LBNL-51797

PARSEC: PARALLEL SELF-CONSISTENT 3D ELECTRON-CLOUD SIMULATION IN ARBITRARY EXTERNAL FIELDS[*]

Andreas Adelmann[†] and Miguel A. Furman, LBNL, Berkeley CA 94720, USA

Abstract

We present PARSEC, a 3D parallel self-consistent particle tracking program which allows electron-cloud calculations in arbitrary external fields. The program is based on an general particle tracking framework called GenTrackE [5]. The Lorentz force equation is integrated with time as the independent variable. A 3D parallel Multigrid solver computes the electric field for the drive beam in the beam frame, while the space-charge field of the electrons is computed in the lab frame. The resulting total field, obtained by superposition, acts on both the beam particles and the cloud electrons. Primary and secondary emission takes place at each time step of the calculation. This sort of computation is only possible by the use of massive parallelization of the particle dynamics and the Poisson solver in combination with modern numerical algorithms such as the Multigrid solver with Gauss-Seidel smoothing.

INTRODUCTION AND MOTIVATION

The electron-cloud effect (ECE) has been investigated in various storage rings for several years now [1]. The ECE arises from the strong coupling of a two-species plasma with the surrounding vacuum chamber. Several analytical models and simulation programs and have been developed to study this effect [2]. Owing to the complexity of the problem, these simulation codes typically make one or more simplifying assumptions, such as: (i) the electrons are dynamical but the beam is a prescribed function of space and time; (ii) the beam is dynamical but the electron cloud is a prescribed function of space and time; (iii) both the beam and the electrons are dynamical, but the electron-wall interaction, particularly the secondary emission process, is either absent or much simplified; (iv) the geometry of the beam and/or vacuum chamber is much simplified (eg. round beams and/or cylindrical chambers); (v) the simulation “looks” at only one specific region of the machine, typically a field-free region or one magnet of a specific kind; (vi) the forces on the particles, both from, and on, the electrons and the beam, are purely transverse. Computer codes involving these approximations, when applied in the proper context, have shed valuable information on one or more aspects of the ECE.

There are problems, however, in which any of these approximations may render the reliability of code inadequate for a quantitative understanding of the dynamics. One such example concerns problems involving very long, intense, bunches with significant variation in the longitudinal profile, which require a self-consistent, fully 3D simulation, including a full description of the storage ring lattice (or at least, a section of the lattice at least as long as the bunch). Another example might be the simulation of damping rings for future linear colliders, which make significant use of wigglers. In this article we report on progress towards the goal of a fully self-consistent and realistic simulation of the ECE which, in its final stage, will not invoke any of the above-mentioned simplifications.

[*] Work supported by the US DOE under contract DE-AC03-76SF00098.

[†] AAAdelmann@lbl.gov

THE OVERALL SIMULATION MODEL

Self-consistent formulation

Let the particle coordinates of particle k be $\vec{x}_k = (q_1, q_2, q_3)_k$, and the normalized velocity be $\vec{\beta}_k = (v_x/c, v_y/c, v_z/c)_k$ where c is the speed of light (all quantities in MKS units unless explicit stated otherwise). We consider $\ell = 1, 2, \cdots$ magnetic elements which makes up what is called the lattice L. Defining $I = \{1, 2, \cdots\}$ and $J = \{1, 2, \cdots\}$ the index sets for the beam particles and electrons, respectively as unique identifiers, we are able to distinguish beam particle ($i \in I$) and electron coordinates ($j \in J$) in an natural way (see Figure 1 as an illustration). For each particle $k \in I \cup J$ we solve formally

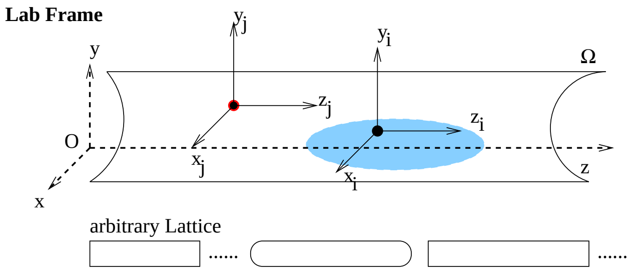

Figure 1: (color) Geometry and Particle domains.

$$\frac{d(m_k \gamma_k c \vec{\beta}_k)}{dt} = \vec{F}(\vec{x}_k, t) \quad (1)$$

$$\vec{F}(\vec{x}_k, t) = \frac{q_k}{\gamma_k m_k}(\vec{E}(\vec{x}_k, t) + \vec{\beta}_k \times \vec{B}(\vec{x}_k, t)) \quad (2)$$

where m_k and q_k is the mass and charge of the particle, respectively, and γ_k its usual relativistic factor.

The lattice magnetic field $\vec{B}_{ext}$ in cylindrical coordinates is represented by:

$$\begin{aligned}B_\rho &= \sum c_{m,n} I'_m(nk_z\rho)\sin(m\phi)\cos(nk_z z)\\ B_\phi &= \sum c_{m,n}\frac{m}{nk_z\rho} I_m(nk_z\rho)\cos(m\phi)\cos(nk_z z)\\ B_z &= -\sum c_{m,n}\frac{m}{nk_z\rho} I_m(nk_z\rho)\sin(m\phi)\cos(nk_z z).\end{aligned} \tag{3}$$

where I_n is the usual modified Bessel function, which satisfies $I'_m(nk_z\rho) = \frac{1}{2}[I_{m-1}(nk_z\rho) + I_{m+1}(nk_z\rho)]$. In this formulation we will treat coasting beams only although it is straightforward to include acceleration. The potential ϕ is obtained by solving two Poisson problems with $\rho_e(\vec{x}_j)$ and $\rho_b(\vec{x}_i)$ the electron and beam charge density. Let $\vec{x}'_i = \mathcal{L}(\vec{x}_i)$ with $\mathcal{L}$ denoting the proper Lorentz transformation from the laboratory to the beam rest frame. The first Poisson problem, in which the beam charge density ρ_b is the source, reads:

$$\begin{aligned}\triangle\phi(\vec{x}'_i) &= -\frac{\rho_b(\vec{x}'_i)}{\epsilon_0},\ \vec{x}'_i \in \Omega \subset \mathcal{R}^3\\ \phi(\vec{x}'_i) &= 0,\ \vec{x}'_i \in \partial\Omega.\end{aligned} \tag{4}$$

Upon Lorentz-transforming back to the Lab frame, this yields both an electric ($\vec{E}_b$) and magnetic field ($\vec{B}_b$). Assuming that the electrons are sufficiently non-relativistic, which is typically a good approximation, we can neglect their contribution to the magnetic field, and we can solve in the laboratory frame for the second Poisson problem, in which the electron-cloud density ρ_e is the source,

$$\begin{aligned}\triangle\phi(\vec{x}_i) &= -\frac{\rho_e(\vec{x}_i)}{\epsilon_0},\ \vec{x}_i \in \Omega \subset \mathcal{R}^3\\ \phi(\vec{x}_i) &= 0,\ \vec{x}_i \in \partial\Omega\end{aligned} \tag{5}$$

thus the full answer is obtained by superposing the two fields:

$$\begin{aligned}\vec{F}(\vec{x}_k,t) = \frac{q_k}{\gamma m_k}(&\vec{E}_e(\vec{x}_j,t) + \vec{E}_b(\vec{x}_i,t) +\\ &\vec{\beta}_k \times (\vec{B}_{ext}(\vec{x}_k,t) + \vec{B}_b(\vec{x}_j,t)))\end{aligned} \tag{6}$$

where $\vec{E}_e$ is the electric self-field of the electrons.

Secondary Emission Model

When an electron strikes the vacuum chamber wall, it can be absorbed or can generate one or more secondary electrons. In our computations we simulate this process by a detailed probabilistic described elsewhere [3]. This process incorporates, as inputs, the measured secondary electron yield (SEY) δ and the energy spectrum of the emitted electrons, $d\delta/dE$ for a given vacuum chamber surface material. The three main subprocesses, namely elastical reflection, rediffusion, and true secondary emission, are included. We are not concerned for the moment with the processes responsible for the generation of primary electrons (chiefly the photoelectric effect, ionization of residual gas, and stray beam particles striking the vacuum chamber wall), since these process is simpler to simulate with phenomenological models.

Time integration

The code integrates (1) using a 4th-order Runge-Kutta method, with adaptive time step control for the electrons. We estimate the time step T_j by considering the cyclotron frequency $\omega_c = eB/m_e$, consequently

$$T_j = \frac{2\pi}{\omega_c K}. \tag{7}$$

is defined upon the factor K. Defining ϵ_{min} and ϵ_{max} the minimum and maximum error tolerated we estimate K using two Runge-Kutta steps and Richardson Extrapolation [6]. Choosing an initial step size T_j by setting $K = 1$ and let u_1 be the result of an Runge-Kutta step of length T_j. Let u_2 is the result of two subsequent Runge-Kutta steps of length $T_j/2$. We then estimate the error by $\epsilon = |u_1 - u_2|$. Richardson Extrapolation is then be used to set K corresponding to an error-range as defined. This procedure will guarantee the minimal work in order to achieve a desired accuracy, considering the dynamic of the individual particle.

```
#1  $\forall i \in I$
#2    for $m = 1$ to $N$
#3      Integrate equation of motion
#4    end for
#5  calculate $\rho_z$ drive beam
#6  generate electons
#7  $\forall j \in J$
#8    while $timeLeft_j > 0.0$ AND NOT ATTHEWALL$_j$
#9      Integrate equation of motion
#10     if($\vec{x}_j \notin \Omega$)
#11       status of particle $j$ equal to ATTHEWALL
#12     end if
#13   end while
#14  $\forall j \in J$
#15   if status of particle $j$ equal to ATTHEWALL
#16     generate secondaries
#17   end if
#18  if secondary generated goto #7
#21  calculate space charge
```

Figure 2: Tracking Algorithm

Particle Tracking Procedure

As a first step towards a full lattice simulation, we model a portion of the magnetic lattice by imposing periodic boundary conditions in the z coordinate, for example one half of the PSR circumference or one FODO cell of the LHC arc. Further, we assume a constant number of particles in the drive beam, and a fixed number N of time steps of size ΔT. All electrons $i \in I$ and protons $j \in J$ are advanced by $N\Delta T$ followed by a space-charge calculation. The simplified algorithm for advancing all particles by $N\Delta T$ in time is pictured in Figure 2, where $timeLeft_j = N\Delta T - t_j$ and ATTHEWALL indicate that a particles hits the vacuum chamber (we left out some of the indexes where the meaning is deducible form the contents). ρ_z is the longitudinal charge density of the drive beam which determines the distribution of the generated primary electrons.

Poisson Solver

For the Poisson solver, this type of simulations is very demanding. First of all, the computational domain Ω is very large *and* almost completely filled with simulation particles. Second, the number of macro particles (or simulation particles) is huge (many times 10^6) and the number of time steps is large as well. The Poisson solver uses a semi-unstructured grid as shown in Figure 3 to decompose Ω. Linear bases function are used to assemble the stiffness matrix $\mathcal{A}$ and the right hand side f_h (discrete charge density) is constructed using a area wighting scheme. The resulting linear system of equations

$$-\Delta\phi = \frac{\rho}{\epsilon_0}, \phi = 0 \text{ on } \partial\Omega \Longrightarrow \mathcal{A}u_h = f_h \qquad (8)$$

is solved using parallel Multigrid. From the solution u_h we back-interpolate and use a second-order finite-difference scheme in order to obtain the two electric fields used in equation (6). Preliminary performance of the parallel Pois-

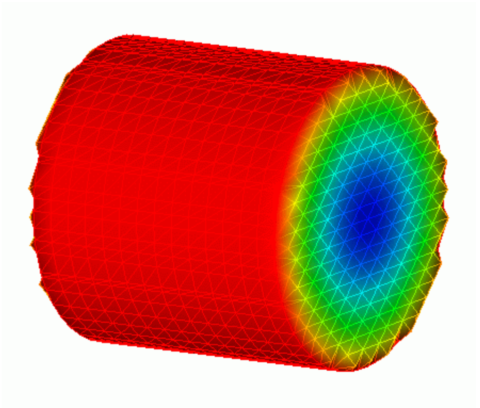
Figure 3: (color) Finite Element Discretization of Ω

son Solver [4] and the parallel grid generators is shown in Table 1 for an toy Poisson problem in $\mathcal{S}^3$ (sphere). We show in Table 1 the scalability of the grid generator and the solver. A method is said to be scalable, if the time (T) times the number of processors used (P) divided by problem size (M) remains bounded as P and M gets increased. The data in Table 1 is given for the grid generation (in column 3) and for one multigrid iteration (in column 5) with an Gauss-Seidel smoother. Table 1 shows excellent scalability with respect to the problem size M which is equivalent to say we can handle in the order of 10^{11} **macro particles** in a simulation with reasonable computing time. For this scaling study we use the Seaborg (IBM SP-3) computer at NERSC.

P	M	T_gP/M	T	TP/M
8	625,464	3.5e-3	3.1	3.9e-5
32	306,080	8.5e-3	0.78	8.1e-5
248	4,751,744	5.90e-3	1.2	6.2e-5
248	36,998,619	7.50e-3	7.7	5.1e-5
960	23,312,735	4.85e-3	4	1.64e-4
2025	405,242,845	6.60e-3	10.7	5.3e-5
4075	7,166,171,845	8.76e-3	160	9.9e-5

Table 1: Scalability of the parallel grid generator T_gP/M and the Poisson solver showing also T, the time in seconds for one Multigrid step

CONCLUSION

The presented code PARSEC is based on GenTrackE, which is written in C++ and is fully parallelized using MPI. PARSEC advances macro particle of the drive beam and the electrons using a 4th-order Runge-Kutta method. Variable time steps for the electrons according to their dynamics are used. The arbitrarily shaped computational domain is discretized using linear finite elements, the resulting linear system of equation is solved efficiently by the use of an massive parallel and scalable Multigrid solver.

We are finalizing the code construction and are about to start simulation of a simplified LHC FODO cell, as well as some part of the Los Alamos proton storage ring. The issue of large aspect ratios in the computational domain and the impact of the accuracy of the Poisson solver will be investigated in detail, a subject which is of general importance in many space charge dominated problems.

ACKNOWLEDGMENTS

We thank Robert Ryne, Mauro Pivi and Andy Wolski for enlightening discussions and NERSC for supercomputer support.

REFERENCES

[1] Various contributions to these proceedings.

[2] Various contributions to the Proceedings of the ECLOUD-02 Workshop [CERN Yellow Report no. CERN-2002-01], http://slap.cern.ch/collective/ecloud02/.

[3] M. A. Furman and M. T. F. Pivi, PRST-AB **5** 124404 (2002).

[4] A. Adelmann and Ch. Pflaum, LBNL Report to be appear (2003).

[5] A. Adelmann, GenTrackE: General Tracking Engine, LBNL Report to be appear (2003).

[6] R. Bulirsch and J. Stoer, Introduction to Numerical Analysis. New York: Springer-Verlag, 1991.