TESLA Report 2003-03

A multigrid based 3D space-charge routine in the tracking code GPT

Gisela Pöplau†‖ ¶, Ursula van Rienen†, Marieke de Loos‡ and Bas van der Geer §

† Institute of General Electrical Engineering, Rostock University, Germany
‡ Eindhoven University of Technology, The Netherlands
§ Pulsar Physics, Soest, The Netherlands

Abstract. Fast calculation of 3D non-linear space-charge fields is essential for the simulation of high-brightness charged particle beams. We report on our development of a new 3D space-charge routine in the General Particle Tracer (GPT) code. The model is based on a non-equidistant multigrid Poisson solver that is used to solve the electrostatic fields in the rest frame of the bunch. Since the multigrid Poisson solver depends only linearly on the number of mesh points for the discretized electrostatic problem the space-charge routine scales linearly with the number of particles in terms of CPU time. This performance allows over a million particles to be tracked on a normal PC. The choice of the routine parameters for an optimal performance will be discussed with the model of a spherical bunch.

PACS numbers: 02.60.Cb, 02.70.Bf, 03.50.De

1. Introduction

Numerical prediction of charged particle dynamics in accelerators is essential for the design and understanding of these machines. Applications such as colliders and SASE–FEL's demand very high quality electron bunches, where any anomaly severely degrades the final performance.

A powerful tool widely used for the study of the behaviour of charged beams is the tracking code GPT (General Particle Tracer) [2]. It calculates the trajectories of a large number of sample–particles through the combined external and self–induced fields generated by the charged particles (the so–called space–charge forces). Depending on charge density and energy, a direct point–to–point model can not be used to calculate space–charge forces because of granularity problems and the inherent $\mathcal{O}(N^2)$ scaling between the number of sample particles and CPU time [11].

In this paper we introduce a 3D model for the fast calculation of space–charge following the ideas in [10]. The space–charge fields are computed in the rest frame by a non–equidistant multigrid scheme. Hence, the numerical effort scales linearly with the number of particles in

‖ supported by a research grant from DESY, Hamburg
¶ correspondence to (gisela.poeplau@etechnik.uni-rostock.de)

terms of CPU time. The new model is well suited for a variety of applications, including the calculation of space–charge fields in a high–brightness photo–injector which are presented in [12].

In this paper we discuss the influence of various parameters on the efficiency of the 3D space–charge routine using the model of a spherical bunch with uniform or Gaussian particle distribution. Related numerical results are given in section 3. For further simulation tests regarding the calculation of important beam parameters such as emittance we refer to [8, 12].

2. The 3D space charge model

The particle tracking is performed by solving the relativistic equations of motion for a set of macro particles (sample particles) representing the distribution of the particles of a bunch. In the GPT code a 5th order embedded Runge–Kutta scheme with adaptive step size control is implemented for the numerical integration of these equations [11]. In each time step of the numerical integration the space–charge fields have to be taken into account. The space–charge calculation with the 3D model is performed as follows:

(i) Laboratory frame $\rightarrow$ rest frame: Transformation of the bunch from the laboratory frame to a rest frame with an average velocity by Lorentz transformation.

(ii) Rest frame: Determination of a non–equidistant 3D Cartesian grid in correspondence to the charge density of the bunch (see subsection 2.1).

(iii) Rest frame: Approximation of the charge density at the grid points.

(iv) Rest frame: Calculation of the electrostatic potential at the grid points via Poisson's equation applying a multigrid algorithm. The finite difference scheme (7–point stencil) is used for the discretization of Poisson's equation (see subsection 2.2).

(v) Rest frame: Derivation of the electric field and trilinear interpolation of the field values to the particle positions.

(vi) Rest frame $\rightarrow$ laboratory frame: Transformation of the field to the laboratory frame by Lorentz transformation.

The efficiency and accuracy of the space–charge calculation mainly depends on the determination of the 3D mesh and the applied multigrid scheme to solve Poisson's equation. Both we describe in the next two subsections.

2.1. The generation of the mesh

The electromagnetic potential is calculated on a 3D Cartesian mesh with an approximation of the charge density at the grid points in the rest frame. The 3D mesh is generated in a box around the bunch. To reduce the number of mesh lines needed, and thus to reduce CPU time, the density of the mesh lines is increased only if the charge–density increases. The actual positioning of the mesh lines is an iterative process. The mesh lines are distributed such that they are spaced according to the distribution of the beam charge density.

Strictly following this rule the resulting mesh spacing can lead to conflicts with the Poisson solver (see subsection 2.2). That is in such cases the multigrid Poisson solver converges only slowly or does not converge at all. Thus the parameter fn is introduced to maintain a maximum difference in spacing between neighboring mesh lines, to avoid the creation of a non–optimal mesh line distribution for the Poisson solver. If, e. g. $fn = 0.25$, then the difference in spacing between neighboring mesh lines can not vary by more than 25%. The effect of fn is shown in Fig. 1. When $fn = 0$, the spacing between all neighboring mesh lines is allowed to vary by 0%, creating an equidistant mesh. Such a mesh is most stable for the multigrid Poisson solver, but it will create many empty mesh boxes. On the other side, setting $fn = 0.2$ results in a dense sampling of the electron bunch and sparse sampling of the surrounding area.

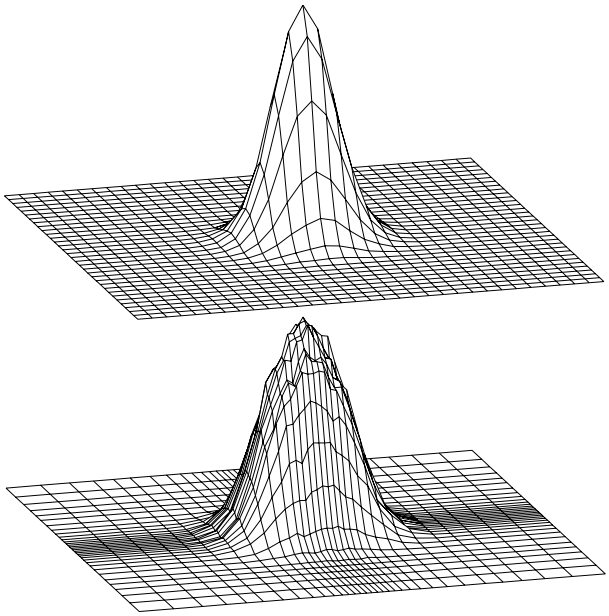

Figure 1. Mesh line positions ((x, y)–plane) for a Gaussian charge density with $fn = 0$ (top) and $fn = 0.2$ (bottom). The vertical axis shows the total charge in each mesh box, where the height of the top has been normalized in both plots.

2.2. The multigrid Poisson solver

The space–charge forces will be calculated in the rest frame of the bunch by means of Poisson's equation given by

$$-\Delta\varphi \;=\; \frac{\varrho}{\varepsilon_0} \qquad \text{in } \Omega \subset \mathbb{R}^3.$$

Here, φ denotes the potential, ϱ the charge density and ε_0 the dielectric constant. The domain Ω is a box around the particle bunch. On the boundary we consider three cases: a beam pipe, a cathode and free space. The beam pipe has ideal conducting walls transversely, what results in Dirichlet boundary conditions, and open boundary conditions longitudinally. The cathode has the same assumptions as the beam pipe accept the cathode surface, here Dirichlet boundary

conditions are assumed. The calculation in free space leads to open boundary conditions and can be used when the bounding box is located far enough from the boundaries of the structure. While a bounding cube is necessary with Dirichlet boundary conditions (see [8]) the open boundary conditions allow to put a bounding box relatively close around the bunch. Thus, a lot of computing time can be saved especially for very short or very long bunches.

Poisson's equation is discretized by finite differences on the non–equidistant mesh described in the previous subsection. An approximation of the charge density $\frac{\varrho}{\varepsilon_0}$ is computed on the mesh points from the charge of the macro particles.

The solution of the resulting system of equations requires a fast and robust solver. State-of-the-art is the application of a multigrid method as Poisson solver. In model cases the numerical effort scales with the number of mesh points. Here, we give only the general idea of a geometrical multigrid algorithm. Details can be found in [3, 1]. The multigrid algorithm operates on a certain number of grids starting with the mesh given by the discretization of Poisson's equation. This mesh is referred to as the fine grid or the fine level. Then a sequence of coarser grids is generated by removing mesh lines. On an equidistant mesh every second mesh line is removed. Now iteratively, a raw approximation of the solution of the systems of equations is obtained by the application of a few steps of a relaxation scheme (e. g. Gauss–Seidel iteration) which is called pre–smoothing. This approximation is then improved by a correction vector obtained on the coarser grids (the so–called coarse grid correction) where restriction and interpolation work as grid transfer operators. After applying interpolation another few steps of relaxation are necessary (post–smoothing). For the space charge calculations a multigrid V–cycle is realized. This scheme goes strictly down from the fine to the coarsest grid and then up again to the fine level.

As shown in [6, 7] the coarsening strategy is crucial for the convergence of the multigrid algorithm on non–equidistant grids. The generation of coarse grids with every second grid line removed as suggested in [1] is not reasonable with the dicretizations for bunches. It would lead to coarser grids with increasing aspect ratio of the mesh spacing. Hence the convergence of a multigrid scheme on such grids would considerably slow down. Here, the removal of mesh lines follows the rule: Two neighboring steps h_1 and h_2 remain also in the next coarser grid as long as either $h_1 \geq sh_{min}$ or $h_2 \geq sh_{min}$, where h_{min} denotes the overall minimal step size of the corresponding fine level. The factor s is chosen as $s = 1.6$ or $s = 1.7$ with the objective to obtain a decreasing aspect ratio of the mesh spacing.

Furthermore, the choice of the multigrid parameters such as the number of pre- and post–smoothing steps, the application of full or half restriction considerably influence the performance of the multigrid scheme. If the convergence of the multigrid algorithm turns out to be not sufficient (e. g. if the coarsening does not come out with decreasing aspect ratio on all levels), multigrid can be applied as a preconditioner for the conjugate gradient algorithm. This method leads to a better convergence at least in cases where a plain multigrid scheme converge too slow [5, 4].

3. Numerical test cases

The new 3D space–charge routine has been tested with a sphere filled with electrons with both uniform and Gaussian distribution. The space–charge forces has been computed for only one time step. For numerical tests with other bunch shapes and tracking examples we refer to [8, 12]. For an optimal performance of the space–charge routine several questions have to be considered:

(i) What is an optimal distribution of mesh lines both for the approximation of the particle distribution of the bunch and for the multigrid performance? (Choice of parameter fn)

(ii) Which number of mesh lines is optimal? (As few as necessary.)

(iii) How does the choice of multigrid parameters influence the performance of the algorithm?

Three possible versions of the multigrid scheme has been investigated. Two of these schemes have been performed with the following components: the Gauss–Seidel red–black iteration has been taken as smoothing operator with 2 pre– and 2 post–smoothing steps MG(2,2), full restriction has been tested versus half restriction. The third algorithm has been the application of multigrid as preconditioner for the conjugate gradient method (MG-PCG). It requires that the multigrid scheme is a symmetric and positive operator [4, 5]. Thus the components have been chosen as follows: two pre–smoothing steps with red–black Gauss–Seidel relaxation, two post–smoothing steps with black–red Gauss-Seidel relaxation and full restriction. Two V–cycles have been performed per CG–iteration step (MG–PCG(2,2)(2)).

3.1. Sphere with uniform particle distribution

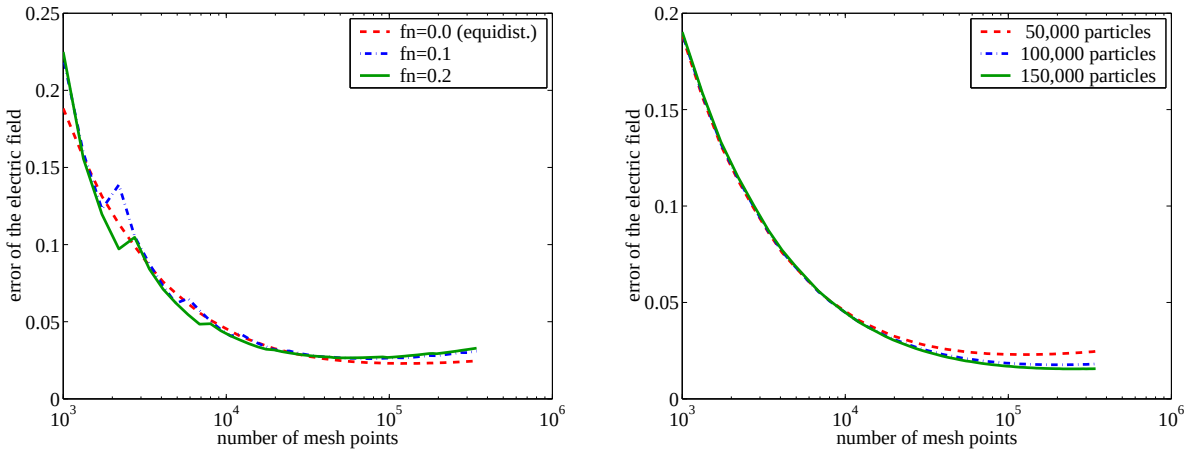

Figure 2. Error of the electric field for uniformly distributed particles in a sphere: Comparison of different meshes depending on parameter fn performed with 50,000 particles (left), comparison of different numbers of particles on an equidistant mesh with $fn = 0.0$.

For the investigation of the first two questions we tested the model of uniformly distributed particles in a sphere with the known analytical electrical field of a uniformly charged sphere. The error for the electrical field has been measured after interpolation at the

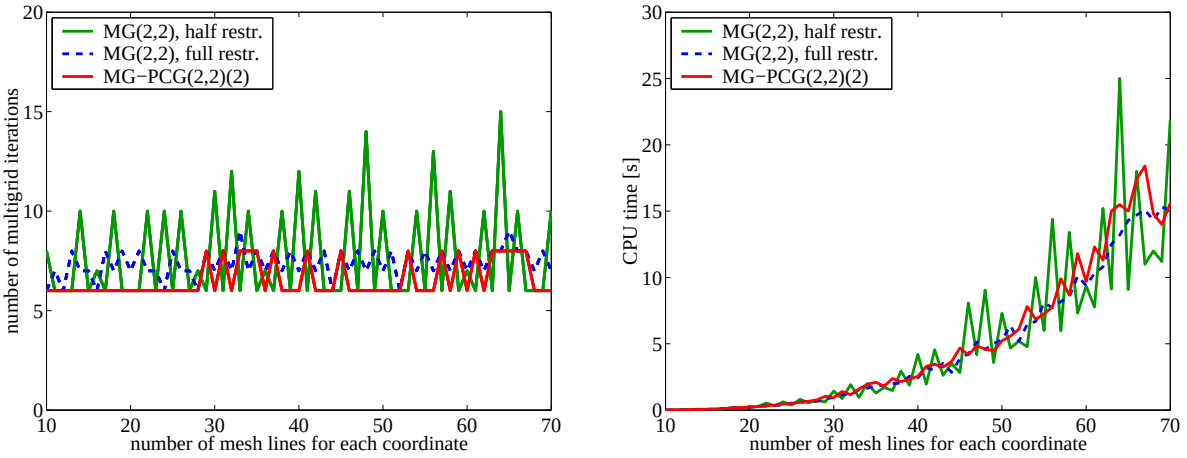

Figure 3. Influence of the choice of multigrid parameters to the performance of the space–charge routine (MG: multigrid, MG–PCG: multigrid preconditioned conjugate gradient): the number of multigrid iterations until the residual is less than 10^{-6} in the maximum norm (left), the related CPU times measured on a 800 MHz Pentium PC (right). The calculations have been performed on an equidistant grid with $fn = 0.0$ with a spherical bunch of 50,000 particles, where the particles have a uniform distribution.

position of the particles. Fig. 2 shows the best convergence for the equidistant grid ($fn = 0.0$) what is not surprising for a uniform particle distribution. Since the multigrid scheme has the best convergence and stability on equidistant meshes, these meshes should be preferred in the case of uniformly distributed particles. Furthermore, it can be concluded from Fig. 2 that neither a large number of mesh points nor a large number of particles improve the final result. Consequently, for a high efficiency of the 3D space–charge routine no more mesh lines or particles than reasonable should be taken for simulations.

Fig. 3 shows the performance of the above introduced 3 multigrid algorithms. As expected the multigrid preconditioned conjugate gradient method has the most stable performance. The multigrid scheme with half restriction as grid transfer operator turns out to be very sensitive to non–optimal meshes. In this case these are meshes with an even number of mesh lines for each coordinate direction. Here, the coarsening can not generate good results concerning the mesh spacing (see subsection 2.2).

3.2. Sphere with Gaussian particle distribution

Tests with the model of a Gaussian particle distribution in a sphere show the advantage of the application of non–equidistant meshes. Fig. 4 shows that meshes with larger aspect ratios in mesh spacing, i. e. larger fn result in a better convergence of the error of the electric field. It can be concluded that meshes with $fn > 0.2$ would not improve the results for the field error. Similar as for the uniformly distributed particles an increasing number of particles (greater than 100,000) in the sphere leads to no smaller field errors.

The multigrid schemes have been performed on a non-equidistant mesh with $fn = 0.2$ (see Fig. 5). Again the multigrid preconditioned conjugate gradient method turns out to be

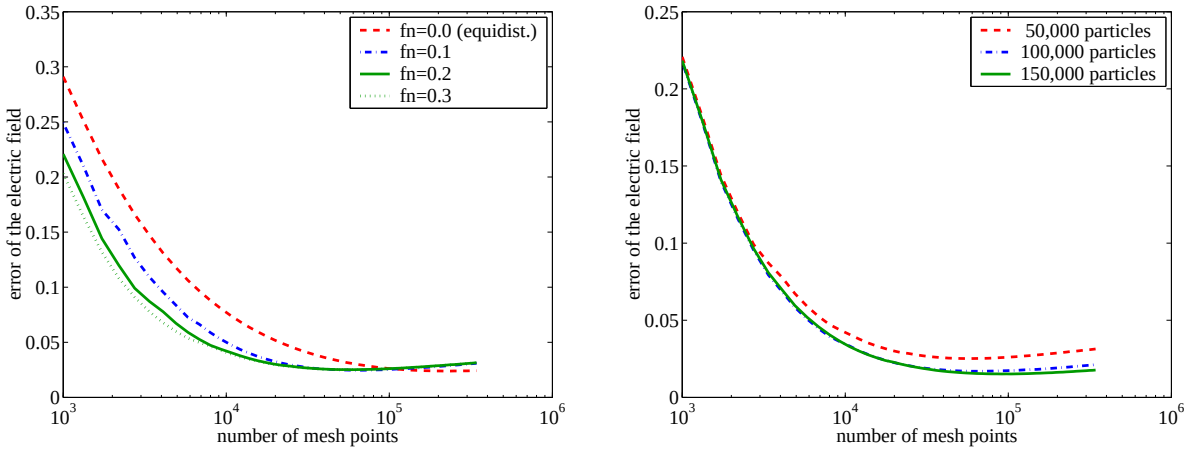

Figure 4. Error of the electric field for Gaussian distributed particles in a sphere: Comparison of different meshes depending on parameter fn performed with 50,000 particles (left), comparison of different number of particles on a non–equidistant mesh with $fn = 0.2$.

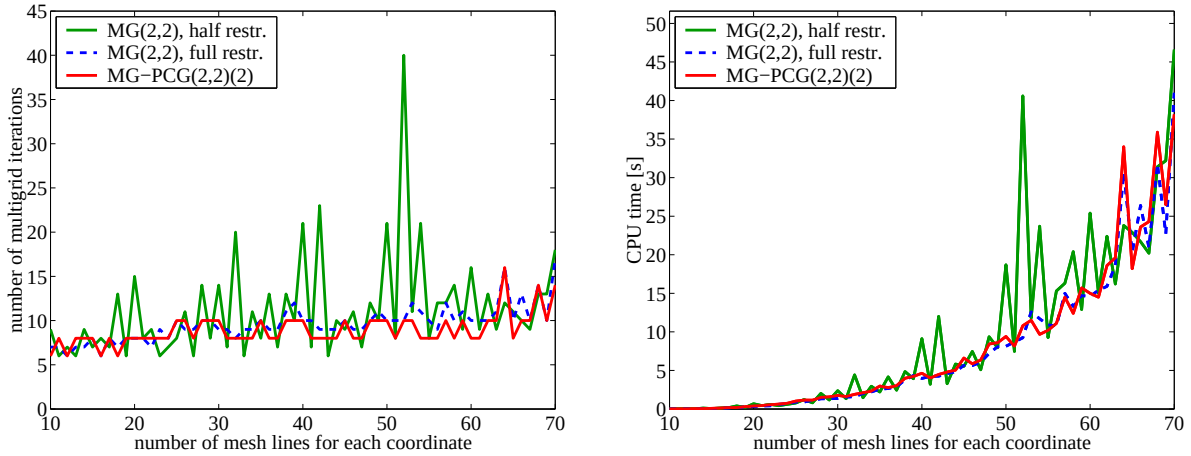

Figure 5. Influence of the choice of multigrid parameters to the performance of the space–charge routine: the number of multigrid iterations until the residual is less than 10^{-6} in the maximum norm (left), the related CPU times measured on a 800 MHz Pentium PC (right). The calculations have been performed with a spherical bunch of 50,000 particles on a non–equidistant grid with $fn = 0.2$, where the particles have a Gaussian distribution.

the most stable algorithm. Multigrid performed with half restriction is as in the previous test case very sensitiv to the distribution of mesh lines.

4. Conclusion

A new 3D space-charge routine implemented in the GPT code has been described in this paper. The new method allowing 3D simulations with a large number of particles on a common PC is based on a multigrid Poisson solver for the calculation of the electrostatic potential in the rest frame.

Various parameters of the routine have been tested for model bunches in order to find out

a stable and fast performance of the space-charge routine. Various applications like colliders or SASE-FEL's require very high quality bunches. Related to such applications we refer to [12], where the space-charge effects of very short and very long bunches with the new 3D space-charge routine in GPT are studied.

References

[1] Briggs W L, van Emden H and McCormick S 2000 *A Multigrid Tutorial. 2nd edition* (SIAM, Philadelphia)

[2] General Particle Tracer (GPT), release 2.60, Pulsar Physics, De Bongerd 23, Soest, The Netherlands

[3] Hackbusch W 1985 *Multi-Grid Methods and Applications* (Springer, Berlin)

[4] Jung M and Langer U 1991, Applications of multilevel methods to practical problems *Surv. Math. Ind.* **1** 217–257

[5] Jung M, Langer U, Meyer A, Queck W and Schneider M 1989, Multigrid preconditioners and their applications. *Proc. of the 3rd GDR Multigrid Seminar (Biesenthal)* 11–52

[6] Pöplau G, van Rienen U, Staats J and Weiland T 2000, Fast Algorithms for the Tracking of Electron Beams *Proc. of EPAC 2000 (Vienna)* 1387–1389

[7] Pöplau G, van Rienen U 2002, Multigrid Solvers for Poisson's Equation in Computational Electromagnetics *Proc. of the 3rd Conf. on Scientific Computating in Electrical Engineering (SCEE–2000)* U van Rienen, D Hecht and M Günther, eds, LNSCE **18**, (Springer, Berlin) 169–176

[8] Pöplau G, van Rienen U, de Loos M J and van der Geer, S B 2002, A Fast 3D Multigrid Based Space-Charge Routine in the GPT Code *Proceedings of EPAC 2002 (Paris)* 1658–1668

[9] ——2002 Fast Calculation of Space Charge in Beam Line Tracking by Multigrid Techniques *submitted to the Proc. of the 4th Conf. on Scientific Computating in Electrical Engineering, SCEE–2002 (Eindhoven)*

[10] Staats J, Weiland T, Kostial S and Richter A 1999, Tracking of Electron Beams with Numerically Determined Space Charge Forces *Proc. of the 1999 Particle Accelerator Conf. PAC'99 (New York)* 2740–2742

[11] Van der Geer S B and de Loos M J 2001 *The General Particle Tracer Code. Design, implementation and application* PhD thesis

[12] Van der Geer S B, de Loos M J, Luiten O J, Pöplau G and van Rienen U 2002, 3D Space-charge model for GPT simulations of high-brightness electron bunches *submitted to this volume*,
also: TESLA Report 2003-04, DESY, Hamburg, 2003.