Accelerating Particle-Mesh Interaction for Particle-in-Cell Simulation

A. Madonna
Graduate Student, University of Padova

Supervisors:
D. Pavarin
Associate Professor
University of Padova

M. Manente
Senior R&D Engineer
Hit09 S.r.l.
Electric Spacecraft Propulsion

- Low absolute thrust (< 0.1N)
- High Specific Impulse ($I_{sp} > 1000s$)
- Long duration
- Used in deep space missions or in Attitude Control Systems
Plasma Physics 101

Charged Particles

Equation of Motion + Lorentz Eq.

\[
\begin{align*}
\dot{x}_i &= v_i \\
\dot{p}_i &= q_i \left( E(x_i) + \frac{v_i \times B(x_i)}{c} \right)
\end{align*}
\]

Maxwell’s Equations

\[
\begin{align*}
\nabla \cdot B &= 0 \\
\nabla \cdot E &= \frac{\rho}{\varepsilon_0} \\
\nabla \times E &= -\frac{\partial B}{\partial t} \\
\nabla \times B &= \mu_0 \left( J + \varepsilon_0 \frac{\partial E}{\partial t} \right)
\end{align*}
\]
Particle-in-Cell Method

1. Interpolation
2. Particle Push
3. Deposition
4. Solve Fields

Δt
F3MPIC

• Particle-in-Cell simulation package
• Developed at CISAS to engineer Helicon thrusters
• Based on 3D Unstructured mesh
PIC on GPU: Key elements

• Create cell clusters
• Sort particles based on clusters
• Atomic operations for charge accumulation

References:

F3MPIC on the GPU: Problems

- Custom AoS data structures
- External 3rd Party FEM Field Solver
- 3D Tetrahedral mesh:
  - No mesh data locality
  - Undeterministic access pattern
  - Explicit particle tracking required
F3MPIC GPU Modules: Baseline

- Convert data to SoA
- Keep field solve on CPU
- Thread Perspective:
  - Particles vs. Nodes
- One CUDA Stream per species
FGM: First Attempts

Superimpose octree to interpolate

Problem: Not good for charge density

Non-uniform partitioning to use shared memory

Good! We want to generalize this!


Octree domain partitioning

- Built adaptively on GPU (with Dynamic Parallelism)
- Establishes sorting parameter (Z-order node numbering + *Thrust* sort)
- Allows coalesced memory access
- Enables effective shared memory usage
- **Objective**: Leaf node == Thread Block
Octree adaptivity

**Particle based**
- Optimal workload distribution
- Requires rebuilding at every time step
- Inefficient and unpredictable use of per-Block resources
- Particles are millions!
- Octree could get too large!

**Mesh based**
- Built only at simulation start
- Good usage of shared resources
- We are bound by the work of the busiest node
Interpolation & Push

- Load cells to shared memory
  - Octree helps caching!
- Coalesced load of particles
- Porting pusher code is straightforward
Particle Tracking

- **Geometrically consistent algorithm**

- **Already proven on GPU**
  N. Brünggel, *Lagrangian Particle Tracking on a GPU*, Lucerne University of Applied Sciences and Arts, 2011

- **Adapted for F3MPIC**
  custom mesh format
Deposition

- Load cells to shared memory
  - Octree helps caching!
- Coalesced load of particles
- Different outcomes possible
  - Node-based Thread Blocks reduce warp divergence!
- Charge accumulation with atomic operations
Octree benefits

DATA LOCALITY (Classic):
- Allows sorting → Coalesced Access
- Enables shared memory usage

WORKLOAD MANAGEMENT:
- Balances workloads
- Reduces warp divergence
- Helps caching behavior

FLEXIBILITY:
- Opens to new functionalities

The Octree is the algorithm backbone!
FGM: Algorithm

**SETUP**
- Initialize Simulation
- Upload Mesh
- Domain Partitioning
- Upload Particles

**TIME LOOP**
- Field Solver
- Update EM
- Tree Traversal
- Sort Particles

- Download Charge Density
- Inject New Particles
- Tracking & Deposition
- Interpolate & Push

\[ \Delta t \]
FGM: Accuracy

F3MPIC

F3MPIC + FGM

- New tracking algorithm
- FMAD on GPU

Charge conservation check
Global: sum of charges 1.231251e-10 C, integral of charge density 1.232762e-10, error 0.122737%
Species 0: sum of charges -1.383879e-11 C, integral of charge density -1.385641e-11, error -0.127917%
Species 1: sum of charges 1.369638e-10 C, integral of charge density 1.371326e-10, error 0.123250%

Charge conservation check
Global: sum of charges 1.230899e-10 C, integral of charge density 1.230899e-10, error 0.000000%
Species 0: sum of charges -1.364573e-11 C, integral of charge density -1.364572e-11, error -0.000078%
Species 1: sum of charges 1.267356e-10 C, integral of charge density 1.267356e-10, error 0.000000%
FGM: Performance

Total interaction times

FGM Breakdown

CPU: Intel Core i7-4770 @ 3.4GHz (Haswell microarchitecture)
GPU: NVIDIA GeForce GTX 780
FGM: Performance

Interpolation & Push

Tracking & Deposition

CPU: Intel Core i7-4770 @ 3.4GHz (Haswell microarchitecture)
GPU: NVIDIA GeForce GTX 780
FGM: Flexibility

- Monte-Carlo Collisions (MCC)
- Not just for plasma: direct particle interaction
  - Octree sorting → Broad phase collision detection
  - Narrow phase detection and contact algorithms: NNS, SPH, ...
- Agnostic to Field Solver
  - FEM
  - CFD
  - Discontinuous Galerkin (DG) – could be on GPU, too!
- Tracking works for polyhedral cells
- Scalable to Multi-GPU (Hilbert curve node numbering?)
FGM: Exploitation

- Combustion with reagent atomization
- Multiphase flows (e.g. Droplets, etc.)
- Erosion phenomena
- Magneto-fluid Dynamics
- ...

QUESTIONS?

Contacts:

Alberto Madonna: madonna.alberto@gmail.com
Marco Manente: m.manente@hit09.com
Daniele Pavarin: daniele.pavarin@unipd.it