A FAST PARTICLE TRACKING TOOL FOR THE SIMULATION OF DIELECTRIC LASER ACCELERATORS

F. Mayet[1]*, R. Assmann, U. Dorda, W. Kuropka[1], DESY, Hamburg, Germany
[1]also at University of Hamburg, Germany

Abstract

In order to simulate the beam dynamics in grating based Dielectric Laser Accelerators (DLA) fully self-consistent PIC codes are usually employed. These codes model the evolution of both the electromagnetic fields inside a laser-driven DLA and the beam phase space very accurately. The main drawback of these codes is that they are computationally very expensive. While the simulation of a single DLA period is feasible with these codes, long multi-period structures cannot be studied without access to HPC clusters.

We present a fast particle tracking tool for the simulation of long DLA structures. DLATracker is a parallelized code based on the analytical reconstruction of the in-channel electromagnetic fields and a Boris/Vay-type particle pusher. Its computational kernel is written in OpenCL and can run on both CPUs and GPUs. The main code is following a modular approach and is written in Python 2.7. This way the code can be easiliy extended for different use cases.

In order to benchmark the code, simulation results are compared to results obtained with the PIC code VSim 7.2.

INTRODUCTION

The concept of dielectric laser accelerators (DLA) has gained increasing attention in accelerator research, because of the high achievable acceleration gradients (~GeV/m) [1]. This is due to the high damage threshold of dielectrics at optical frequencies. In order to simulate the interaction of the incoming electron bunch with the electromagnetic fields inside the laser illuminated dielectric structure self-consistent Particle-In-Cell (PIC) codes are usually employed. This way both the evolution of the fields inside the acceleration channel and the beam dynamics can be simulated very accurately. The main drawback of PIC codes is that they can be computationally very expensive and thus are usually used on HPC clusters. In this work we focus on so called grating type DLAs (see Fig. 1). In the context of the Accelerator on a CHip International Program (ACHIP) the typical period length of a grating DLA is 2 micron and the channel width <1 micron. The in-channel fields can be decomposed into an infinite sum of so called *spatial harmonics* (see section *Theoretical Background*) [2]. In order to resolve higher harmonic contributions to the field, a sufficiently high spatial grid resolution has to be used. Together with the well-known *Courant-Friedrichs-Lewy* stability condition [3] for time domain algorithms

$$\frac{c_0 \Delta t}{\sqrt{\Delta x^2 + \Delta y^2 + \Delta z^2}} < 1 \tag{1}$$

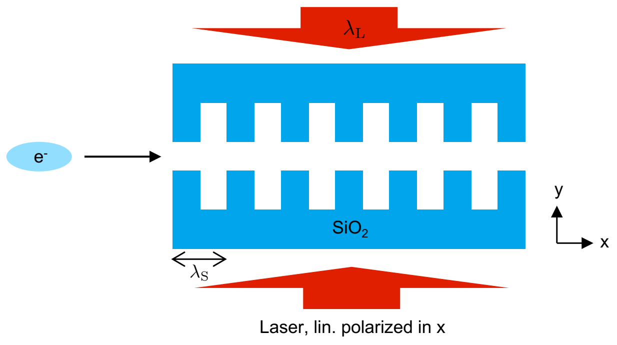

Figure 1: Schematic of a dual grating DLA illuminated from both sides with a linearly polarized laser field with wavelength λ_L. λ_S is the period length, which is connected to the laser wavelength by the *synchronicity condition* $\lambda_S = \beta\lambda_L$.

this results in typical time steps Δt in the order of $< 10^{-16}$ s. In addition to that - as of now - standard PIC codes cannot exploit the periodic nature of the DLA fields. In sum this results in computationally very expensive simulations, especially for long structures (100s of periods) or even whole DLA beam lines.

We present a specialized approach, which can be used to simulate many-period structures within a fraction of the time needed by PIC codes by employing key simplifications and assumptions. We compare results obtained with the code based on this approach to reference simulations done with the PIC code VSim 7.2 [4]. This way the validity of the simplifications and assumptions used in our code is evaluated.

DLATRACKER

In this section the specialized particle tracking code DLATracker is presented. DLATracker is a fully parallelized code written in Python [5] and OpenCL [6] (using PyOpenCL [7]) and implements the well-established Boris/Vay-type particle pusher algorithm [8, 9]. It calculates the external electro-magnetic fields using an analytical description of the in-channel DLA fields. This way the self-consistent FDTD field calculation of the usual PIC cycle is replaced by a computationally inexpensive analytical calculation (see Fig. 2). In addition to that this essentially also eliminates the need for a spatial grid. Field values are not interpolated. In terms of computation time, this simplification is the main advantage of DLATracker in comparison to fully self-consistent PIC codes. Since the computational kernel of DLATracker is written in OpenCL the code can also be

* frank.mayet@desy.de

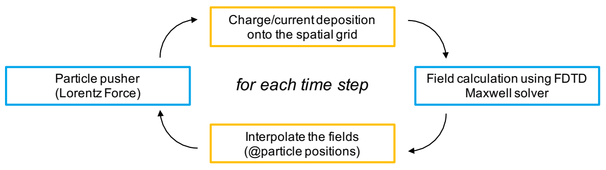
Charge/current deposition onto the spatial grid
Particle pusher (Lorentz Force)
for each time step
Field calculation using FDTD Maxwell solver
Interpolate the fields (@particle positions)

Figure 2: The main PIC loop. DLATracker eliminates both the computationally expensive FDTD field calculation and the use of a spatial grid. This way field values do not need to be interpolated.

executed on GPUs, which can reduce the simulation time by orders of magnitude.

In the following the theory of the analytical treatment is briefly introduced.

Theoretical Background

For the following calculation an in x-direction linearly polarized plane wave which is travelling along the y-direction with a wavelength of λ_0 is assumed. It is incident on a grating perpendicular to the grating structure, which implies a structure being periodic in x-direction. The problem is assumed to be pseudo-2-dimensional in the sense that in z-direction the structure is infinite.

Using *Maxwell's Equations* we see that our plane wave has a non-vanishing magnetic field only in z-direction. The magnetic field after passage of the grating can be described as a composition of an infinite number of *spatial harmonics*, or diffraction modes (cf. *Floquet Theorem* $\longrightarrow$ Fourier series):

$$B_z(x,y,t) = \sum_{n=-\infty}^{\infty} B_{z,0}^{(n)} \cdot e^{i(nk_x x + k_y y - \omega t + \phi_0)}, \qquad (2)$$

where $k_x = 2\pi/\lambda_p$ is the wave vector w.r.t the grating period. The term ϕ_0 is an arbitrary phase offset. Here it describes the particle to laser phase relation. Inserting B_z into the wave equation for vacuum it can be seen that $k_y^2 = k_0^2 - n^2 k_x^2$. Hence

$$B_z(x,y,t) = \sum_{n=1}^{\infty} B_{z,0}^{(n)} \cdot e^{i\left(y\sqrt{k_0^2 - n^2 k_x^2} + nk_x x - \omega t + \phi_0\right)} \qquad (3)$$

Using $\nabla \times \vec{B} = -i \cdot \frac{k_0}{c}\vec{E}$ the x and y components of the electric field can be calculated from B_z. In order to efficiently accelerate a moving particle it has to be phase synchronous with the parallel component of the electromagnetic field. From this it follows that

$$\frac{\omega}{nk_x} = \beta c \Leftrightarrow k_x = \frac{k_0}{n\beta} \Leftrightarrow \beta = \frac{\lambda_x}{n\lambda_0}, \qquad (4)$$

where k_0 is the laser wave number and βc is the particle velocity. Equation (4) is the *synchronicity condition* for grating accelerators. In the following calculations we use $n = 1$ as the synchronous order and hence $k_x = k_o/\beta$.

Inserting k_x into the expressions for the field components and looking only at the real part yields

$$\Re(\vec{B})(x,y,t) = \sum_{n=-\infty}^{\infty} |B_{z,0}^{(n)}| \cdot e^{-\delta^{(n)} y} \cdot \begin{pmatrix} 0 \\ 0 \\ \cos\left(nk_x x - \omega t + \phi_0 + \phi^{(n)}\right) \end{pmatrix},$$

$$\Re(\vec{E})(x,y,t) = \sum_{n=-\infty}^{\infty} |E_{x,0}^{(n)}| \cdot e^{-\delta^{(n)} y} \cdot \begin{pmatrix} \sin\left(n\frac{k_0}{\beta}x - \omega t + \phi_0 + \phi^{(n)}\right) \\ \sqrt{\frac{n^2}{n^2-\beta^2}} \cdot \cos\left(n\frac{k_0}{\beta}x - \omega t + \phi_0 + \phi^{(n)}\right) \\ 0 \end{pmatrix}, \qquad (5)$$

where $\delta^{(n)} = k_0\sqrt{\frac{n^2}{\beta^2} - 1}$ and $|B_{z,0}^{(n)}|$ and $\phi^{(n)}$ are the amplitudes and phases of the complex weights of the spatial harmonics respectively. It can be seen that the field falls off exponentially in y. The particles are accelerated in evanescent field components, which is required by the *Lawson-Woodward Theorem*. Furthermore it can be seen that in order to be able to calculate the field the complex $B_{z,0}^{(n)}$ need to be known. In practice they can be retrieved by doing a Fourier analysis of numerical FDTD field calculations. This method is also routinely employed in other groups (cf. U. Niedermayer et al.).

Assumptions and Simplifications

In contrast to a self-consistent PIC code DLATracker currently uses several assumptions to simplify the computations. As can be seen in the derivation above one major simplification is the assumption of a plane wave excitation. Also the interaction with the material is not taken into account. At the moment only a stopping field is implemented. The code can take the beam space charge into account, but wake fields are currently not implemented.

Implementation

As described above the main code is implemented in Python and the computational kernel in OpenCL. Because of that the code runs on any OpenCL 1.2 capable hardware/OS. In order to run a simulation the user has to supply an input file in YAML [10] format. Currently arbitrary beam lines consisting of vertical or horizontal grating DLAs (defined by their spatial harmonic weights), static fields and drifts can be specified. Particle distributions can be either generated internally or supplied in a propriatary format. In addition to that ASTRA [11] distributions are also supported as input.

Benchmarks

Physics In order to benchmark the validity of the results obtained with DLATracker it is benchmarked against both ASTRA and VSim 7.2. ASTRA is used to test the validity of the beam dynamics results produced by the particle pusher implementation in simple field configurations. VSim 7.2. is used as a benchmark for the actual DLA simulation. In the following an exemplary comparison of longitudinal phase space properties obtained with VSim 7.2 and DLATracker is shown. Figure 3 shows the energy gain of an 50 fC 94 MeV 6D Gaussian electron bunch with a bunch length of 0.5 micron during traversal of a β-matched 13 period grating DLA, which is illuminated with two 2 micron lasers. The plot shows both the mean energy gain of the whole bunch, as well as the energy gain of the central particle of the bunch. It can be seen that DLATracker replicates

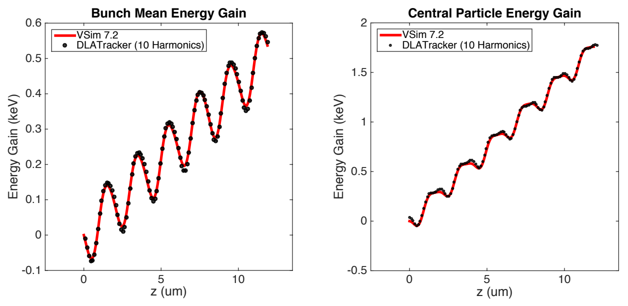

Figure 3: Comparison of the mean and single particle energy gain during traversal of 6 DLA periods.

the VSim results very well. Differences can be attributed to possible inaccuracies of the field reeconstruction. Also in the VSim simulation not only in-channel fields are taken into account. This is an important difference, since inside the dielectric material strong EM fields can be present. The interaction of the material with particles (scattering, etc.) is not taken into account by both codes. From the shown results overall a reasonable agreement between the two codes can be found even in this early stage of the code.

Speed One of the main advantages of DLATracker in comparison to usual PIC codes is its speed. This is mainly due to the key simplifications described above, but also due to the fact that it can run on GPUs. Therefore DLATracker can be used on single workstations and not only in distributed HPC environments. Table 1 and Fig. 4 show speed benchmarks of DLA Tracker. The benchmark shown in Table 1 shows a comparison between VSim and DLATracker, which was conducted on a single workstation. The speed advantage is immediately evident when comparing the two codes on the CPU. Using the GPU further reduces computation time substantially. Figure 4 shows the DLATracker computation time depending on the number of particles. It can be seen that for large numbers of particles the GPU can take advantage of the sheer amount of parallel compute units compared to the 8-core CPU.

CONCLUSION AND OUTLOOK

We have presented an alternative method to conduct grating DLA simulations. The tool is meant to provide a fast means to do quick exploratory studies of long structures

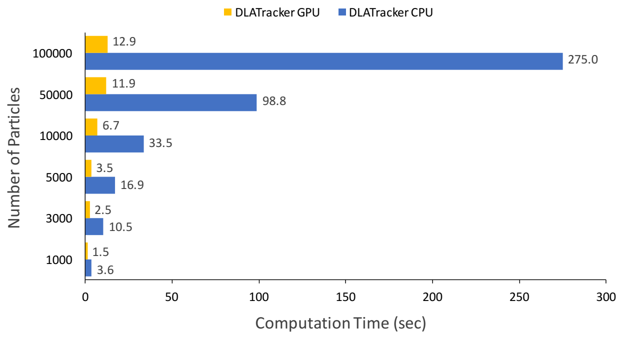

Figure 4: Comparison of the scaling behavior of the CPU and GPU implementation for different numbers of tracked particles respectively (no space charge → embarrassingly parallel).

Table 1: Computation time needed for tracking 5000 particles through a 15 period dual grating DLA at steady state. CPU: 8-Core Xeon E5-1680 3 GHz, GPU: AMD D700. The calculations are carried out with double precision and with same longitudinal resolution / time step.

Code	Space Charge	Time (s)
DLATracker GPU 3D	NO	1.4
DLATracker CPU 3D	NO	6.2
DLATracker GPU 3D	YES	$7.0 \cdot 10^2$
DLATracker CPU 3D	YES	$4.8 \cdot 10^3$
VSim 7.2 2D	YES	$4.3 \cdot 10^3$
VSim 7.2 3D	YES	$1.1 \cdot 10^4$

on single workstations prior to setting up lenghty and potentially expensive PIC simulations. The code is currently work-in-progress and is routinely updated and expanded in functionality. The benchmarks show a good agreement with both ASTRA and VSim, while at the same showing substantial speed advantages on single workstations espacially when GPUs are used.

DLATracker currently simulates an ideal DLA with plain wave excitation. In reality for example laser pulse properties (both spatial and temporal) alter the field properties inside the channel. Also - as already mentioned above - the interaction of the particles with the dielectric material is currently not modeled. The same is true for wake fields. Also so-called chirped gratings for sub-relativistic electrons are currently not supported.

All of the above needs to be adressed in future versions of DLATracker.

ACKNOWLEDGMENTS

This research is funded by the Gordon and Betty Moore Foundation as part of the Accelerator on a Chip International Program. We thank Uwe Niedermayer, who also works on the analytical reconstruction of grating DLA fields, for fruitful discussions and input.

REFERENCES

[1] R. J. England *et al.*, “Dielectric laser accelerators”, *Rev. Mod. Phys.* 86, 1337–1389 (2014).

[2] J. Breuer *et al.*, “Dielectric laser acceleration of electrons in the vicinity of single and double grating structures – theory and simulations”, *J. Phys. B: At. Mol. Opt. Phys.* 47, 234004 (2014).

[3] R. Courant *et al.*, *H. Math. Ann.* (1928) 100: 32.

[4] Tech-X, VSim, available from `www.txcorp.com`

[5] Python Software Foundation. Python Language Reference, version 2.7. Available at `http://www.python.org`

[6] The Khronos Group. OpenCL - The open standard for parallel programming of heterogeneous systems, Specification Version 1.2, available at `http://www.khronos.org`

[7] A. Klöckner *et al.*, “PyCUDA and PyOpenCL: A scripting-based approach to GPU run-time code generation”, *Parallel Computing*, Volume 38, Issue 3, March 2012, Pages 157-174.

[8] J. P. Boris, “Relativistic plasma simulation-optimization of a hybrid code”, *Proc. Fourth Conf. Num. Sim. Plasmas*, Naval Res. Lab., Washington, DC, USA, 3-67, 2-3 November 1970.

[9] J.-L. Vay, “Simulation of beams or plasmas crossing at relativistic velocity”, *Physics of Plasmas* (1994-present), 15(5):056701, 2008.

[10] C. Evans *et al.*, YAML 1.2 (3rd Edition) Specification, Available at `http://www.yaml.org`

[11] K. Floettmann, ASTRA - A space charge tracking algorithm, `http://www.desy.de/~mpyflo/`

ISBN 978-3-95450-182-3