Advanced CFD Modeling using GeForce GPUs

Auro Ashish Saha*
Department of Mechanical Engineering
Pondicherry Engineering College
Pondicherry, 605014
India

P. Balamurugan, K. Raj Kiran, B. Padma, A. Ramkumar
PG Student
Department of Mechanical Engineering
Pondicherry Engineering College
Pondicherry, 605014
India

Advanced applications of CFD for multiphysics modelling of electrokinetic, capillary, turbulent and rarefied hypersonic flows is discussed in this paper. Due the complexity of the geometry involved and the underlying physics associated with the phenomena to be studied, multiphysics study requires enormous computational resources. The CFD computations are performed within a parallel environment for accelerating solution time by distributing the processes on many nodes in a clustered supercomputing distributed parallel environment or by using the multi core CPU in a single node. Many CFD applications today can be accelerated significantly by using GPUs as the compute engine instead of the conventional CPUs. The current study investigates the significant improvements in the CFD simulations using multi core central processing unit (CPU) and graphical processing unit (GPU).

Keywords: CFD, GPU, Modeling, CUDA, GeForce, Fermi, Kepler, Tesla

Nomenclature

- \( C_0 \): Ionic concentration in bulk solution
- \( C_a \): Capillary number
- \( E \): Electric field intensity
- \( F \): Faraday’s Constant
- \( F_s \): Volumetric force representing the surface tension
- \( h \): Height of microchannel
- \( k \): Turbulence kinetic energy
- \( L \): Penetration length
- \( n \): Normal vector
- \( \hat{n} \): Unit vector normal to the surface
- \( \hat{n}_w \): Unit vector normal to the wall
- \( P \): Pressure
- \( Re \): Reynolds number
- \( R_g \): Universal gas constant
- \( t \): Time
- \( \hat{t}_w \): Unit vector tangent to the wall

- \( u_f \): Fluctuating velocity
- \( U \): Average velocity
- \( v \): Contact line velocity
- \( V \): Velocity vector
- \( z_e \): Valence number of univalent fluid
- \( z_{e+} \): Valence number of the positive/negative ions in the fluid

Greek symbols

- \( \varepsilon \): Permittivity of fluid, Turbulence eddy dissipation
- \( \lambda \): Debye length
- \( \mu \): Viscosity of fluid
- \( \phi \): Applied potential
- \( \psi \): Potential due to electrical double layer
- \( \rho \): Density of fluid
- \( \rho_e \): Charge density
- \( \xi_0 \): Zeta potential or wall potential
- \( \theta \): Contact angle
- \( \sigma \): Surface tension
- \( \tau \): Stress tensor
- \( \mu_t \): Turbulent viscosity
- \( \kappa \): Curvature of the surface

Subscripts

- \( d \): Dynamic
- \( e \): Equilibrium/Static
- \( s \): Surface tension
- \( sg \): Solid-gas
- \( sl \): Solid-liquid
- \( w \): Wall

Abbreviations

- CFD: Computational Fluid Dynamics
- CPU: Central Processing Unit
- CUDA: Compute Unified Device Architecture
- DI: De Ionized
- DSMC: Direct Simulation Monte Carlo
- FEM: Finite Element Method

*Address all correspondence to this author. Email: asaha@pec.edu
1 Introduction

Computer modeling has made impressive progress in scientific, engineering, biological and medical applications in recent years. CFD simulations are receiving more and more attention from the medical sector due to their ability to mimic effects that directly or indirectly impact the diagnosis [1]. Surface tension gradient is the important phenomenon involved in the flow at zero gravity or micro gravity condition which is more prominent and acts as the major transport phenomenon in the flow. Capillary flow has importance in space cryogenics in storage of propellant, heat pipes used in heat transfer for space equipment, transfer of cryogenic propellant to engine [2]. A hypersonic vehicle entering an atmosphere will go through many different flow regimes due to the change in atmospheric density with altitude and therefore, there is an extremely important role for computational models in the development of hypersonic vehicles [3]. Both FEM and FVM based solvers have been used for the computational study. FEM based solver is used for the study of electrokinetic phenomena and FVM based solver is used for other applications discussed in this paper.

2 Modeling Electrokinetic Flows

Electroosmotic flow refers to the bulk flow of an aqueous solution induced by the application of the electric field. When a charged surface comes in contact with ionizing fluid, the ions having polarity opposite to the surface charge are attracted towards the surface. These ions accumulate near the surface, forming an electric double layer (EDL). When electric potential is applied across a capillary, the ions in the EDL move under the influence of the electric field. Electroosmosis, electrophoresis, streaming potential, and sedimentation potential are broadly classified as electrokinetic phenomena.

Electric double layer (EDL) plays a fundamental role in many real-world systems. For instance, milk exists only because fat droplets are coated with EDL that prevent their coagulation into butter. EDLs exist in practically all heterogeneous fluid-based systems, such as blood, paints, inks, ceramic slurries and cement slurries.

Force acting on ions is accommodated as a body force in Navier-Stokes equation

$$\rho \nabla \nabla \nabla = - \nabla p + \mu \nabla^2 \nabla + \rho E$$

where \( \rho \) is the density, \( \mu \) is the coefficient of viscosity, \( \nabla \) is the Laplacian operator, \( \rho \) is the charge density and \( E \) is the electric field. Continuity equation for steady flow with constant properties

$$\nabla \cdot \nabla = 0$$

The electric field \( E \) has two contributions, the field due to applied potential \( \phi \)

$$\nabla^2 \phi = 0$$

and the field due to EDL \( \psi \)

$$\nabla^2 \psi = - \frac{\rho \psi}{\varepsilon}$$

Charge density is governed by the equation

$$\rho_e = F C_0 z_e e^{(-F w / R_g T)} - F C_0 z_e e^{(z_e F \psi / R_g T)}$$

For a symmetric electrolyte of valence number \( z_e \) the equation reduces to

$$\rho_e = 2 F C_0 \sinh \left( \frac{z_e \psi}{R_g T} \right)$$

where \( \psi \) is the potential due to electrical double layer, \( F \) is Faraday's constant, \( R_g \) is universal gas constant

Elmer, a FEM based solver is an open source (GPL) computational tool for multiphysics problems [4] is used here to study the electrokinetic phenomena. For combined pressure and electroosmotic driven flow in a two dimensional straight microchannel with DI water as working medium, the electric potential and velocity distribution is shown in the Fig. 1. It is observed that different velocity distribution in the microchannel are obtained for the different applied potentials and fluid physical and dielectric properties [5].

3 Modeling Capillary Flows

Condition for thermodynamic equilibrium for systems containing three phase contact line is expressed by Young equation as shown in Fig. 2.

$$\sigma \cos \theta = \sigma_{ag} - \sigma_{ad}$$

Penetration of wetting liquid due to capillary pressure is expressed by Young-Laplace equation as shown in Fig. 3.

$$\Delta P = \sigma \left( \frac{1}{R_1} + \frac{1}{R_2} \right)$$

The different contact angle definitions are:
Microscopic - angle at microscopically small distance near the three phase contact line
Macrosopic - angle at macroscopically large distance from the contact line
Apparent - used for interpretation of experimental results

Volume of Fluid (VOF) technique is the most commonly used technique for the simulation of free surfaces within commercial and academic CFD packages. The model can handle flows that undergo arbitrary topologic transformations and density ratios of up to 1000:1 or higher. Considering that solution of the fluid dynamics problems involving contact angle dynamics depends on fluid physical properties and contact angle it is necessary to keep the grid size for which simulation time is within reasonable limits.

**Free-surface flow model** ⇒ Volume of fluid (VOF) method (interface capturing) describes the interface evolution by a convection equation for a volume fraction function.

**VOF model**:
Navier Stokes equation:

\[
\rho \left[ \frac{\partial \mathbf{v}}{\partial t} + (\mathbf{v} \cdot \nabla) \mathbf{v} \right] = -\nabla P + \mu \nabla^2 \mathbf{v} + \mathbf{F}.
\]

Continuity equation:

\[
\nabla \cdot \mathbf{v} = 0
\]

Passive transport equation (VOF):

\[
\frac{\partial F}{\partial t} + \nabla \cdot (F \mathbf{v}) = 0
\]

\[
F = \frac{\text{cell volume occupied by liquid}}{\text{total volume of the control cell}}
\]

\[
\rho = F \rho_2 + (1-F) \rho_1
\]

\[
\mu = F \mu_2 + (1-F) \mu_1
\]

Continuum Surface Force (CSF) model:
\[ \hat{F}_z = \sigma \kappa \nabla F \]

\[ \frac{d}{dt} (\rho \delta \lambda_{\text{avg}}) = 2 \sigma \cos \theta + \Delta P h - \frac{12 \mu L}{h} - \lambda_{\text{avg}} \]

\[ \frac{d^2}{dt^2} L^2 + B \frac{d}{dt} L^2 = A \]

\[ L = \left( \frac{A}{B \exp(-B t)} + \frac{\Delta}{B} + \frac{1}{2} \right) \]

\[ A = \frac{4 \sigma \cos \theta + 2 \Delta P h}{\rho h^2} , B = \frac{12 \mu}{\rho h^2} \]

**Analytical model** ⇒ Lucas-Washburn equation model describes the asymptotic dynamics of capillary filling ignoring the inertia effects. Reduced-order model on the other hand accounts for inertia.

**Lucas Washburn model:**

\[ \frac{\partial P}{\partial x} = -\mu \frac{\partial^2 u}{\partial y^2} \]

\[ \Delta P = \sigma \left( \frac{1}{R_1} + \frac{1}{R_2} \right) \]

\[ \Delta P = \sigma \frac{1}{R} \]

\[ R = \frac{h}{2 \cos \theta} \]

**Dynamic contact angle model:**

It is expressed as function of equilibrium contact angle, system thermophysical properties and instantaneous interfacial speed. The dynamic contact angle are hydrodynamic, molecular-kinetic and empirical models based. Hydrodynamic model is based on the viscous dissipation of liquid and does not account surface characteristics of solid. Molecular-Kinetic model incorporates energy dissipation of liquid occurs at the moving contact line and accounts surface characteristics of solid. Empirical model usually fits the experimental data and there is no theoretical background.

The dependence of the dynamic contact angle on the velocity suggested by Cox [6], is given by

\[ \theta_d = \theta_e + 144 \text{Ca} \]

This equation is based on hydrodynamic theory of the moving contact line which emphasizes the viscous energy dissipation in the bulk and removes the singularities at the triple line by using a microscopic slip boundary condition. The above equation is valid for \( \theta_d < 135^\circ \).

The dependence of the dynamic contact angle on the velocity \( v \) proposed by Blake [7] is given by

\[ \cos \theta_d = \cos \theta_e - \frac{\sigma_0}{\sigma} \sinh^{-1} \frac{v}{v_0} \]

The model is based on the molecular kinetic theory of the moving contact line and considers the energy dissipation at the
Meniscus front  
Meniscus displacement

Fig. 4: Capillary phenomena in 3D and 2D microchannel

Triple line to occur through effective free-energy barriers for the displacement away or towards the three phase contact line of the fluid molecules.

The dependence of the dynamic contact angle on the velocity is suggested empirically by Jiang et al. [8], and is given by

$$\cos \theta_d = \cos \theta_s - (1 + \cos \theta_s) \tanh(4.96Ca^{0.702})$$

where $Ca = \frac{\mu V}{\sigma}$. This equation is applicable to any macroscopic geometry as long as the effects of gravity, inertia and adsorption are absent.

All the simulations hereafter in this study are based on OpenFOAM [9, 10], which is fundamentally a tool for solving partial differential equations mainly for CFD, stress analysis, electromagnetics and finance while allowing unrestricted parallel computations.

The capillary phenomena in three and two dimensional microchannel with DI water as working medium is shown in Fig. 4. The surface tension effect of the side walls for the three-dimensional case causes faster meniscus displacement [11].

4 Modeling Turbulent Flows

The equations governing turbulent flow are:

Reynolds-averaged Navier Stroke equation:

$$\frac{\partial \rho U}{\partial t} + \nabla \cdot (\rho U \otimes U) = \nabla \cdot \{ -\rho \mathbf{u} \}$$

Turbulence kinetic energy equation:

$$\frac{\partial (\rho k)}{\partial t} + \nabla \cdot (\rho U k) = \nabla \cdot \{ \mu \nabla k + \frac{\mu_t}{\sigma_k} \nabla \mathbf{U} \} + P_k - \rho e$$

Fig. 5: Velocity distribution in channel for turbulent flow

Turbulence eddy dissipation equation:

$$\frac{\partial (\rho e)}{\partial t} + \nabla \cdot (\rho U e) = \nabla \cdot \left[ (\mu + \mu_t) \nabla e \right] + \frac{\varepsilon}{k} (C_{1k} P_k - C_{2k} e)$$

The turbulent flow simulation results depicting the velocity distribution in the microchannel geometry with whole blood as working fluid is shown in Fig. 5.

5 Modeling Rarefied Flows

Direct Simulation Monte Carlo (DSMC) method uses a probabilistic approach for simulation of flows of dilute gas [12]. In gas dynamics studies, the basic criterion of the flow regime is the Knudsen number.

$$Kn = \lambda / L$$

where $\lambda$ is molecular mean free path of the gas, $L$ is the characteristic length of the flying object. Rarefied gas flows are primarily classified in four flow regimes:

- $Kn \leq 0.01$  Continuum flow regime
- $0.01 \leq Kn \leq 0.1$  Slip regime
- $0.1 \leq Kn \leq 10$  Transition regime
- $Kn \geq 10$  Free molecule regime

A hypersonic vehicle entering an atmosphere will go through many different flow regimes due to the change in atmospheric density with altitude that are characterized by Knudsen number.

Figure 6 shows the overall temperature distribution for DSMC simulation in a supersonic corner [13] after a time lapse of 0.01 s.
6 Accelerating CFD Simulations

For accelerating the solution time, CFD computations of greater complexity are performed within a parallel environment. CFD simulation in a cardiac system which takes days to complete can become an attractive diagnostic tool, only if these simulations were completed quickly [14]. DNS turbulence modeling is the most accurate way of simulating combustion and is very resource demanding, both on the CPU hours and memory [15]. CFD simulations use high-order discretization for optimizing the accuracy at a given resolution and doubling the resolution increases the CPU time by a factor of $2^4$.

Amdahl’s law indicates that the maximum speed-up through parallel processing is set by the amount of code which has to run serial and shows there is a hard limit to this speed up. Transistors counts and densities on integrated circuits doubled approximately every two years for the period between 1958 - 2012 and are to double only every three years for the period after 2013. Around 2020 Moore’s law will cease to exist and no longer provide faster and more powerful PCs as it will be physically impossible to etch transistors nearing the size of atoms.

For scientific computations the following are the currently used computational hardware technologies:

1. Multicore, cluster, massively parallel based on CPU with either shared memory, distributed memory and hybrid distributed shared memory.
2. Heterogeneous parallel computers using both multicore CPUs and many thread GPUs mostly dependent on PCIe bandwidth.
3. Fused CPU-GPU Architectures in which the CPUs and GPUs are integrated onto the same chip.
4. CPU parallelisation uses OpenMPI library.
5. GPU parallelisation uses the CUDA library.

GPUs are massively parallel computers for work that can use large scale decomposition and offer orders of magnitude speedups. GPUs have a much simpler individual processing units and cannot match a CPU for general purpose performance. GPUs lack direct hardware control and overheads in transferring the data from main memory. GPUs outperform the CPUs depending on requirements and nature of work involved. CPUs are better when GPUs cannot be used due to limitations on decomposition of work combined with associated overheads. As the compute power of GPUs is tremendous and they are already being used for general purpose scientific computations [16]. The libraries tested for accelerating CFD simulations in OpenFOAM are given in Table 1.

The GPU libraries implemented in OpenFOAM for CFD acceleration are based on:

1. Cuda For FOAM Link (Cuflink) [17–19], an open source library for GPU acceleration in OpenFOAM. It supports double precision.
2. GPU linear solvers library for OpenFOAM (ofgpu) [20] by Symscape under GPL license. It supports only single precision.
3. SpeedIT Plugin [21] to OpenFOAM by Vratis released for demonstration purposes for GPU acceleration. It supports only single precision.

The specification of the hardware used for testing the CFD acceleration in this study is given in Table 2. Unlocked double precision floating-point performance ranges between 1/2 to 1/8 of peak single-precision floating point performance. Swap is used to increase the RAM capacities for larger models that need a higher memory and CPU time. The calculations get slowed down due to additional communication overheads.

At run-time, the user can select the CPU or GPU based lin-

<table>
<thead>
<tr>
<th>Table 1: OpenFOAM CFD Solver Libraries</th>
</tr>
</thead>
<tbody>
<tr>
<td>Solver Library</td>
</tr>
<tr>
<td>----------------</td>
</tr>
<tr>
<td>laplacianFoam</td>
</tr>
<tr>
<td>icoFoam</td>
</tr>
<tr>
<td>simpleFoam</td>
</tr>
<tr>
<td>interFoam</td>
</tr>
<tr>
<td>dsmeFoam</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 2: Specification of Hardware</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CPU</strong></td>
</tr>
<tr>
<td>Intel Core 2 Quad Q8400/4S @ 2.66/3.20 GHz</td>
</tr>
<tr>
<td>Number of Cores</td>
</tr>
<tr>
<td>Total Memory</td>
</tr>
<tr>
<td>Total Swap</td>
</tr>
<tr>
<td><strong>GPU</strong></td>
</tr>
<tr>
<td>GeForce GTX 560 Ti/660 @ 1.66/1.03 GHz</td>
</tr>
<tr>
<td>Number of Cores</td>
</tr>
<tr>
<td>Compute Capability</td>
</tr>
<tr>
<td>Global Memory</td>
</tr>
</tbody>
</table>
Table 3: OpenFOAM CFD Linear System Solvers

<table>
<thead>
<tr>
<th>Linear Solver</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PBICG</td>
<td>Preconditioned bi-conjugate gradient solver for asymmetric matrices</td>
</tr>
<tr>
<td>PCG</td>
<td>Preconditioned conjugate gradient solver for symmetric matrices</td>
</tr>
<tr>
<td>GAMG</td>
<td>Generalised geometric-algebraic multigrid solver</td>
</tr>
<tr>
<td>smoothSolver</td>
<td>Solver using a smoother for both symmetric and asymmetric matrices</td>
</tr>
<tr>
<td>diagonalSolver</td>
<td>Diagonal solver for both symmetric and asymmetric matrices</td>
</tr>
</tbody>
</table>

Fig. 7: Comparison of speed up for CPU and GPU simulations

ear solver as shown in Table 3, to be used to solve each matrix equation generated by a given application. The linear solvers are complimented with preconditioners and smoothers namely DIC, DILU and Gauss-Seidel.

**laplacianFoam:**

Speed up of GPU solver is observed to increase substantially with increase in computational cells when compared with the CPU solver as shown in Fig. 7 for Fermi GPU. The GPU computations are found to run 1.2 times faster with Kepler GPU.

**interFoam:**

As the fluid advances the liquid volume fraction approaches 1 in the microchannel as shown in Fig. 8.

Parallel speed up shows a non linear behavior with increasing liquid volume fraction as shown in Fig. 9. Clock time depends on the time duration of capillary phenomena for the geometry and time step size.

CPU execution time is observed to increase by 8 times, whereas the GPU execution time increases by 2 times with 4 times increase in cell number as shown in Table 4. GPUs performance is good when there are large number of cells.

Fig. 8: Snap shot of meniscus front for capillary flow in a pilared microchannel

Fig. 9: Variation of CPU clock time with liquid volume fraction

Table 4: Fermi and Kepler GPU Parallel Performance

<table>
<thead>
<tr>
<th></th>
<th>CPU</th>
<th>GPU</th>
</tr>
</thead>
<tbody>
<tr>
<td>23 s</td>
<td>1675 s for 1967 cells (Fermi)</td>
<td>450 s for 1967 cells (Fermi)</td>
</tr>
<tr>
<td>175 s</td>
<td>8760 cells (Fermi)</td>
<td>1050 s for 8760 cells (Fermi)</td>
</tr>
<tr>
<td>12 s</td>
<td>8760 cells (Kepler)</td>
<td>336 for 1967 cells (Kepler)</td>
</tr>
<tr>
<td>82 s</td>
<td></td>
<td>764 for 8760 cells (Kepler)</td>
</tr>
</tbody>
</table>

**icoFoam:** The Fermi GPU performance was carried for the case considering 200 x 200 computational cells. The clock time obtained is 8200 s for CPU simulation and 912 s, for GPU simulation using SpeedIT library. The clock time for simulations performed with double precision solvers for CPU and
GPU using Cufflink library, are 2512 s and 2854 s, respectively. The GPU computations are fast for single precision accuracy, whereas, double precision speed up of GPU solver is observed to increase dramatically as shown in Table 5.

Figure 10 shows the comparison of Kepler GPU performance with CPU with increasing computational cells.

**simpleFoam:**
With increase in computational cells, both single precision CPU and GPU solver become unstable, whereas, double precision speed up of GPU solver is observed to increase dramatically as shown in Table 5.

GPU execution time is observed as 3895 s, 9234 s and 24325 s for 500000, 1000000 and 2000000 cells, respectively using Kepler GPU with simpleFoam solver. Using Fermi GPU the execution time is observed as 28521 s for 2000000 cells.

dsmcFoam: The simulations were performed using 1 CPU and 4 CPU cores, and the clock times are 23625 s and 10162 s, respectively. The speed up obtained with 4 CPU cores is little more than 2. The time step size considered for the study is 1E-6 s.

Figure 11 shows that the results of advancing hardware capability of GPU architecture. Kepler compute capability is observed to be higher than Fermi.

### Table 5: Performance of CPU and GPU

<table>
<thead>
<tr>
<th>Cells</th>
<th>SP Clock Time, s</th>
<th>DP Clock Time, s</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CPU SpeedIT ofgpu</td>
<td>GPU Cufflink</td>
</tr>
<tr>
<td>12225</td>
<td>-  177</td>
<td>207</td>
</tr>
<tr>
<td>48900</td>
<td>- -  304</td>
<td>1861</td>
</tr>
<tr>
<td>1,95,600</td>
<td>- - -</td>
<td>22534</td>
</tr>
</tbody>
</table>

#### 7 Conclusions

Acceleration of CFD simulation using both CPU/GPU has been addressed here. Computational resources required for unsteady flows are much greater than that for steady flows. It was shown that significant performance improvement can be achieved by GeForce GPUs that are designed for consumer graphics and professional visualization, which do not have full double precision floating point performance support. Acceleration of unsteady simulations is very much desirable to enable simulation for longer trajectories of time. Even with GPUs there is a need for greatly improved parallel algorithms to exploit their full hardware potential for different multiphysics computational applications. Further investigation is needed for the applications discussed using Tesla GPUs that are specially designed for parallel computing and programming and offers exclusive high performance computing features as full double precision floating point performance support.

### Acknowledgements

This paper is based on the invited lecture on the topic Advanced CFD Modeling delivered by AAS for Faculty Development Program on CFD at SRM Easwari Engineering College, Chennai - 600089, India on 14-06-2013.

### References


[17] Combest, D. P., and Day, J., 2012. Cufflink: a library for linking numerical methods based on CUDA C/C++ with OpenFOAM. Version 0.1.0 ed...

[18] Bell, N., and Garland, M., 2012. Cusp: Generic parallel algorithms for sparse matrix and graph computations. Version 0.3.0 ed...


[20] Symscpe, 2012. GPU Linear Solvers for OpenFOAM. Version 0.2 ed...

[21] Vaitis, 2012. SpeedIT Plugin for OpenFOAM. Version 2.1 ed...