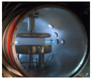
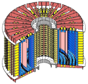
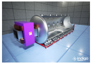


3^{rd} workshop on Gas-filled Detectors and Systems



GPU based transport simulations in gazeous detectors: Uroboros and more...



Samuel Salvador

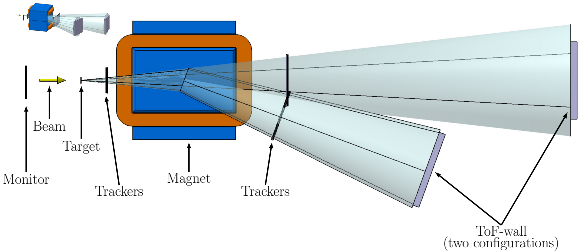
Laboratoire de Physique Corpusculaire de Caen
Normandie Univ, ENSICAEN, UNICAEN, CNRS/IN2P3, LPC Caen, 14000 Caen, France

January 24th 2019

The context

FRACAS: A large acceptance mass spectrometer

- Fragmentation cross sections of ^{12}C
- Targets of medical interest (C, H, O, N, Ca)
- From 100 to 400 MeV/n
- ARCHADE centre around 2023



Designing the gas detectors using simulations

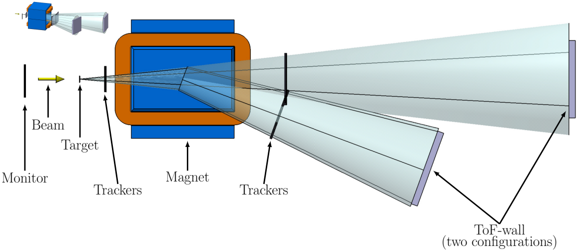
- Technology
- Gap and gas mixture
- Strips or pixels read-out

Accurate and fast MC simulations!

The context

FRACAS: A large acceptance mass spectrometer

- Fragmentation cross sections of ^{12}C
- Targets of medical interest (C, H, O, N, Ca)
- From 100 to 400 MeV/n
- ARCHADE centre around 2023



Designing the gas detectors using simulations

- Technology
- Gap and gas mixture
- Strips or pixels read-out

Accurate and fast MC simulations!

What can be used? (That I know of!)

COMSOL

- Electrostatic and plasma physics

Garfield++

- Pretty well-known and tested
- Does a lot of things (like a lot!)
- Macroscopic and microscopic simulations
  - Microscopic is quite slow! (and serial)

Any macroscopic Monte Carlo simulation self made

- Not really that complicated (uses swarm parameters)
- Particles “follow” the drift lines
- Only effective description of space charge effects

Well, can we achieve a higher throuput?

Microscopic Monte Carlo simulation for particle drift in an electromagnetic field

- Highly parallel simulation
- Based on GP-GPU computing

- Must include most of Garfield features
- Perform at least as good!

Uroboros

"One is the all", third century, Egypt

Well, can we achieve a higher throuput?

Microscopic Monte Carlo simulation for particle drift in an electromagnetic field

- Highly parallel simulation
- Based on GP-GPU computing
- Must include most of Garfield features
- Perform at least as good!

Uroboros

“One is the all”, third century, Egypt

GPU based algorithms: when gamers rule the world

GP-GPU: global programming on graphic processor units

- Originally developed for gaming purposes (obviously)
- Designed to execute rasterization (transforming vectors into pixel values)
- Displays a calculated frame of millions of pixels (4k display: ∼8.3 Mpixels) in <7 ms

High parallelism ⟹ around 80k concurrent threads (on a Titan V100)

CUDA++: nVidia GPU language

- Based on C++
- Needs some training and re-thinking to code for parallelism
- Attached strictly to hardware ⟹ very efficient!
- Large amount of libraries available (cuBLAS, cuFFT,...)

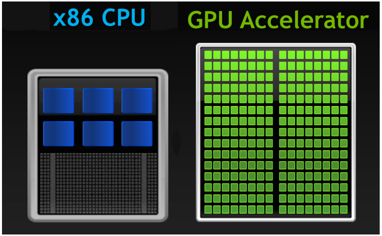
GPU based algorithms: when gamers rule the world

GP-GPU: global programming on graphic processor units

- Originally developed for gaming purposes (obviously)
- Designed to execute rasterization (transforming vectors into pixel values)
- Displays a calculated frame of millions of pixels (4k display: ~8.3 Mpixels) in <7 ms

High parallelism ⟹ around 80k concurrent threads (on a Titan V100)

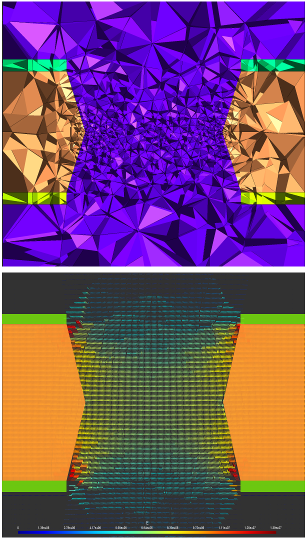
CUDA++: nVidia GPU language



- Based on C++
- Needs some training and re-thinking to code for parallelism
- Attached strictly to hardware ⟹ very efficient!
- Large amount of libraries available (cuBLAS, cuFFT,...)

Uroboros: What does it do?

- Beam tracks: γ, e, p, α, C
  - Analytically deposited energy (using Landau-Vavilov limit and δ-rays)
- or point-like sources
  - Thermal initial energy with N numbers of primary particles
- 2D or 3D calculations (for memory space concerns)
- Generate simple analytic fields or loads field maps
  - BEM or FEM generated
  - "Ramo" field maps (for each electrode) $\Rightarrow$ Signals generation
- Detector geometry
- 3D periodicity



Uroboros: How?

Complete microscopic level physics

- Gases available: Ar, CO_2, N_2, CF_4, CH_4, iC_4H_{10}, O_2
- Interaction cross sections for mixture gases
- Electron anisotropy scattering for nonpolar molecules[1]:

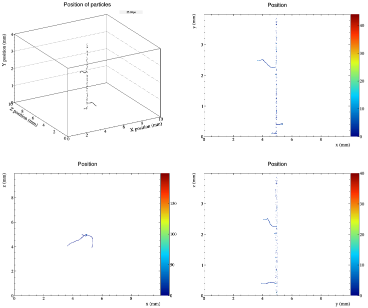
$$\theta = cos^{-1}\left(1 - \frac{2R(1-\xi)}{1+\xi(1-2R)}\right),$$

$$where \quad \xi(E) = f\left(\frac{\sigma m}{\sigma}\right)$$

otherwise, screened-Coulomb scattering:

$$\theta = cos^{-1}\left(1 - \frac{2R}{1+8\frac{E}{E_0}(1-R)}\right).$$

- Penning transfer probability
- Three different integration algorithms: Euler, Leap-Frog or PEFRL[2] (δt=25 fs)

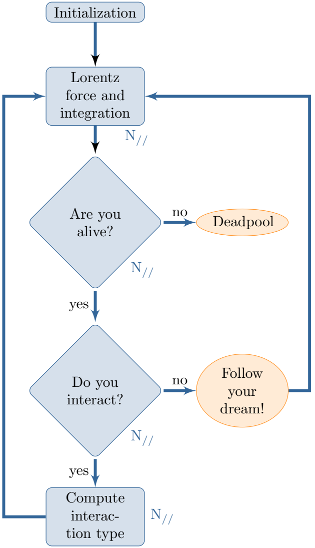


---

[1] A. Okhrimovskyy *et al.*, Phys. Rev. E, 65, 037402

[2] I.P. Omelyan *et al.*, Comp. Phys. Comm., 146, 188-202, 2002

Uroboros: A bit of algorithmics



Initialization

Lorentz force and integration

$N_{//}$

Are you alive?

no

Deadpool

$N_{//}$

yes

Do you interact?

no

Follow your dream!

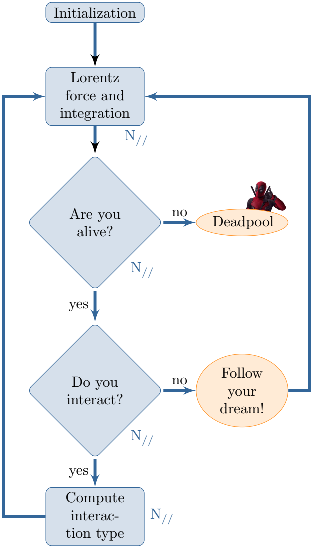
$N_{//}$

yes

Compute interaction type

$N_{//}$

Uroboros: A bit of algorithmics



Initialization

Lorentz force and integration

$N_{//}$

Are you alive?

no

Deadpool

$N_{//}$

yes

Do you interact?

no

Follow your dream!

$N_{//}$

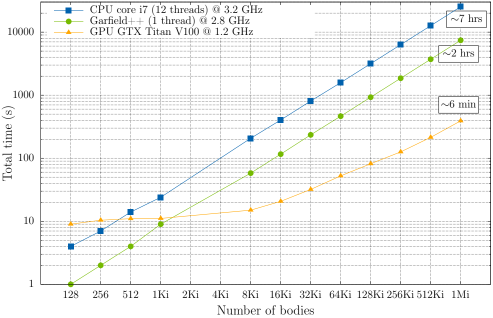
yes

Compute interaction type

$N_{//}$

Speed-up

CPU vs. GPU comparison



- PPAC in 50 mbar Ar/CO_2 80/20 with 1.6 mm gap
- Drift of *N* electrons
- Almost same CPU/GPU code
- Simple microscopic drift in Garfield
- Comparison of total computation time
- 19 times faster than Garfield!

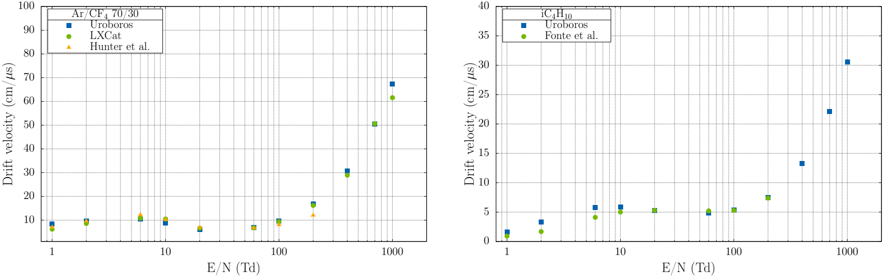
Allows for systematic studies with MC simulation for high gains

You're sure it works?

Comparisons with swarm parameters from the literature and LXCat

LXCat (BOLSIG+): Online Boltzmann equation solver for low-temperature plasmas[1]

- Drift velocities



---

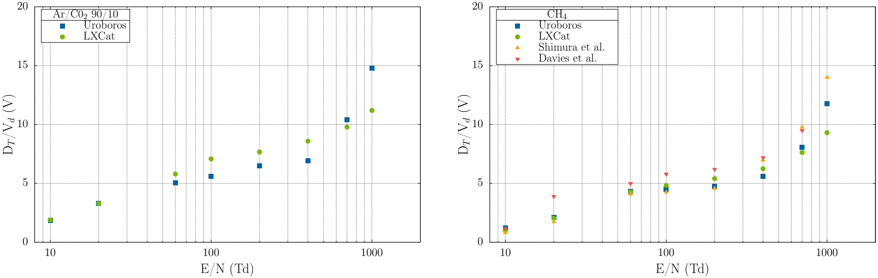
[1] G.J.M. Hagelaar and L.C. Pitchford, "Solving the Boltzmann equation to obtain electron transport coefficients and rate coefficients for fluid models", Plasma Sci Sources and Tech 14, 722 (2005).

Edgar Barlerin's master thesis

You're sure it works?

Comparisons with swarm parameters from the literature and LXCat

- Diffusion coefficient

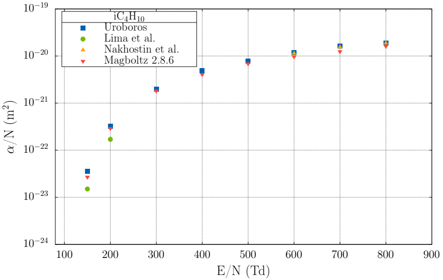


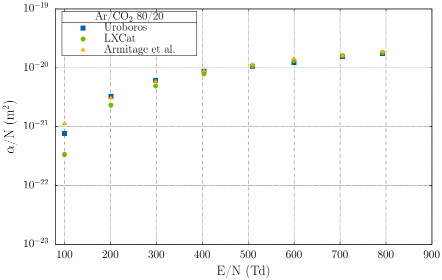
Edgar Barlerin's master thesis

And?

Comparisons with swarm parameters from the literature and LXCat

- First Townsend coefficient α

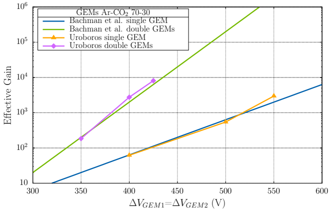




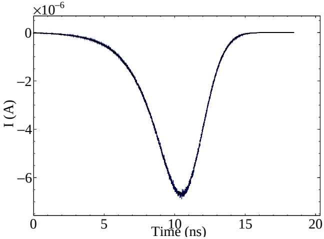
Some discrepancies but overall good agreement with experimental data

That's it?

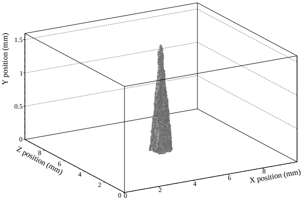
Gain comparison for GEM detectors



Signal generation

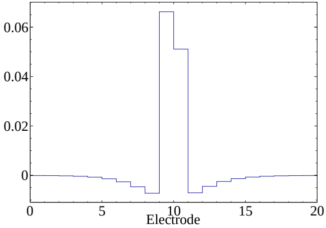


Avalanche generation



^{12}C 200 MeV/n in PPAC 1.6 mm gap, iC_4H_{10}

Position reconstruction



Intermission

Let's have a (short) break...

What's next?

Include real time space charge effects!

Next level programming

- Use an element method (FEM of BEM) to calculate the field
- Include particles (electrons and ions) as charge density distributions in Poisson equation
- Re-calculate the field every N-steps

Why is that challenging?

- An avalanche contains about 10^6 electrons
- Nearly impossible (for now) to include individual charges
- Use approximation method such as Barnes-Hut or FMM
- Need to include octree in the code (not so difficult...)
- Re-calculate the field with the new method
- Iterate!

What's next?

Include real time space charge effects!

Why?

- Very high intensity beams
- Design for high gain detectors
- Ions charging up in specific detectors

Next level programming

- Use an element method (FEM of BEM) to calculate the field
- Include particles (electrons and ions) as charge density distributions in Poisson equation
- Re-calculate the field every N-steps

Why is that challenging?

- An avalanche contains about 10^6 electrons
- Nearly impossible (for now) to include individual charges
- Use approximation method such as Barnes-Hut or FMM

What's next?

Include real time space charge effects!

Next level programming

- Use an element method (FEM of BEM) to calculate the field
- Include particles (electrons and ions) as charge density distributions in Poisson equation
- Re-calculate the field every N-steps

Why is that challenging?

- An avalanche contains about 10^6 electrons
- Nearly impossible (for now) to include individual charges
- Use approximation method such as Barnes-Hut or FMM
- Need to include octree in the code (not so difficult...)
- Re-calculate the field with the new method
- Iterate!

What's next?

Include real time space charge effects!

Next level programming

- Use an element method (FEM of BEM) to calculate the field
- Include particles (electrons and ions) as charge density distributions in Poisson equation
- Re-calculate the field every N-steps

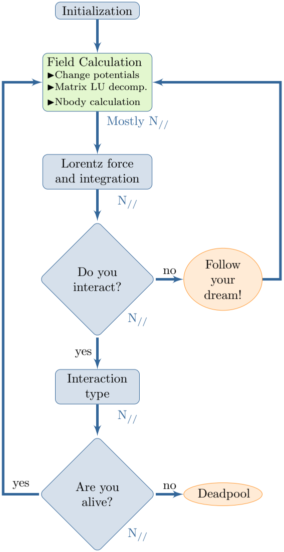
Why is that challenging?

- An avalanche contains about 10^6 electrons
- Nearly impossible (for now) to include individual charges
- Use approximation method such as Barnes-Hut or FMM
- Need to include octree in the code (not so difficult...)
- Re-calculate the field with the new method
- Iterate!

So, what did we do about it?

Ouroboros BEM edition!

- Build (or read) the geometry using predefined shapes
- BEM to calculate the electric field
- Fill matrix with cell properties
- Change each cell potential according to the floating charges
  - Exact method (brute-force)
- Solve problem with LU decomposition (80% time consuming)
- Add the electric field generated by the charges
  - Exact or approx. method (Barnes-hut)



Ouroboros_BEM

Drawbacks

- Quite slow (10 times slower)
- Memory limitation on # of cells in the geometry (12 GB = ~12000 cells)
- Cannot use geometry symmetries

Remaining work

- Dynamic mesh refinement
- Dielectric material properties
- Ramo electric signals