ADAPTIVE SPACE-CHARGE MESHING IN THE GENERAL PARTICLE TRACER CODE

S.B. van der Geer, Pulsar Physics, Eindhoven, The Netherlands *
M.J. de Loos, O.J. Luiten, Eindhoven University of Technology, The Netherlands
G. Pöplau, U. van Rienen, Rostock University, Rostock, Germany †

Abstract

Efficient and accurate space-charge calculations are essential for the design of high-brightness charged particle sources. Space-charge calculations in the General Particle Tracer (GPT) code make use of an efficient multigrid Poisson solver developed for non-equidistant meshes at Rostock University. GPT uses aggressive mesh-adaptation with highly non-equidistant spacing to speed up calculation time, where the mesh line positions are based upon the projected charge density. Here we present a new meshing scheme where the solution of an intermediate step in the multigrid algorithm is used to define optimal mesh line positions. An analytical test case and comparison with a molecular dynamics calculation of an ultrafast electron diffraction experiment demonstrate the capabilities of this new algorithm in the GPT code.

INTRODUCTION

The design of charged particle accelerators and beamlines heavily relies on numerical simulations. When non-linear space-charge effects, path-length differences and non-linear optics are significant, the favorite design method is solving the relativistic equations of motion of a large number of sample (macro) particles through the electromagnetic fields of the set-up. A major complication is that for high-brightness beams also mutual Coulomb interactions, known as space charge forces, need to be taken into account.

Several methods exist to solve the space-charge forces, ranging from brute force $O(N^2)$ to highly sophisticated meshing techniques. The design of a suitable space-charge model is further complicated by the fact that different applications require different levels of accuracy. In most cases stochastic effects are negligible, and in that case Particle-In-Cell (PIC), where Poisson's equation is solved in the co-moving frame, is the method of choice. Typically the basis of PIC is an FFT, but this has the disadvantage that it requires an equidistant mesh and this is wasteful in terms of CPU and memory requirements. For this reason, the PIC module of GPT uses an non-equidistant mesh in combination with the MOEVE Poisson solver [1] that is designed to handle such meshes.

* Corresponding author: info@pulsar.nl
† Supported by BMBF under contract number 05K10HRC

PARTICLE-MESH MODEL IN GPT

In the tracking code GPT several space charge models are implemented [4]. The 3D model we consider here is based on the particle-mesh method (see [3] and citations therein). Hereby the bunch is modeled as a certain distribution of macro particles. All fields are calculated in the electrostatic approximation in the rest frame of the bunch, implicitly assuming only a few percent energy spread.

After the transformation into the rest frame a mesh is constructed and the charge of the particles is assigned to the mesh points. Now, the potential φ can be obtained from Poisson's equation given by

$$\begin{aligned} -\Delta\varphi &= \frac{\varrho}{\varepsilon_0} && \text{in } \Omega \subset \mathbb{R}^3, \\ \varphi &= 0 && \text{on } \partial\Omega_1, \\ \frac{\partial\varphi}{\partial n} + \frac{1}{r}\varphi &= 0 && \text{on } \partial\Omega_2, \end{aligned} \tag{1}$$

where ϱ denotes the space charge distribution, ε_0 the dielectric constant and r the distance between the centre of the bunch and the boundary. Usually, the domain Ω is a rectangular box constructed around the bunch. On the surface $\partial\Omega = \partial\Omega_1 \cup \partial\Omega_2$ ($\partial\Omega_1 \cap \partial\Omega_2 = \emptyset$) perfectly conducting boundaries ($\partial\Omega_1$) or open boundaries ($\partial\Omega_2$) can be applied.

For the solution of the Poisson equation we applied a discretization with second order finite differences. This leads to a linear system of equations of the form $L_h u_h = f_h$, where u_h denotes the vector of the unknown values of the potential and f_h the vector of the given space charge density at the grid points. The step size h indicates a certain refinement level and the operator L_h is the discretization of the Laplacian.

ADAPTIVE MESHING

The adaptive GPT mesh (first in release 2.7) is an adaptive discretization that allocates the mesh lines dynamically due to the charge density in the bunch [4]. The number of mesh lines is chosen according to the number and the distribution of the particles, respectively. Efficient space charge calculations can be performed with the MOEVE Poisson solver that has been constructed especially for such non-equidistant meshes [1]. The adaptive GPT mesh is highly reliable but the construction is very complex. For example it has to be ensured that neighboring step sizes do not differ more than a certain factor in order to ensure the convergence of the multigrid Poisson solver [3].

The Self-Adaptive Multigrid Mesh

Recently, a new approach for the adaptive discretization for space charge computations has been implemented in GPT. The self-adaptive multigrid algorithm is based on an extended τ-criterion. The main idea is to start with a relatively coarse grid and to refine it step by step according to a certain criterion. The application of the τ-criterion together with the Poisson solver of MOEVE was introduced in GPT and discussed in [2]. The disadvantage of this 'plain' τ-criterion is that it does not work well for very long or very short bunches, because it can not be decided if different discretizations of the coordinate directions are necessary.

The adaptive multigrid method based on the τ-criterion was introduced in [5] with the following notations. The step sizes h and $2h$ refer to the step sizes on the fine and the next coarser grid (usually with double mesh size), respectively. The operators I_h^{2h} and $\widehat{I}_h^{2h}$ denote different restriction operators. In our implementation the injection was chosen for $\widehat{I}_h^{2h}$ and the full weighting restriction for I_h^{2h}. The τ-criterion is based on the so-called $(h,2h)$ relative truncation error τ_h^{2h} with respect to the restriction operators I_h^{2h} and $\widehat{I}_h^{2h}$. It is defined by $\tau_h^{2h} := L_{2h}\widehat{I}_h^{2h}u_h - I_h^{2h}L_h u_h$. More details can be found in [5].

The values $\tau_h^{2h}(i,j,k)$ are now available at the mesh points (i,j,k) with $i = 0,\ldots,N_x-1$, $j = 0,\ldots,N_y-1$ and $k = 0,\ldots,N_z-1$, where N_x, N_y and N_z are the numbers of mesh lines in x-, y- and z-direction, respectively. For the extended τ-criterion we introduce the differences of the values τ_h^{2h} for neighboring mesh points $D\tau_h^{2h} = (D_x\tau_h^{2h}, D_y\tau_h^{2h}, D_z\tau_h^{2h})$. Here, $D_x\tau_h^{2h} = \tau_h^{2h}(i+1,j,k) - \tau_h^{2h}(i,j,k)$ with $i = 0,\ldots,N_x-2$ denotes the differences in x-direction, D_y and D_z are defined analogously. The self-adaptive multigrid scheme with the extended τ-criterion is given as follows:

Algorithm: Self-Adaptive Multigrid

1. Start on a relatively coarse mesh.
2. Perform a few multigrid cycles on $L_h u_h = f_h$.
3. Compute τ_h^{2h} and $D\tau_h^{2h}$.
4. Add mesh lines locally in x-direction, if $|D_x\tau_h^{2h}| > \varepsilon_1$ and $a\cdot\max(|D_x\tau_h^{2h}|) > \max(|D_y\tau_h^{2h}|)$, $a\cdot\max(|D_x\tau_h^{2h}|) > \max(|D_z\tau_h^{2h}|)$ with $a>1$. Analogously, add mesh lines locally in y- and z-direction
5. Proceed from 2. as long as $|D\tau_h^{2h}| > \varepsilon_2$.

Main advantages of this approach are that the generated hierarchy of meshes now matches the hierarchy of meshes of multigrid and the values τ_h^{2h} are provided directly by the multigrid algorithm.

RESULTS

Analytical Test Case

One of the most stringent tests in our suite of analytical benchmarks is the field-error of a hard-edged cylinder with an aspect ratio varying over 6 orders of magnitude. Typical rms errors are in the 10% range, as the field is almost singular near the end of bunches with extreme aspect ratios. Nevertheless we want the code to survive these cases, with controlled behavior regardless of the number of mesh lines and particles. The main result of Figure 1 is that the new code is on average about a factor of two faster. However, from an independent test, not shown in this paper, we see that long bunches with aspect ratios $< 10^{-2}$ start to suffer in terms of accuracy. This is currently under investigation.

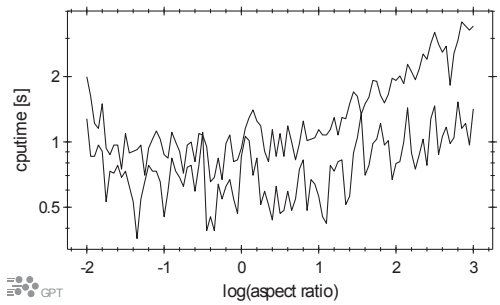

Figure 1: CPU time for the current (top line) and new (bottom line) meshing scheme.

Ultrafast Electron Diffraction

Ultrafast Electron Diffraction (UED) is an emerging technology where a sub-ps electron pulse is sent through a sample and the diffraction pattern is recorded. The technique is very demanding in terms of beam quality, as it defines the so-called coherence length. Furthermore it is very challenging to get short pulses with sufficient charge due to Coulomb repulsion. A promising method to produce femtosecond pulses with up to about a million electrons is rf bunch compression [6]. Here the longitudinal dynamics in the beamline is entirely space-charge driven, and accurate simulation tools are crucial for the design of the device.

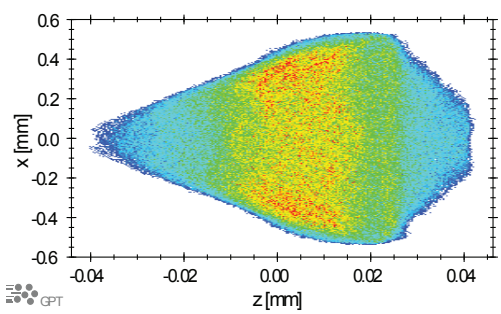

Figure 2: Real space distribution just before the longitudinal focus of the UED setup.

The new meshing algorithm is not yet sufficiently stable to track through the entire device. We therefore rely on the 'standard' meshing of GPT to produce snap shots of the particle distribution, and these snapshots are used to test the new meshing algorithm. One of the most critical parts is shown in Figure 2 obtained just before the sample where space-charge forces are significant because the bunch is being compressed to a few femtoseconds.

Figure 3 represents the corresponding space charge calculation results. A comparison is made between a full molecular dynamics result of GPT version 3.0, the current PIC meshing based on charge density and the new hierarchical meshing routine. The top plot can be considered correct, as it is based on all interactions between all particles. The resulting granularity noise is a striking feature of this approach, but typically these fluctuations can be neglected and then PIC codes are just much faster. Both meshing schemes come close to the correct result, but we must note that the built-in routine is still slightly better in terms of accuracy. The reason for that can be understood from the longitudinal meshing sequence of the new algorithm for stage 0 to 4, as shown in Figure 4. In this draconian test-case there are small fluctuations at the finest level, giving rise to a non-smooth longitudinal electric field.

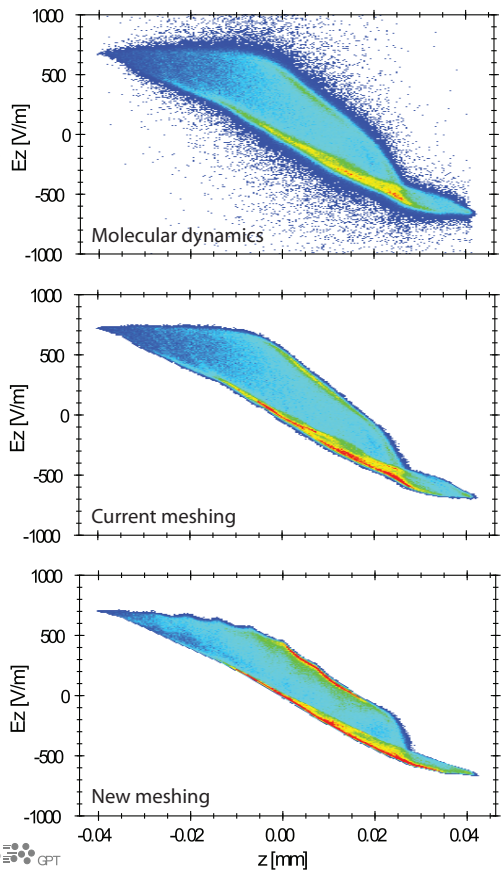

Figure 3: GPT simulations with various space-charge models corresponding to Figure 2.

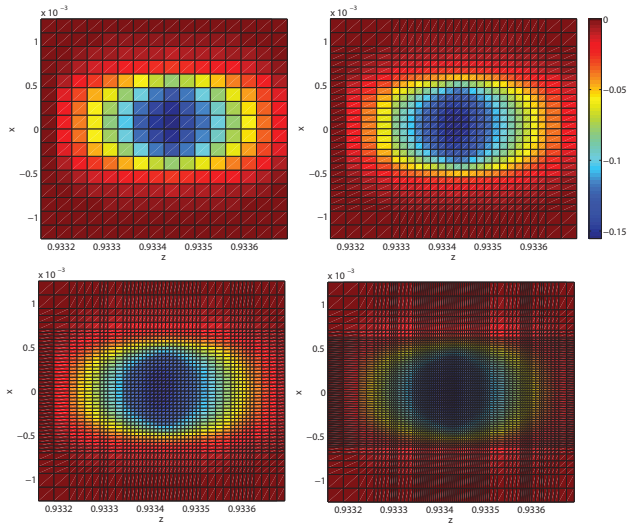

Figure 4: Meshes with increasing density obtained with the new algorithm for the bunch shown in Figure 2.

CONCLUSIONS AND OUTLOOK

A new meshing scheme has been implemented and tested in the GPT code. In theory this new scheme should be faster, easier to implement, a better match for the multigrid solver, and more robust. The first results have been promising, showing an improvement in CPU time of about a factor of 2. However, the new scheme is not yet as robust as the 'standard' meshing of GPT and we also observe reduced accuracy for very long bunches. Nevertheless, given the performance increase and elegance of the scheme, we are confident that eventually the new approach will outperform the current version.

REFERENCES

[1] G. Pöplau. *MOEVE 2.0: Multigrid Poisson Solver for Non-Equidistant Tensor Product Meshes*. Universität Rostock, 2006.

[2] G. Pöplau and U. van Rienen. Efficient 3D space charge calculations with adaptive discretization based on multigrid. In *Proceedings of IPAC 2010 (1st International Particle Accelerator Conference), Kyoto, Japan*, pages 1832–1834, 2010.

[3] G. Pöplau, U. van Rienen, S.B. van der Geer, and M.J. de Loos. *Multigrid algorithms for the fast calculation of space-charge effects in accelerator design.* IEEE Transactions on Magnetics, 40(2):714–717, 2004.

[4] *General Particle Tracer (GPT) code*, Pulsar Physics, The Netherlands, `www.pulsar.nl/gpt`.

[5] U. Trottenberg, C. Oosterlee, and A. Schüller. *Multigrid*. Academic Press, San Diego, 2001.

[6] T. van Oudheusden, P.L.E.M. Pasmans, S.B. van der Geer, M.J. de Loos, M.J van der Wiel, O.J. Luiten, *Compression of subrelativistic space-charge-dominated electron bunches for single-shot femtosecond electron diffractionon*. PRL, 105 (26), art. no. 264801 (2010).