

Large Scale Particle-in-Cell Simulations of Laser-Plasma Interactions Relevant to IFE

Frank S. Tsung, W. B. Mori, B. J. Winjum, T. Grismayer,
V. K. Decyk UCLA

- Motivation/Description of the problem.
- Typical Problem Size Today.
- What can we do with 1-2 order of magnitude increase in computing power?
- What are we doing with GPU's?

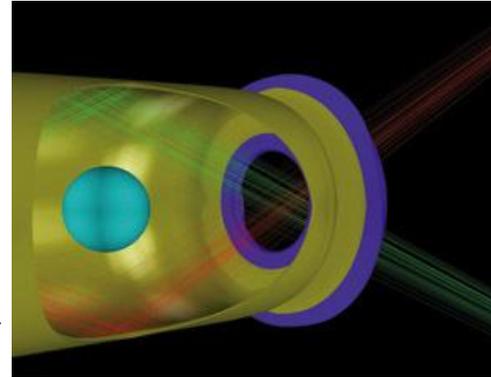
Project Summary -- Large Scale PIC Simulations of LPI's Relevant to Inertial Fusion Energy (IFE)



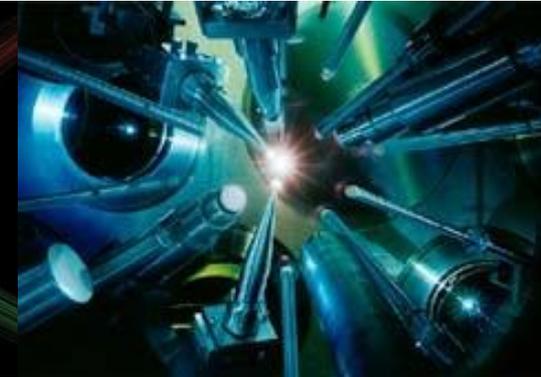
In inertial fusion energy (both ID and DD) laser plasma interactions, where the incident laser decays parametrically into two daughter waves degrade implosion.

They can be:

- SRS: where the incident laser decays into a backward going laser and a forward going plasma wave
- 2 plasmon: the laser decays into two plasma waves (near the quarter critical surface)

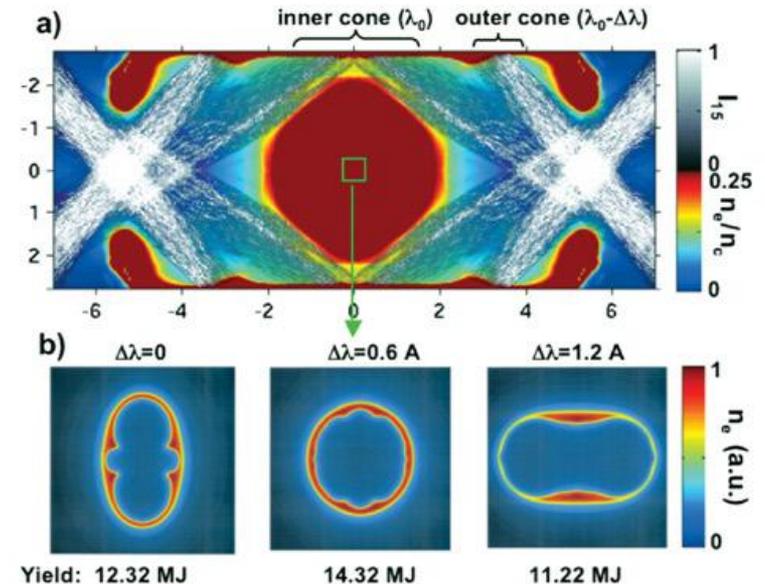


indirect drive (e.g. NIF)



direct drive (e.g. Omega @ U.Rochester)

- LPI can degrade IFE in 2 ways:
 - laser reflection
 - generation of fast electrons which pre-heat the core and degrades implosion.
- Recent NIF results show the control of LPI (in that case, seeded SBS from crossing beams) can be important in maintaining implosion symmetry. At full power, seeded & coupled SRS can effect implosion symmetry in an analogous way.



PIC simulations: OSIRIS framework

osiris
v2.0

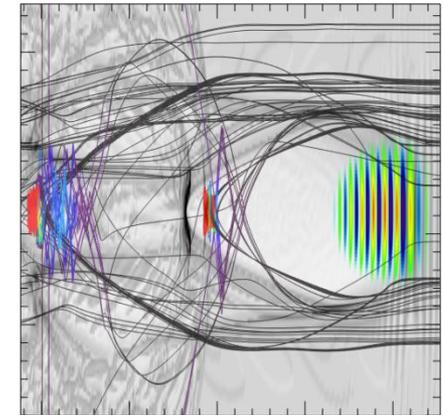
osiris framework

- Massively Parallel, Fully Relativistic Particle-in-Cell (PIC) Code
- Local FDTD field solver
- Visualization and Data Analysis Infrastructure (viz_xd)
- Strong scaling to at least 5000 processors
- Developed by the osiris.consortium
- [®] UCLA + IST + USC

Jaguar (Cray XT5)

Cores	Timing (secs)	Speedup	Efficiency (%)
1024	2793.7	1024.0	100.0
2048	1499.2	1908.2	93.2
4096	779.8	3668.4	89.6
8192	449.8	6360.6	77.6
16384	222.7	12844.0	78.4
32768	127.5	22447.0	68.5
65536	71.3	40151.0	61.3

In Recent Strong Scaling Studies, OSIRIS is shown to be >80% efficient on ~300k cores on the BlueGene Supercomputer Jugene, > 60% efficient on >64k cores @ Jaguar (Cray XT5) (>97% efficient via weak scaling)



New Features in v2.0

- Bessel Beams
- Binary Collision Module
- Tunnel (ADK) and Impact Ionization
- Dynamic Load Balancing
- Higher Order Shape Functions
- Perfectly Matched Layer (PML)
- Parallel I/O



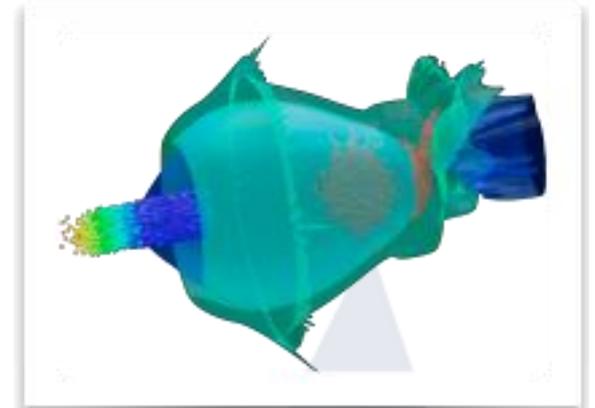
UPIC: UCLA Particle-in-Cell Framework

Features of UPIC:

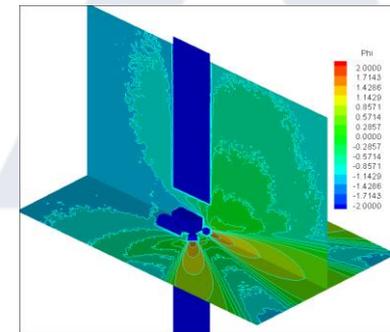
- Provides trusted components for rapid construction of new parallel PIC codes (You-PICK)
- Support multiple physics models, levels of accuracy, optimizations, computer architectures.
- Supports both MPI and threaded programming models.
- Hides parallel processing by reusing communication patterns: Physicists only need to know the data layout.
- Components used in wide variety of applications: Magnetic Fusion, Space Physics, **Plasma Accelerators (QuickPIC)**, Cosmology, Quantum Plasmas, Ion Propulsion (DRACO).

(V. K. Decyk, Comp. Phys. Comm. **17**, 95 (2007).)

Recently a subset of UPIC (2D ES PIC) has been ported to the GPU, and we will show some preliminary results and discuss the move to new multi-core architectures.



QuickPIC: Plasma Accelerators
(C. K. Huang, *et al*)



DRACO: Ion Propulsion
(J. Wang, *et al*)

2. Current HPC Requirements (2D two plasmon study with overlapping beams)

- Architectures
 - Franklin/Cray XT4
- Compute/memory load
 - 2,048 cores, $\sim >1\text{TB}$ total memory, 100 wallclock hours total (8 restarts), roughly 0.6 GB/core, 200,000 core hours/run
- Data read/written
 - 4TB simulation data per simulation
- Necessary software, services or infrastructure
 - HDF5/MPI
- Current primary codes and their methods or algorithm
 - OSIRIS/fully explicit, relativistic EM PIC
- Known limitations/obstacles/bottlenecks
 - OSIRIS has shown good scaling for up to $\sim 300,000$ cores, we do not expect any obstacles in the near future.
- Temporal scales:
 - Laser period = 1fs
 - Growth time $\sim 1\text{ps}$
 - Simulation time $\sim 10\text{ps}$
 - ~~Pulse Duration 20ns~~
- Spatial Scales:
 - Debye Length $\sim 10^{-2}-10^{-1}$ microns
 - Laser wavelength $\sim 10^{-1}$ microns
 - Laser hotspot width $\sim 3-5$ microns
 - speckle length ~ 100 microns
- Simulation Size:
 - $100 \mu\text{m} \times 30 \mu\text{m}$ (7,500 x 2,222 cell)
 - 4 billion particles
 - $\sim 300,000$ simulation steps

2. Future HPC Requirements (3D 2 plasmon Study)

- There will be interesting physics in the next 3-5 years with a 1 or 2 order of magnitude increase in computing power
 - 1 --> interaction of multiple beams (>2)
 - 2 --> full 3D simulations of two interacting beams (described below, normalized to XT4 hours)
- Compute/memory load
 - 100,000 cores, ~>750 TB total memory, 500 wallclock hours roughly 8-9 GB/core (50 million core hours/run)
- Data read/written
 - 1000TB simulation data per simulation
- Necessary software, services or infrastructure
 - HDF5/MPI
- Temporal scales:
 - Laser period = 1fs
 - Growth time ~1ps
 - Simulation time ~ 10ps
 - Pulse Duration 20ns
- Spatial Scales:
 - Debye Length ~ 10^{-2} - 10^{-1} microns
 - Laser wavelength ~ 10^{-1} microns
 - Laser hotspot width ~3-5 microns
 - speckle length ~ 100 microns
- Simulation Size:
 - 100 μm x 30 μm x 9 μm (7,500 x 2,222 cell x 750 cells)
 - 3 trillion particles
 - ~100,000 simulation steps

Timing Results on GPU's

2D ES Benchmark with 256x512 grid, 4,718,592 particles, 36 particles/cell, dt = .025

NVIDIA GTX 280 compared to the 2.66 GHz Intel Nehalem Host:

NVIDIA Tesla (C1060) compared to the 2.66 GHz Intel Nehalem Host:

Deposit

- **GTX 280:** 0.21/0.24 nsec/particle/time step (cold/hot), a speedup of **40/36(cold/hot)**.
- **Tesla:** 0.23/0.25 nsec/particle/time step, a speedup of **37/34**.

Push

- **GTX 280:** .53/.73 nsec/particle/time step, a speedup of **35/26**.
- **Tesla:** .56/.77 nsec/particle/time step, a speedup of **33/25**.

Total Particle Time (excluding field solvers which takes a small %):

- **GTX 280:** .78/1.67 nsec/particle/time step, a speedup of **33/17**.
- **Tesla:** .82/1.83 nsec/particle/time step, a speedup of **30/15**.

- Problem areas:
 - Very difficult to debug, emulator not very faithful.
 - Not yet parallel (using MPI)
- To debug, we run a Fortran code on the host simultaneously.
 - We can run either the CUDA or Fortran routine at any point
 - Copy out from CUDA and compare

Future looks very promising.

- The lesson learned in developing GPU codes can be implemented on traditional CPU's.
- Models which contain more calculations per memory read, such as EM/PIC or GK/PIC, should achieve better speedup on the GPU.
- Software development should improve in future
 - Emerging standards should help: OpenCL , co-Array Fortran.
 - Non-standard features and extra manual labor should disappear.
 - More libraries becoming available: BLAS, FFT, CUDPP

2. HPC Usage and Methods for the Next 3-5 Years (Overlapping beams in 2D or 3D SRS)

- Upcoming changes to codes/methods/approaches
 - We have begun to port UPIC to multi-core architectures such as the GPU (in upcoming slides)
- Changes to Compute/memory load
 - In the next 3-5 years, we plan to study the effects of the interaction of multiple (>2) beams, and 3D effects, which can be 10x to 250x times larger than current simulations.
- Changes to Data read/written
 - OSIRIS now uses parallel HDF5
- Changes to necessary software, services or infrastructure
 - How will restart be handled when the system size is > 100TB?
- Anticipated limitations/obstacles/bottlenecks on 10K-1000K PE system.
 - OSIRIS has shown good (>80%) strong scaling for 300k processors, restart can be an issue (writing 100TB can be costly)

Table 1: Warm plasma with $v_{th} \cdot dt = 0.025$

	Intel Nehalem (ns)	Tesla C1060 (ns)	GTX 280 (ns)
Push	18.6	0.67	0.64
Deposit	8.7	0.25	0.24
Sort	0.4	0.29	0.26
Total	27.7	1.21	1.13

Table 2: “Hot” plasma with $v_{th} \cdot dt = 0.1$

	Intel Nehalem (ns)	Tesla C1060 (ns)	GTX 280 (ns)
Push	18.9	0.77	0.73
Deposit	8.7	0.26	0.24
Sort	0.4	0.81	0.70
Total	28	1.83	1.67

Table 3: Frozen (asymptotic) plasma with $v_{th} \cdot dt = 0.0$

	Intel Nehalem (ns)	Tesla C1060 (ns)	GTX 280 (ns)
Push	18.6	0.56	0.53
Deposit	8.5	0.23	0.21
Sort	0.4	0.04	0.04
Total	27.5	0.82	0.78