac{\partial u}{\partial z}$ is the rate of change of velocity in the vertical direction, g is gravity, ρ_1 is the upper layer density and ρ_0 is the reference density. The Richardson number gives the ratio of buoyancy forces to shear forces.

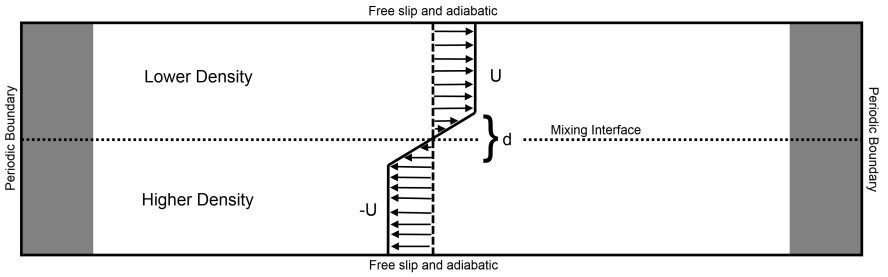

Fig. 8: The numerical model setup for stratified shear flow.

When the value for Richardson number is below 0.25, shear forces overwhelm the buoyancy forces to the extent that Kelvin-Helmholtz instabilities begin to occur [9]. The code developed reliably shows the development of K-H instabilities at low values of Richardson number as shown in figure 9, where $Ri = 0.2$ and $Re = 5000$. In this case the density gradient is generated by a change in temperature of 0.5°C.

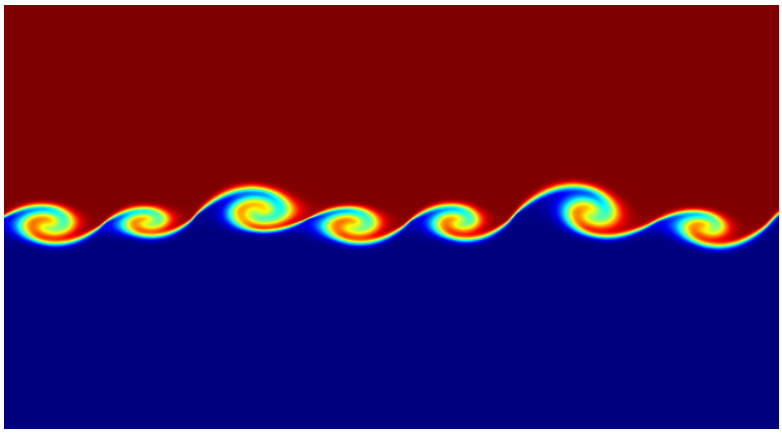

Fig. 9: Kelvin-Helmholtz instabilities in a stratified shear layer at $Ri = 0.2$ and $Re = 5000$.

Conclusion

The developed code was tested against several simplified cases of SCBLs, i.e. a lid driven cavity, a flat plate boundary layer and a stratified shear layer. In all cases it

was shown the numerical model agrees with previous works to within reasonable expectations. This work is to be extended to investigating the case of sheared convective boundary layers.

Acknowledgements

This work has been supported by an Australian Research Council Discovery Project Grant DP110102343.

References

[1] J.R. Conzemius, E. Fedorovich, Dynamics of shear convective boundary layer entrainment, J. Atmospheric Sci., 66, (2006) 1151-1178.

[2] L.S. Caretto, A.D. Gosman, S.V. Patankar, Two calculation procedures for steady, three-dimensional flows with recirculation, Proc. of the 3rd Int. Conf. on Numerical Methods in Fluid Mechanics, Lecture Notes in Physics, Vol. 19, 1972, Pages 60-68.

[3] R.W. MacCormack: The effect of viscosity in hypervelocity impact cratering, AIAA Paper, 1969, Pages 69-354.

[4] Nvidia: CUDA Programming Guide on http://www.nvidia.com/cuda/.

[5] R.S. Bernard, A MacCormack scheme for incompressible flow, Computers & Mathematics with Applications, 24, (Issues 5–6), (1992) 151-168.

[6] R.H. Pletcher, J.C. Tannehill, D.A. Anderson: Computational Fluid Mechanics and Heat Transfer, third ed., CRC Press, Boca Raton, 2012.

[7] U. Ghia, K.N. Ghis, C.T. Shin: High-Re Solutions for Incompressible Flow Using the Navier-Stokes Equations and a Multigrid Method, J. Computational Physics, 48, (1982) 387-411.

[8] F.M. White: Fluid Mechanics, fifth ed., McGraw-Hill, 2003.

[9] E.J. String, H.J.S Fernando: Entrainment and mixing in stratified shear flows, J. Fluid Mechanics, 428, (2001) 349-386

[10] N.J. Stewart, D.W. Holmes, W. Lin, et al: Comparison of Semi-Implicit and Explicit Finite Difference Algorithms on Highly Parallel Processing Architectures, submitted to Australasian Conference on Computational Mechanics 2013, Sydney, AU