Simulating radiation from Laser-wakefield accelerators

Alexander Debus, Richard Pausch, René Widera, Michael Bussmann, Ulrich Schramm
Acknowledgements

PIConGPU

Michael Bussmann¹, Florian Berninger¹, Heiko Burau¹, Thomas E. Cowan¹, Alexander Debus¹, Axel Hübl¹, Thomas Kluge¹, Richard Pausch¹, Ulrich Schramm¹, René Widera¹

Wolfgang Hönig², Guido Juckeland², Wolfgang Nagel², Felix Schmitt²

Laser wakefield experiments

Arie Irman¹, Axel Jochmann¹, Jurjen Couperus¹

¹ Helmholtz-Zentrum Dresden-Rossendorf
² Zentrum für Informationsdienste und Hochleistungsrechnen
Laser-wakefield accelerators can accelerate electrons to the GeV level and are promising for compact x-ray sources of high brilliance.

- High-power lasers generate plasma waves in underdense plasmas.
- Into these plasma waves electrons can be (self-) injected and over mm- distances accelerated to MeV to GeV energies.

Multi-100 TW Laser system
Pulse <50fs
Plasma radiation has the potential of being a quantitative comparison between theory and experiment.

- Highly nonlinear processes make laser-accelerated electrons challenging to control.
- Especially diagnostics and control of the electron (self-)injection process is necessary.
- Electron spectra are measurements after the fact. (Currently best source for Theory vs. Experiment comparisons.)
- Radiation spectra tell us more on the processes during laser-plasma interaction.

**Spectra**

<table>
<thead>
<tr>
<th>Energy (MeV)</th>
<th>Density (cm⁻³)</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>2.7 x 10¹⁹</td>
</tr>
<tr>
<td>54</td>
<td>2.6 x 10¹⁹</td>
</tr>
<tr>
<td>115</td>
<td>2.3 x 10¹⁹</td>
</tr>
<tr>
<td></td>
<td>2.2 x 10¹⁹</td>
</tr>
<tr>
<td></td>
<td>2.1 x 10¹⁹</td>
</tr>
<tr>
<td></td>
<td>2.0 x 10¹⁹</td>
</tr>
<tr>
<td></td>
<td>1.8 x 10¹⁹</td>
</tr>
</tbody>
</table>

\[ E_p = 144 \text{ MeV} \]
\[ \Delta E/E_p = 12\% \]

2.3 J, 25 fs, \( I_{peak} : 2.4 \times 10^{19} \text{ W cm}^{-2} (a_0=3.4) \]
Detecting radiation is straightforward in experiment, but poses a challenge for computation.

- Physics: Calculate radiation and characterize signatures from electron injection and betatron oscillations.

- Goal: Calculate radiation from ALL electrons with full angular characteristics.

Radiation codes as post-processing after PIC quickly become impracticable.

CPU-PIC: 200 TB raw particle data within 3 weeks.

- EM fields from the PIC grid do not have enough spatial resolution for X-Ray radiation arising from large Doppler-shifts.

- Approach here: Liénard-Wiechert potentials → Calculate from trajectories.
PIConGPU - a GPU driven 3D-PIC code provides the basis.

3D3V relativistic particle-in-cell code

Villasenor-Buneman charge conserving current deposition

NGP / CIC / TSC macro-particle distribution functions

Boris Push particle pusher

Yee-scheme / Directional splitting Maxwell-Solver
64 NVIDIA Fermi GPUs

45 min

0.09 seconds per time step

FORTRAN LEGACY

128 AMD cores

8 days

480x480x3070 Cells, 2.8 x 10^9 macro particles, 30,000 time steps
Strong and Weak Scaling on NVIDIA TESLA M2090

Scaling PIConGPU 3D on TESLA M2090

weak scaling: 28 mill. particles, 192x192x192 cells per GPU
strong scaling: 535 mill. particles 512x512x512 cells
Radiation calculations are highly parallel!

Spectral energy density

\[
\frac{d^2 I}{d\omega \, d\Omega} = \frac{e^2}{4\pi^2 c} \left| P \cdot \sum_j \int_{-\infty}^{+\infty} \frac{n \times [n - \beta_j \times \dot{\beta}_j]}{(1 - \beta_j \cdot n)^2} e^{i \omega (t - n \cdot r_j(t)/c)} \right|^2
\]

Radiation amplitude for each particle at each time step, each observation direction and each frequency.

- \( n \) ... normalized observation vector  (Yes, this is a far field solution!)
- \( j \) ... particle number
- \( \beta_j \) ... normalized particle velocity
- \( r_j \) ... particle position
- \( P \) ... Polarisation filter vector ( (1,1,1) accepts all polarisations )

Walter D. Jackson, *Classical Electrodynamics*, 3rd ed. (John Wiley & Sons, 1999), Spectral energy density
Discrete Fourier transform (DFT) is a good choice for GPU implementations.

- For FFT the trajectories of the particles need to be kept in memory, but not for DFT.

- DFT does not require interpolation of velocities and accelerations to equidistant retarded times.

- From the physics side DFT is more flexible, because frequencies can be chosen arbitrarily.

  → Good for small-bandwidth „sky surveys“ for radiation localisation calculations or logarithmic spectra from X-ray to IR wavelengths.

- However: Since resulting spectra at different observation directions have to be stored in CPU memory anyway, FFT is still interesting in a GPU-CPU-hybrid implementation. ( GPU: Amplitudes, interpolation and superposition CPU: Calculating FFTs. )
Strong scaling: radiation code becomes the main activity.

- Strong scaling is near ideal (overhead < 1%)
- 128 spectra à 2048 frequencies; $1.1 \times 10^9$ particles in sim.; 70,000 part./s/GPU
The code is validated using analytic results of Thomson scattering.

- (Nonlinear) Thomson scattering from is a test for relativistic particles in high-intensity laser fields against analytic results.
- The core implementation of radiation code with the numerics was validated independently from the PIC-Code using predefined trajectories.
- The following tests (single particles and plasma) target the integration into PIConGPU, correct laser initialisation and numeric issues of the PIC code rather than the radiation code.
Nonlinear Thomson scattering with relativistic single particles

**Analytic solution**

- Peak laser intensity $I_0 = 4.9 \cdot 10^{18} \text{ W/cm}^2$ \hspace{1cm} ($a_0 = 1.5$)

---

**Theory:**
Esarey *et al.*, Nonlinear Thomson scattering of intense laser pulses from beams and plasmas
Nonlinear Thomson scattering with relativistic single particles

**Numeric** solution with PIConGPU
Nonlinear Thomson scattering in plasma

**Analytic solution**

- Peak laser intensity $I_0 = 8.7 \cdot 10^{18} \text{ W/cm}^2 \quad (a_0 = 2.0)$

---

**Theory:**
Esarey *et al.*, Nonlinear Thomson scattering of intense laser pulses from beams and plasmas
Nonlinear Thomson Scattering in plasma

**Numeric** solution with PIConGPU

Radiation GPU - logarithmic

Radiation GPU - linear
Very preliminary results: Radiation from Laser-wakefield accelerators

- 30 min without radiation $\rightarrow$ 23h 15min with 128 spectra à 2048 frequencies
- 66 Mio particles on 8 GPU; 10% of all macro particles radiate

Disclaimer:
Wrong resolution due to lack of GPUs,
This is a test, not physics!

For full resolution $\sim$1000 GPUs are required.
Very preliminary results: Laser-wakefield accelerator

368.00 fs

Radiation GPU - logarithmic

Radiation GPU - linear

$\theta^\ast$
Conclusions

• The GPU-based 3D-PIC code PIConGPU provides the basis for full-scale radiation calculations using all simulated particles.

• For computing the radiation on GPU minimal memory footprint a discrete Fourier approach has been implemented.

• Benchmarks for strong scaling and physics tests for the integration of the radiation code into PIConGPU have been shown.

• First Laser-wakefield radiation tests provide proof-of-principle.

Near future:

• Full scale simulation LWFA PIC-simulation with more GPUs.

• FFT-based GPU-CPU Hybrid code (CPU provides memory and FFT) uniformly spaced spectral data in many directions over the full solid angle.
Thank you!
“Almost all Programming can be viewed as an Exercise in Caching”
Everything depends on Data Structures PART 1 — Cell-based Data

Super Cell

Each thread in a thread block corresponds to one cell in the super cell
Everything depends on Data Structures PART 2 — Particle Data

Attribute Frame (the data structure formerly known as „tile“, „pool“, ....)

Each thread in a thread block is started for one particle in the frame

Cell index tells you the particle’s local cell index inside the super cell
Encapsulation of all memory transfer functions

GPU1

cudaMemcpy, MPI_ISEND
asynchronous
any data (particle, cell)

GPU2

Local Domain Model
a thread can assume that all the data it needs is in its local memory (SISD)
Events help to direct kernel execution, data exchange, etc.

**EventTask tree**

1

1

1

1

<table>
<thead>
<tr>
<th>Depends on</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parallel to</td>
</tr>
</tbody>
</table>

**Parallel hierarchy types**

- TASK_HOST (not parallel, limitation of cuda runtime)
- TASK_CUDa (use streams for parallel work on gpu)
- TASK_MPI
Defining the algorithmic structure of a task graph

1. getSizeFromDevice();  // memcpy from device
2. EventTask e_split = __getTransactionEvent();
   __startTransaction( e_split );  // depend from e_split
   cudaKernel_X();  // create a kernel task
3. EventTask e1 = __endTransaction();
   __startTransaction( e_split );
   cudaKernel_Y();
4. synchronizeWithNeighbor();  // memcpy+mpi task
   EventTask e2 = __endTransaction();
   __setTransactionEvent( e1+e2 );  // combine parallel work
5. cudaKernel_Z();
Communicate and compute concurrently using Tasks

- Compute Current
  - Compute electric Field
  - Compute magnetic Field
- Exchange Current
  - Exchange electric Field
  - Exchange magnetic Field
Strong and Weak Scaling on NVIDIA FERMI

Strong and Weak Scaling PIConGPU 3D

Tesla S2050

minimum of 1.11 ns per particle and time step

3D Villasenor-Buneman (CIC)
Boris-Push
Yee-Lattice
LWFA of electrons