Available online at www.sciencedirect.com

ScienceDirect

Procedia IUTAM 15 (2015) 305 – 312

Procedia IUTAM

www.elsevier.com/locate/procedia

IUTAM Symposium on Multiphase flows with phase change: challenges and opportunities, Hyderabad, India (December 08 – December 11, 2014)

Study of disintegration of a high speed liquid jet using VOF method

Rajesh Reddy[a,*], R. Banerjee[a]

[a]*Department of Mechanical and Aerospace engineering, Indian Institute of Technology Hyderabad, Hyderabad and 502205, India*

Abstract

Numerical simulations are carried out to look at the primary atomization of a 2-D planar liquid jet. A finite volume method based solver is developed and interface capturing is done by volume of fluid (VOF) method. The solver uses a projection algorithm to solve the governing equations. Preconditioned conjugate gradient method is used to solve the pressure poisson equation. This part of the solver is ported on to graphics processing unit (GPU) to meet the computational demand required. The solver is validated against standard benchmark test cases. Initially the parallelized version on GPU is compared with the serial version on single CPU core to estimate the speed up achieved. Effect of liquid inlet velocity on jet disintegration is studied.

Peer-review under responsibility of Indian Institute of Technology, Hyderabad.

Keywords: VOF method; GPU computing; sprays;

1. Introduction

Breakup of a liquid jet emanating from an orifice is observed in several engineering and scientific applications like fuel injection in IC engines and gas turbines, spray painting, etc. The description of the moving interface is computationally very challenging in such type of flows. Traditional CFD approaches model the primary jet breakup using constitutive models. However, these models are very empirical in nature and require several model inputs which is typically derived from downstream experimental data. Additionally, obtaining experimental data in the dense spray region is extremely difficult. Hence, more recently high fidelity two-phase interface tracking methods are being increasingly used to simulate primary jet breakup [1,2]. The problem can be dealt from either Eulerian or Lagrangian approach. Eulerian model based on volume of fluid (VOF) method is used here because of its inherent ability to handle interfacial flows that undergo large topology changes including merging and breakup. The present solver is aimed at understanding the spray ejecting from an injector, which is a computationally expensive problem. Solution for the pressure Poisson equation consumes major part of the computational time. Hence, in order to reduce the computational time, parallelization of the pressure Poisson equation solver is done by programming on GPU in the present solver.

* Corresponding author. Tel.: +91-9866304076.
E-mail address: me10p006@iith.ac.in

Peer-review under responsibility of Indian Institute of Technology, Hyderabad.
doi:10.1016/j.piutam.2015.04.043

1.1. Introduction to GPU computing

In GPU computing, a Graphics Processor Unit is used in conjunction with CPU. It has become a recent trend in high performance computing to use GPUs for executing a part of the program in parallel. The reason being its high processing power and relatively low cost. Current GPUs are incorporated with hundreds of lightweight cores which can accelerate compute intensive applications substantially. The GPUs are efficient for data parallel applications like Computational fluid dynamics (CFD), as it has a Single Instruction Multiple Data (SIMD) device architecture. In the present work, GPU programming is done on a parallel computing platform called compute unified device architecture (CUDA), developed by NVIDIA Corporation[3].

Nomenclature

C	volume fraction
Eo	Eotvos number
F	surface tension force per unit volume
g	gravitational acceleration
L	liquid sheet thickness
p	pressure
R	radius of bubble
Re	Reynolds number
u	velocity component in x-direction
v	velocity component in y-direction
v	velocity vector

Greek symbols

Δt	time step
Δx	grid spacing in x-direction
Δy	grid spacing in y-direction
η	Kolmogorov length scale
μ	dynamic viscosity
ρ	density
σ	surface tension

Subscripts

l	liquid
g	gas

Abbreviations

CSF	Continuum Surface Force
CUDA	Compute Unied Device Architecture
EI-LE	Eulerian Implicit Lagrangian Explicit
ENO	Essentially Non-ooscillatory
GPU	Graphics Processing Unit
VOF	Volume of Fluid

2. Governing equations

The incompressible, variable density and isothermal flow of immiscible fluids is governed by

$$\nabla \cdot \mathbf{v} = 0 \tag{1}$$

$$\rho(\frac{\partial \mathbf{v}}{\partial t} + \mathbf{v} \cdot \nabla \mathbf{v}) = -\nabla p + \nabla \cdot [\mu(\nabla \mathbf{v} + \nabla \mathbf{v}^T)] + \mathbf{F} \tag{2}$$

Eq. (1) and (2) represent the conservation of mass and momentum, respectively. **F** in Eq. (2) is the body force term which can be gravity or surface tension force per unit volume. The location of the interface is determined by using VOF methodology. The VOF method solves an additional advection equation for volume fraction C, which is defined as fraction of reference phase fluid occupied in a computational cell.

$$\frac{\partial C}{\partial t} + \mathbf{v} \cdot \nabla C = 0 \tag{3}$$

From the estimation of volume fraction, the effective density and viscosity in a cell can be obtained as (assuming gaseous phase as reference phase)

$$\rho = \rho_g C + \rho_l (1 - C) \tag{4}$$

$$\mu = \mu_g C + \mu_l (1 - C) \tag{5}$$

3. Numerical modeling

Finite volume method (FVM) is used to represent the flow governing partial differential equations in the form of algebraic form. The grid is two dimensional, Cartesian, structured and collocated. A projection algorithm namely simplified marker and cell (SMAC) is used to solve the Navier-Stokes equations. VOF method, which is used for interface capturing consists of two parts, popularly known as interface reconstruction and interface advection. Youngs method[4] is adopted for interface reconstruction. Interface advection is done using Operator split scheme with Eulerian implicit-Lagrangian explicit method given by Aulisa *et al*[5]. Surface tension term is modelled using continuum surface force (CSF) method given by Brackbill *et al*[6].Convective term is discretized using a second order ENO scheme. Space derivatives are discretized using a second order central difference scheme. Pressure Poisson equation is solved by using symmetric Gauss Siedel preconditioned conjugate gradient (SGSPCG) method.

4. Parallelization on GPU architecture

In interest of brevity, introduction to GPU architecture and CUDA platform are not presented here. These details can be found in CUDA programming guide[7]. The pressure Poisson equation in the flow solver is ported on to GPU to accelerate the computation. As the remaining part of the solver is processed on CPU, the required data is to be transferred on to the device memory before the GPU execution starts. To achieve high performance, this data transfer between the host and device memory is to be minimised.

Conjugate gradient (CG) method is rarely applied directly to solve the linear system of equations, because of slow convergence issues. Generally, CG method is applied in preconditioned form to accelerate the convergence. The non preconditioned CG method is straight forward to implement. In-addition, the sparse matrix vector multiplication, dot product and other mathematical operations in CG algorithm are convenient to parallelize. The computational grid in CUDA is handled by dividing it in to two dimensional blocks. Each block consists, say m x n number of threads. A thread with a unique thread id is created corresponding to every cell center node in the actual computational domain. Symmetric Gauss Siedel (SGS) is used as preconditioner in CG algorithm. Applying SGS is inherently serial in nature. SGS method basically consists of two steps namely, forward Gauss Siedel (GS) sweep and backward GS sweep. For the algorithm to run on parallel threads it is necessary that there are no dependencies between variables on different threads. To resolve the dependencies, the current implementation employs Red-Black GS method[8] in forward sweep, followed by Black-Red GS method in reverse sweep. For a second order stencil used to solve 2-D Poisson equation, two colors (say red and black) are required to generate sets of points which are not related with each other. Now each colour can be processed separately and the calculations within a color are done in parallel. Each color is processed sequentially. Every particular thread is mapped to a corresponding point in the required color by using the block-id and thread-id provided by CUDA. The threads are mapped to points of one colour at a time. Fig. 1 shows colored domain and thread mapping arrangement for a sample domain size of 8X4 interior cells.

A single kernel is used to apply the boundary conditions on all boundaries. For invoking boundary conditions a one dimensional grid is assumed and as many threads as boundary points are created. A separate kernel is implemented to calculate the summation of elements in a vector. This kernel assumes a 1-D grid and calculation is done by using shared memory, which is several order faster than global memory.

NVIDIA GeForce GTX480 model GPU is used for present simulations. The GPU was hosted by a FUJITSU CELSIUS R670 workstation (via PCI Express interconnect), consisting of dual Intel Xeon 2.8 GHz processors, 24 GB memory and 2 TB hard disk. Serial simulations are done with Linux Redhat C++ compiler (g++). Parallel code compilation was performed with nvcc compiler. All the simulations are done with double precision floating point values.

5. Validation

5.1. Validation of interface tracking algorithm

VOF method is successfully tested for vortex in a box test introduced by Rider and Kothe[9]. This test case is believed to provide a complete assessment of the volume tracking algorithm. The details of the test procedure and its verification by present solver has been described in detail by Rajesh and Banerjee[10].

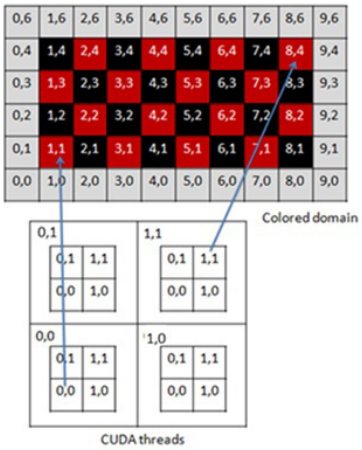

Fig. 1. Colouring of the domain with CUDA thread mapping arrangement.

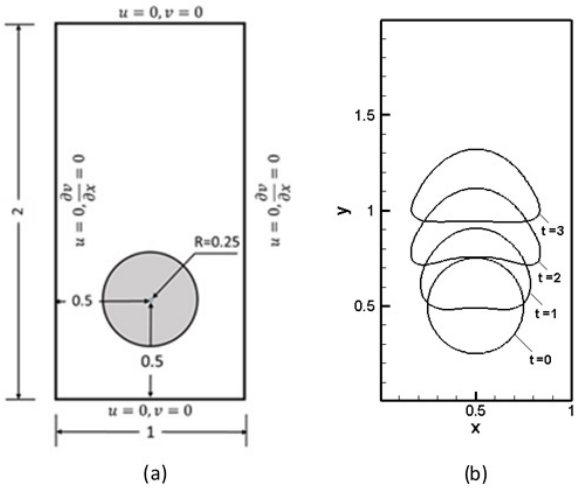

Fig. 2. (a) Initial configuration and boundary conditions for bubble rise test case; (b) Bubble position at different instants.

5.2. *Validation of flow solver*

The overall performance of the solver is established by validating it against the standard benchmark test case of buoyancy driven gas bubble in a quiescent surrounding fluid. Test case presented by Hysing et al.[11] is taken as reference. Initial configuration and boundary conditions are illustrated in Fig. 2(a) . The density and viscosity ratios are taken to be equal to 10. Surface tension constant is equal to 24.5. Gravitational velocity U_g is defined as $\sqrt{2gR}$. Reynolds number and Eotvos number are defined based on gravitational velocity and diameter of bubble. For the present case $Re = 35$ and $Eo = 10$. Time scale is defined as $t = \frac{2U}{g}$. The results are quantified in terms of bubble centroid position and rise velocity.

Simulations are done up to t=3 on three different grids of sizes 40x80, 80x160 and 160x320. Fig. 2(b) shows the interface shape of the rising bubble at different time instants on grid 80x160. The solution obtained on 80x160 grid is compared with the solution from Hysing et al. Fig. 3 shows the comparison of bubble centroid and mean rise

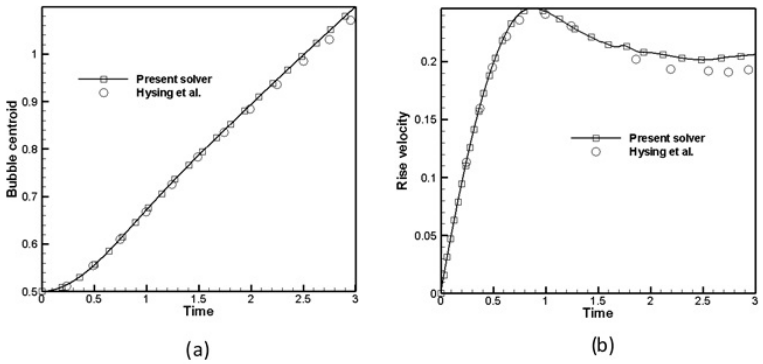

Fig. 3. (a) Centroid of bubble with time; (b) Rise velocity of bubble with time.

velocity. The bubble centroid increases monotonically with respect to time. However in case of bubble rise velocity, it initially increases with time, following which there is marginal decrease in the rise velocity before settling to an almost constant value. This is because by this time the buoyant force is balanced by the drag force and the bubble attains its terminal velocity. Fig. 2 and Fig. 3 show that the present solver is able to capture the bubble motion and show a good agreement with the reference literature[11]. Fig. 3(b) shows a small deviation in the rise velocity when compared with the reference result[11]. This difference is believed to be due to the surface tension term. Present solver uses standard CSF algorithm proposed by Brackbill et al[6]. The curvature calculation from volume fraction and the resulting jump condition at the liquid/gas interface results in spurious velocity at the interface which becomes increasingly apparent for surface tension dominated problems like the present case. As expected, surface tension effects are strong enough to hold the bubble together and no break up occurs. The bubble attains an ellipsoidal shape at the end of the simulation.

5.3. Performance acceleration with GPU parallelization

In order to estimate the performance of the GPU based parallelization, the bubble rise problem was solved using both the serial and GPU based parallelized version of the code. The performance is reported in terms of workunits.

Table 1. Performance acceleration with GPU. (tested by running for first 30 timesteps)

Grid	workunits		speedup
	CPU (x e^{-7})	GPU (x e^{-7})	
40x80	2.107	1.714	1.23X
80x160	2.042	0.3196	6.39X
160x320	2.062	0.1478	13.95X

A workunit is defined as the computational time required for each control volume per iteration and is mathematically expressed as $workunit = \frac{T_n}{N*I_n}$ where N is the number of control volumes in the domain, T_n is the time taken by the pressure Poisson solver for n number of time steps and the total number of iterations done during the n time steps is I_n.

Speed up obtained with GPU based solver is presented in Table 1. As can be seen from the table, the acceleration due to GPU parallelization increases with increase in the size of computational grid. This is because with larger grid sets, the time required by the GPU to perform the computations becomes larger than the time required for data transfer between the host and the device. Hence, the non-computational overhead decreases, which results in increase of the computational efficiency of the GPU.

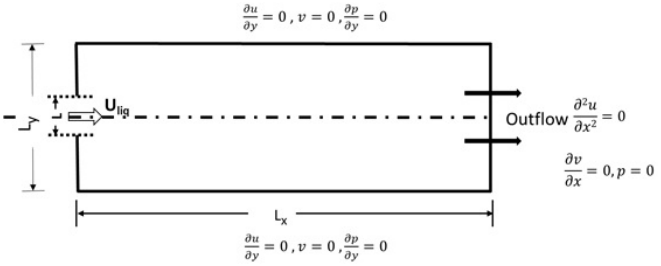

Fig. 4. Computational domain with BC.

6. Results and discussion

6.1. Liquid jet disintegration

The validated solver is applied to study the primary atomization of a liquid sheet emanating in to a quiescent gaseous environment. A planar liquid sheet of thickness L=100 m will be entering the domain with a specified inlet velocity. Computational domain of 10L x 5L is considered. Fig. 4 shows the numerical setup used for the simulation. Physical parameters used in the simulation are listed in Table 2. Simulations are performed for three different liquid inlet velocities. Simulation parameters along with non-dimensional numbers are given in Table 3. Liquid sheet thickness at inlet is taken as characteristic length. Liquid velocity at inlet is taken as characteristic velocity.

Table 2. Parameters used.

Phase	ρ (kg/m^3)	μ (kg/ms)	σ (N/m)	Sheet thickness (μm)
Liquid	700	1e-3	0.03	100
Gas	25	1e-5		

Table 3. Dimensionless numbers for different cases.

Case	u_l at inlet (m/s)	u_g at inlet (m/s)	*Re*	*We*
A	30	0	2100	2100
B	40	0	2800	3733
C	50	0	3500	5833

All the simulations are performed on a grid size of 1024x512. The grid size employed here satisfactorily resolves up to the Kolmogorov length scale (η). For the case with maximum Re (case C), grid spacing is approximately 4.4 η. No velocity perturbation is given at liquid inlet. For the spray simulations, GPU based solver has shown a speedup of approximately 18X. This speedup is obtained by running the simulation for 30 time steps and is calculated based on workunit definition.

The development of the planar jet with time, for the cases A, B and C is given as a sequence of images in Fig. 5-7 respectively. It can be seen from the Fig. 5 that the liquid sheet observes a high shear at the tip, as soon as it exits the nozzle and encounters the stationary gas. The tip of the sheet turns in to an umbrella shape, which leads to formation of ligaments at top and bottom edges. Breakup starts occurring from these ligaments, resulting in primary atomization. Corrugations start to appear on the surface of the sheet only after a certain distance from the injection region. Even though slight instabilities develop on the liquid sheet surface, they are not amplified with the progress of the flow. The droplets pinched off from the ligaments are convected upstream. This droplet pinching from ligaments is well

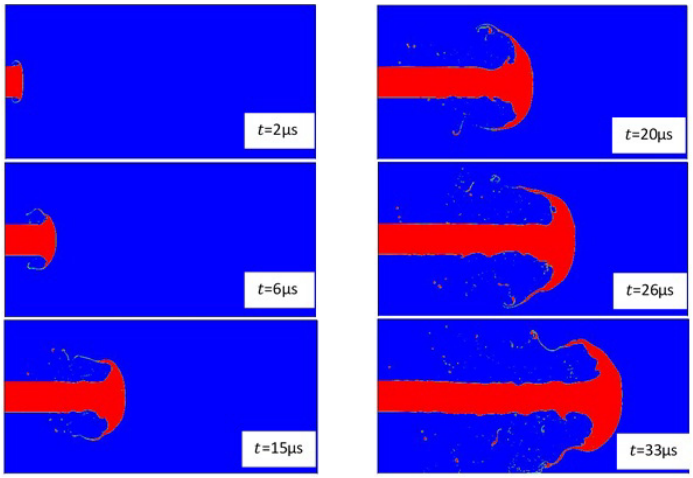

Fig. 5. Liquid jet disintegration for case A.

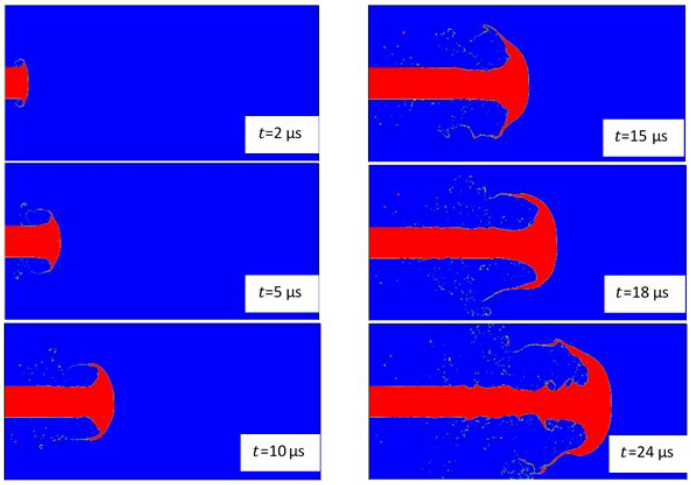

Fig. 6. Liquid jet disintegration for case B.

captured only if the mesh spacing is comparable to the thin liquid structure that forms during breakup. A portion of the fine droplets formed are entrained by the recirculating flow. Hence there is a possibility of interaction of droplets of different sizes and also droplets may interact with the liquid sheet core resulting in slight interface deformation. These interactions can affect the droplet size distribution.

The structure of sheet disintegration seems to be identical for all the three cases simulated. With the increase in liquid inlet velocity the droplet density is observed to be increased (qualitative assessment from the Fig. 5-7). As mentioned by Shinjo and Umemura[12], it is difficult to capture the final pinch-off moment of liquid structure in Eulerian-Eulerian formulation. This is because the liquid structure smaller than grid size will be automatically recognized as pinch-off. Artificial droplets of the size of grid spacing can be expected. In the present simulations, this effect is mitigated by employing the sufficiently fine grid. So the droplet generation shown by simulations is mostly

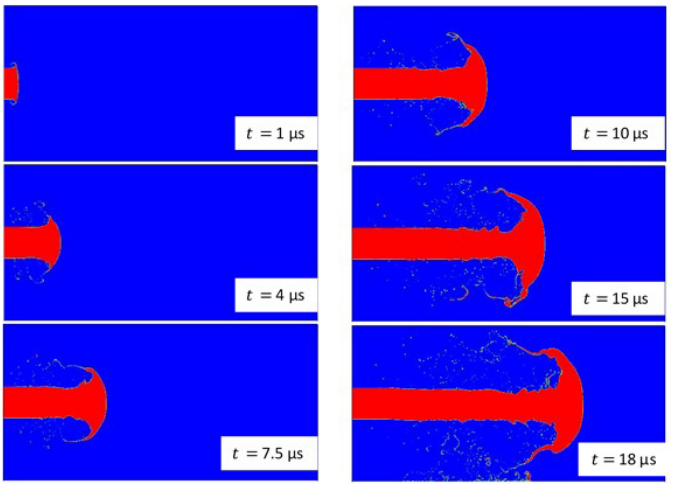

Fig. 7. Liquid jet disintegration for case C.

believed to be driven by surface tension effects. It is interesting to observe that the present 2-D simulation qualitatively represent the flow pattern of Diesel jet spray (3-D simulation) by Shinjo and Umemura [12].

7. Conclusion

A VOF based two-phase CFD solver was developed and validated against standard benchmark test cases. The pressure Poisson part of the solver was parallelized on GPU architecture. Significant speed up was achieved for the standard test cases performed. Also the parallelization efficiency increased with increase in grid size. The developed solver was used to study the primary jet breakup of 2-D planar liquid jet emanating in to a quiescent gaseous environment. Physically realistic solutions were obtained with the simulations. The tip of the liquid sheet forms an umbrella shaped front, during its propagation against the gas. The core of the liquid sheet remained intact throughout the simulation. Ligaments are formed from the umbrella shaped front. These ligaments are disintegrated in to smaller fragments and droplets. The effect of liquid inlet velocity on sheet disintegration was qualitatively seen. A speed up of approximately 18X was observed with GPU based solver for the stated spray simulations.

References

1. Fuster D, Bague A, Boeck T, Moyne L L, Leboissetier A, Popinet S, et al. Simulation of primary atomization with an octree adaptive mesh refinement and VOF method. Int J Multiph Flow 2009;35:550 65.
2. Mnard T, Tanguy S, Berlemont a. Coupling level set/VOF/ghost fluid methods: Validation and application to 3D simulation of the primary break-up of a liquid jet. Int J Multiph Flow 2007;33:510 24.
3. https://developer.nvidia.com/about-cuda.
4. Tryggvason G, Scardovelli R, Zaleski S. Direct Numerical Simulations of Gas- Liquid Multiphase Flows. Cambridge University Press; 2011.
5. Aulisa E, Manservisi S, Scardovelli R, Zaleski S. A geometrical area-preserving Volume-of-Fluid advection method. J Comput Phys 2003;192:355 64.
6. Brackbill JU, Kothe DB, Zemach C. A continuum method for modeling surface tension. J Comput Phys 1992;100:335 54.
7. NVIDIA Cuda C programming guide, 2012.
8. Demmel JW, Applied numerical linear algebra, SIAM, 1997.
9. Rider WJ, Kothe DB. Reconstructing Volume Tracking. J Comput Phys 1998;141:112 52.
10. Reddy R, Banerjee R. Simulation of gas blasted liquid sheet on GPU architecture. 26th Annual Conference on Liquid Atomization and Spray Systems, Bremen, Germany 2014.
11. Hysing S, Turek S, Kuzmin D, Parolini N, Burman E, Ganesan S, et al. Quantitative benchmark computations of two-dimensional bubble dynamics. Int J Numer Methods Fluids 2009;60:1259- 88.
12. Shinjo J, Umemura A. Detailed simulation of primary atomization mechanisms in Diesel jet sprays (isolated identification of liquid jet tip effects). Proc Combust Inst 2011;33:2089- 97.