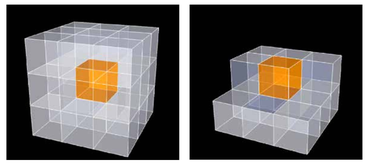
Accelerating Dissipative Particle Dynamics with GPU

Lin Chen-sen[1)] Chen Shuo[1)] Li Qi-Liang[2)] Yang Zhi-Gang[2)]
1) School of Aerospace Engineering and Applied Mechanics, Tongji University, Shanghai,200092, China
2) Shanghai Automotive Wind Tunnel Center, Tongji University, Shanghai,201804,China

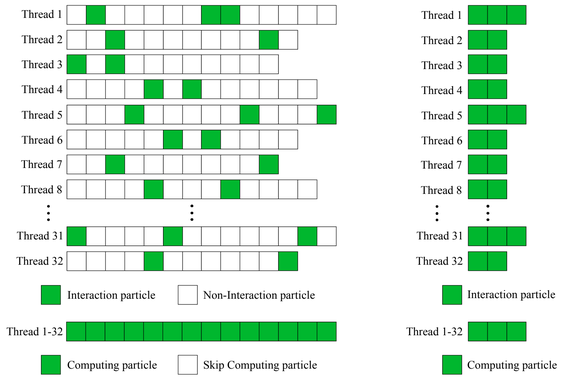
Introduction

The implement of GPU parallel computing of dissipative particle dynamics based on CUDA was carried out. Some issues involved, such as thread mapping, parallel cell-list array updating, generating pseudo-random number on GPU, memory access optimization and loading balancing are discussed in detail; Furthermore, Poiseuille flow and suddenly contracting and expanding flow were simulated to verify the correctness of GPU parallel computing. The results of GPU parallel computing of DPD show that speedup is up to 50X compared with CPU serial computing.

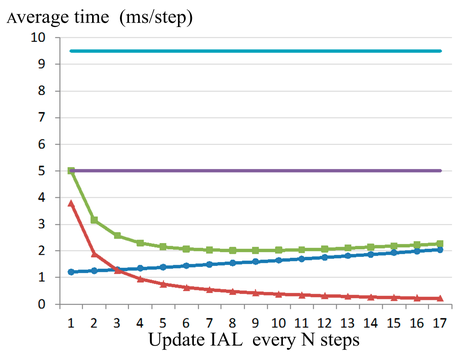
Methods



Cell approach
(avoid searching global, even 14 cells are sufficient)

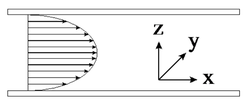


Original algorithm VS IAL algorithm

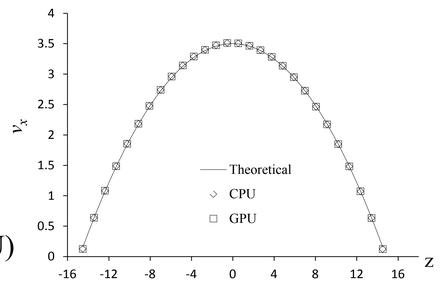


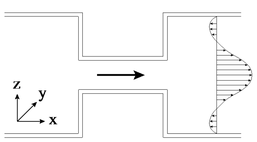
Effect of IAL algorithm with different parameters

Verification and Speedup

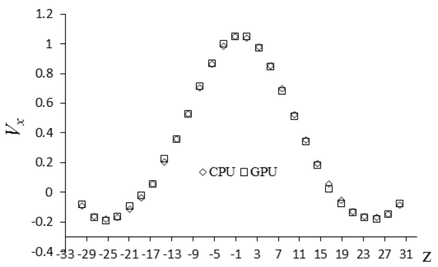


Velocity profile 1
(Theoretical&CPU&GPU)





Velocity profile 2
(CPU&GPU)



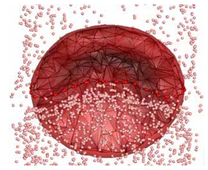
Zone size (number of particles)		6×4×8 (864)	64×8×32 (67584)	256×16×32 (540672)	256×32×32 (1081344)
CPU update speed (particles/s)		1.65×10^5	3.10×10^5	3.11×10^5	3.07×10^5
GPU update speed (particles/s)	Thread - Cell	5.25×10^4	1.74×10^6	3.16×10^6	3.59×10^6
	Speedup	0.32	5.61	10.16	11.69
	Thread - particle	1.51×10^5	3.69×10^6	5.23×10^6	5.82×10^6
	Speedup	0.92	11.90	16.82	19.15
	Interaction-List	9.78×10^5	1.35×10^7	1.46×10^7	1.49×10^7
	Speedup	5.92	43.61	46.84	48.35

Speedup in different zone sizes and different algorithm

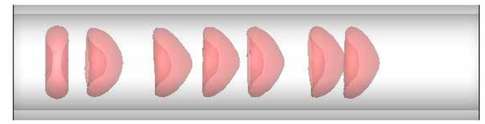
if $tid \leqslant n_{particles}$ **then**
  $f_{net} \leftarrow 0$
  load RNGstate ← RNGstate_global[tid]
  for $i < nlist\_length$ **do**
    $R \leftarrow$ callRNG()
    $f_{tid,nid} = F(velocity[tid], position[tid], velocity[nid], position[nid], R)$
    $f_{net} \leftarrow f_{net} + f_{tid,nid}$
  end for
  atomic $force[tid] \leftarrow force[tid] + f_{net}$
  store RNGstate_global[tid] ← R
end if

Algorithm of generating random number on GPU

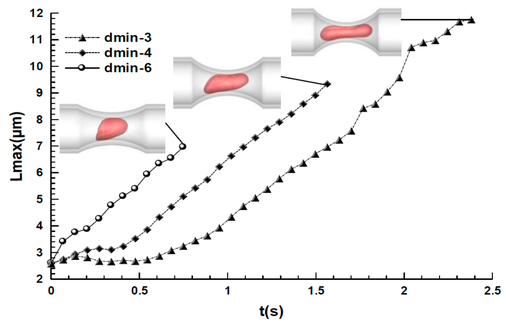
Simulation of red blood cells with GPU



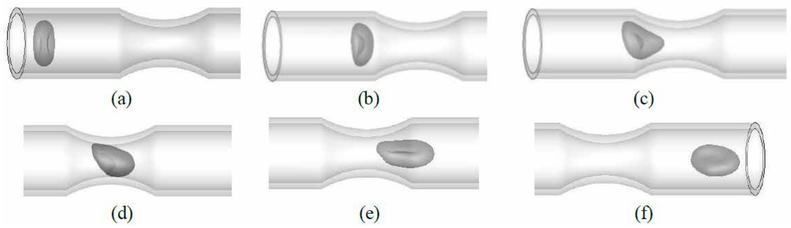
Coarse grained Simulation of Red Blood Cells Stretching Deformation



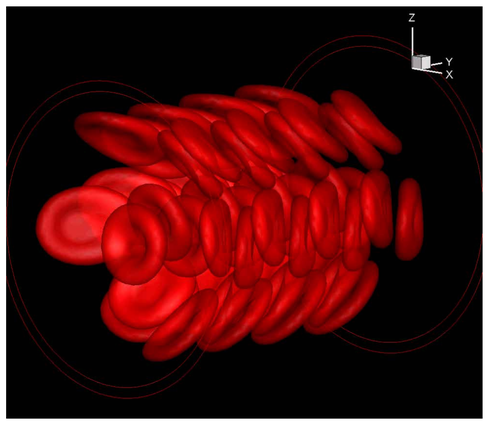
Evolution of a red blood cell passing through a duct



Stretching response for constriction with different size at different time



Evolution of a red blood cell passing through a converging-diverging duct



Simulation of massive red blood cell (ongoing)