Available online at www.sciencedirect.com
SCIENCE DIRECT
Computer Physics Communications 164 (2004) 306–310
Computer Physics Communications
www.elsevier.com/locate/cpc

A treecode algorithm for simulating electron dynamics in a Penning–Malmberg trap ☆

A.J. Christlieb [a,*], R. Krasny [a], J.P. Verboncoeur [b]

[a] *Department of Mathematics, University of Michigan, Ann Arbor, MI 48109, USA*
[b] *Department of Electrical and Computer Engineering, University of California, Berkeley, CA 94720, USA*

Available online 27 July 2004

Abstract

A treecode algorithm is presented for computing the electrostatic potential and electric field in a system of charged particles. The algorithm is grid-free and with N particles it reduces the operation count to $O(N \log N)$, as opposed to $O(N^2)$ which is required for direct summation of pairwise interactions. The key idea is to replace the particle–particle interactions by particle–cluster interactions which are evaluated using a Taylor approximation in Cartesian coordinates. The treecode is combined here with a boundary integral method to simulate electron dynamics in a Penning–Malmberg trap.

PACS: 52.65

Keywords: Charged particles; Electrostatic potential; Treecode algorithm

1. Introduction

Consider a system of charged particles $\vec{x}_i$, $i = 1, \ldots, N$, with charges q_i and potential energy function

$$\Phi(\vec{x}) = \frac{1}{\epsilon_0} \sum_{i=1}^{N} G(\vec{x}, \vec{x}_i) q_i, \tag{1}$$

where $G(\vec{x}, \vec{x}_i)$ is the Green's function at $\vec{x}$ due to particle $\vec{x}_i$. The electric field is $\vec{E} = -\nabla\Phi$ and in a Lagrangian simulation it must be evaluated at each particle location (omitting the infinite self-interaction term). The simplest approach is to compute the $O(N^2)$ particle–particle interactions directly, but this is prohibitively expensive when N is large. The well-known particle-in-cell method reduces the cost to $O(N \log N)$ using a grid-based Poisson solver [1,2]. The more recently developed treecode algorithm is a grid-free alternative that also reduces the cost to $O(N \log N)$ [3,4]. In this approach the particles are grouped into a set of clusters having a tree structure and the particle–particle interactions are replaced by particle–cluster interactions which are evaluated by a far-field multipole approximation. A divide-and-

☆ Supported by National Science Foundation grants DMS-9977371, DMS-0107187.
* Corresponding author.
E-mail address: christli@umich.edu (A.J. Christlieb).

0010-4655/$ – see front matter
doi:10.1016/j.cpc.2004.06.076

conquer strategy is used to select the clusters. Our implementation of the treecode uses a Taylor polynomial in Cartesian coordinates for the far-field approximation. The Fast Multipole Method is a more elaborate procedure that uses spherical harmonics, cluster-cluster approximations, and a pre-determined interaction list [5]. Since treecode algorithms are grid-free, they avoid spurious grid-based effects and may be especially appropriate for problems involving sharp gradients.

The first treecode algorithms were developed for particle simulations in free-space, but they can be combined with boundary integral methods to treat bounded domains as well [6–8]. In the present work we apply this approach to simulate electron dynamics in a Penning–Malmberg trap. This system was previously investigated by experiments and particle-in-cell simulations, and many interesting phenomena were revealed such as metastable crystalline states and complex dynamics [9,10]. The following sections describe the Penning–Malmberg trap, treecode algorithm, and boundary integral method, and then the simulation results are presented. Although a 2D example is treated here, the method applies more generally; it has been used to simulate a 1D virtual cathode [11] and vortex sheet roll-up in 3D incompressible fluid flow [12].

2. Penning–Malmberg trap

A Penning–Malmberg trap is a grounded conducting cylinder that confines an electron plasma to a bounded domain [9,10]. The electrons are confined by applying a magnetic field along the cylinder axis and holding the cylinder end caps at constant voltage, so that the electrons bounce back and forth along the magnetic field lines. For example in [10], the end cap potentials were held at $V \sim 50$ Volts and the magnetic field strength was $B \sim 1$ Tesla. Under these conditions, the time required for an electron to complete one bounce is much smaller than the characteristic $\vec{E} \times \vec{B}$ time scale. This implies that the plasma is well described by a 2D particle model in which the electrons behave like line charges being convected with velocity $\vec{v} = \vec{E} \times \vec{B}/B^2$ [13]. This is the model adopted here. In our simulations, the applied magnetic field $\vec{B}$ is a specified constant and the electric field $\vec{E}$ is computed by the treecode.

3. Treecode algorithm

Here we describe the evaluation of the electrostatic potential $\Phi(\vec{x})$ and afterwards we indicate how the electric field is obtained by a similar procedure. The algorithm has two main steps; the particles are grouped into a set of clusters having a tree structure and then $\Phi(\vec{x})$ is computed with the aid of the tree. The first level in the tree has a single cluster containing all the particles (called the root cluster) and each successive level is obtained by subdividing the clusters at the previous level into four sub-clusters (see Fig. 1).

Once the tree is constructed, the entire particle distribution is written as a union of clusters, $\{\vec{x}_i,\ i = 1,\ldots,N\} = \bigcup_{j=1}^{M} C_j$, where C_j denotes a cluster and M is the number of clusters in the union. The clusters are chosen adaptively from different levels in the tree,

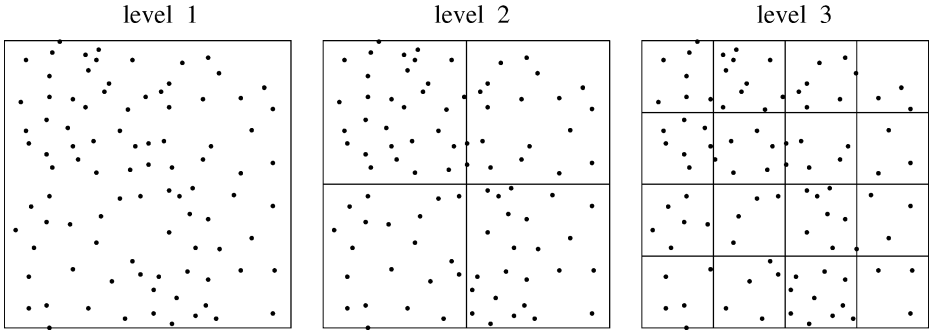

Fig. 1. Schematic diagram of tree construction.

but they are mutually disjoint. The potential is then expressed as

$$\Phi(\vec{x}) = \frac{1}{\epsilon_0}\sum_{i=1}^{N} G(\vec{x},\vec{x}_i)q_i = \frac{1}{\epsilon_0}\sum_{j=1}^{M}\sum_{i\in C_j} G(\vec{x},\vec{x}_i)q_i$$
$$= \frac{1}{\epsilon_0}\sum_{j=1}^{M}\Phi(\vec{x},C_j), \tag{2}$$

where

$$\Phi(\vec{x},C) = \sum_{i\in C} G(\vec{x},\vec{x}_i)q_i \tag{3}$$

denotes the particle–cluster interaction between $\vec{x}$ and C (see Fig. 2). The next step is to Taylor expand the potential $G(\vec{x},\vec{x}_i)$ with respect to $\vec{x}_i$ about the cluster center $\vec{x}_c$. Using Cartesian coordinates $\vec{x}_i = (x_i, y_i)$ this gives

$$G(\vec{x},\vec{x}_i) = G\big(\vec{x},(\vec{x}_i-\vec{x}_c)+\vec{x}_c\big)$$
$$\simeq \sum_{k=0}^{p}\sum_{l=0}^{k}\frac{1}{l!(k-l)!}\partial_{x_i}^{l}\partial_{y_i}^{k-l}G(\vec{x},\vec{x}_c)$$
$$\times (x_i-x_c)^l(y_i-y_c)^{k-l}, \tag{4}$$

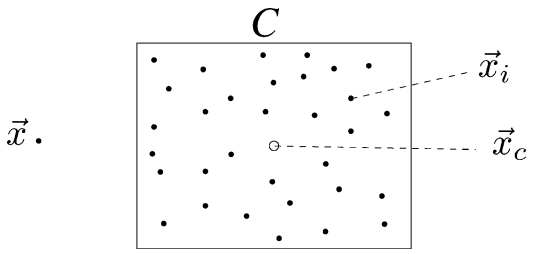

Fig. 2. Schematic diagram of a particle–cluster interaction. $\vec{x}$: particle, C: cluster, $\vec{x}_c$: cluster center, $\vec{x}_i$: generic particle in cluster C.

where p is the order of the approximation. Using Eq. (4), the particle–cluster interaction is given by

$$\Phi(\vec{x},C) \simeq \sum_{i\in C}\sum_{k=0}^{p}\sum_{l=0}^{k}\frac{1}{l!(k-l)!}\partial_{x_i}^{l}\partial_{y_i}^{k-l}G(\vec{x},\vec{x}_c)$$
$$\times (x_i-x_c)^l(y_i-y_c)^{k-l}q_i$$
$$= \sum_{k=0}^{p}\sum_{l=0}^{k}\frac{1}{l!(k-l)!}\partial_{x_i}^{l}\partial_{y_i}^{k-l}G(\vec{x},\vec{x}_c)$$
$$\times \sum_{i\in C}(x_i-x_c)^l(y_i-y_c)^{k-l}q_i$$
$$= \sum_{k=0}^{p}\sum_{l=0}^{k}T_{l,k}(\vec{x},\vec{x}_c)M_{l,k}(C), \tag{5}$$

where $T_{l,k}(\vec{x},\vec{x}_c)$ is the (l,k)-Taylor coefficient of the Green's function and $M_{l,k}(C)$ is the corresponding moment of the cluster. Note that the cluster moments are independent of the location $\vec{x}$, so they can be computed and stored for use with different $\vec{x}$. Also, the Taylor coefficients are independent of the particles in the cluster and they can be efficiently computed to high order using a three term recurrence relation [12]. The simulations below used either $p=4$ or $p=8$ for the order of the Taylor approximation.

In practice, the particle–cluster interaction $\Phi(\vec{x},C)$ is evaluated using the recursive function shown in Fig. 3. The decision to accept or reject the Taylor approximation is made by considering the ratio r_c/R, where r_c is the cluster radius and $R = |\vec{x}-\vec{x}_c|$ is the particle–cluster distance. If the approximation is rejected, the function descends to the next level of the tree and considers the sub-clusters of the given cluster. Finally, the potential $\Phi(\vec{x})$ is evaluated by calling **ComputePotential**$(\vec{x}, \mathit{root})$, where *root* is the entire particle distribution.

This completes our description of the treecode algorithm. The electric field $\vec{E} = -\nabla\Phi$ is obtained by a similar procedure after analytically differentiating the

function **ComputePotential**$(\vec{x}, C)$
  is the Taylor approximation sufficiently accurate?
    if yes, compute $\Phi(\vec{x}, C)$ by Taylor approximation in Eq. (5)
    if no, does C have any sub-clusters?
      if yes, call **ComputePotential**$(\vec{x}, C')$ for each sub-cluster C' of C
      if no, compute $\Phi(\vec{x}, C)$ by direct summation

Fig. 3. Function for evaluating $\Phi(\vec{x}, C)$.

potential in Eq. (5) with respect to $\vec{x}$; further details are explained in the references [3,4,12]. For problems defined on a simple bounded domain as in the Penning–Malmberg trap it is possible to use a modified Green's function given by the method of images, but we prefer to use a boundary integral method in order to accommodate more general domains in the future.

4. Boundary integral method

The potential is required to satisfy a Dirichlet boundary condition on the cylinder wall and hence it is the solution of a Poisson equation,

$$\Delta\Phi(\vec{x}) = -\frac{1}{\epsilon_0}\sum_{i=1}^{N}\delta(\vec{x}-\vec{x}_i)q_i,$$
$$\Phi(\vec{x}) = 0 \quad \text{for } \vec{x}\in\partial\Omega, \tag{6}$$

where $\partial\Omega$ is a circle (cross-section of the cylinder wall). The solution is expressed as $\Phi = \Phi_P + \Phi_H$, where Φ_P is a particular solution of the Poisson equation and Φ_H is a homogeneous solution that enforces the boundary condition. We take

$$\Phi_P(\vec{x}) = \frac{1}{\epsilon_0}\sum_{i=1}^{N} G(\vec{x},\vec{x}_i)q_i, \tag{7}$$

where $G(\vec{x},\vec{x}_i)$ is the free-space Green's function as in the previous section. The homogeneous solution satisfies

$$\Delta\Phi_H(\vec{x}) = 0, \quad \Phi_H(\vec{x}) = -\Phi_P(\vec{x}) \text{ for } \vec{x}\in\partial\Omega, \tag{8}$$

and it is expressed as a single layer potential,

$$\Phi_H(\vec{x}) = \int_{\partial\Omega} G(\vec{x},\vec{y})\sigma(\vec{y})\,\mathrm{d}s(\vec{y}), \tag{9}$$

where the source strength $\sigma(\vec{y})$ has to be determined. The physical interpretation is that $\Phi_H(\vec{x})$ arises from a distribution of line charges on the boundary. Next we let $\vec{x}$ approach a point $\vec{y}\in\partial\Omega$ and apply the boundary condition in Eq. (8). This leads to a 1st kind integral equation for $\sigma(\vec{y})$ which is solved by a collocation method at points $\vec{y}_j\in\partial\Omega$, $j=1,\ldots,N_{\partial\Omega}$. The potential at an arbitrary point in the domain is then given by

$$\Phi(\vec{x}) = \frac{1}{\epsilon_0}\sum_{i=1}^{N} G(\vec{x},\vec{x}_i)q_i + \sum_{j=1}^{N_{\partial\Omega}} G(\vec{x},\vec{y}_j)\sigma(\vec{y}_j)\omega_j, \tag{10}$$

where ω_j are the quadrature weights for the layer potential. Each term in Eq. (10) and the associated electric fields can be evaluated by the treecode.

5. Results

The treecode algorithm and boundary integral method were applied to simulate electron dynamics in a Penning–Malmberg trap. Time-stepping was performed by the leap-frog method [1,2]. Two examples of preliminary results are presented here.

Fig. 4 shows the merger of two regions each containing $N = 50$ K identically charged particles. The greyscale denotes the particle density. The initial particle locations were chosen randomly from a Gaussian pdf that was sharply peaked in each region and rapidly decaying away from the center. Initially the regions are disjoint and radially symmetric. At later times they draw closer together and form a rotating elliptic shape with almost uniform density. The small-scale

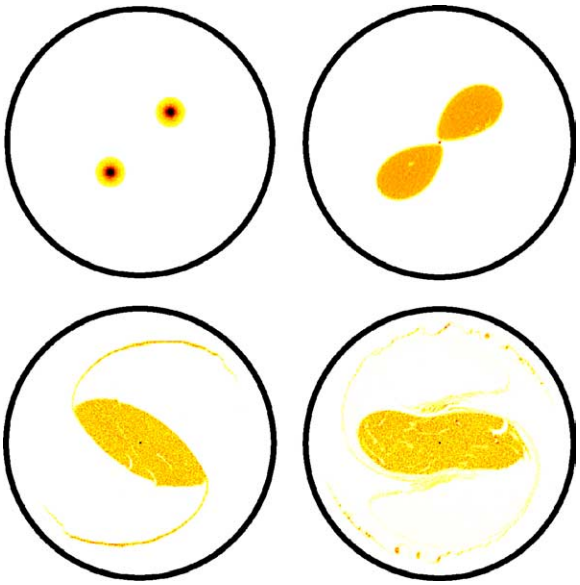

Fig. 4. Merger of two disjoint regions of electrons; time increases from left to right and top to bottom. The order of the Taylor expansion is $p = 8$.

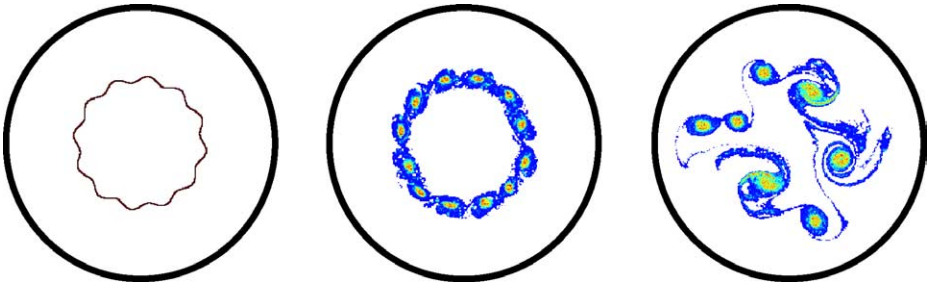

Fig. 5. Evolution of a ring of charge; time increases from left to right. The order of the Taylor expansion is $p = 4$.

features within the ellipse (gaps, scars) arise from the random fluctuations in the initial condition. Thin filaments are ejected from the ellipse and undergo a short wavelength Kelvin–Helmholtz instability. The results have some features in common with studies of vortex merger and axisymmetrization in 2D incompressible fluid dynamics [14,15].

Fig. 5 shows the time evolution of a ring of $N = 50$ K particles. At the initial time (left), the ring was given a radial perturbation of azimuthal wavenumber $m = 12$. At an intermediate time (middle), a regular state containing twelve crystals has formed, but at later times (right) this state breaks down and some of the crystals are in the process of merging, as in the previous example. This scenario agrees qualitatively with experimental results [9], but the breakup of the regular crystalline state in the present simulation is due to numerical errors rather than genuine physical effects.

6. Conclusions

We described a grid-free treecode algorithm for computing the electrostatic potential and forces induced by a set of charged particles, and presented simulations of electron dynamics in a Penning–Malmberg trap. With $N = 50$ K–100 K particles and 8th-order Taylor approximations, the treecode is much faster than direct summation. Since treecode algorithms are grid-free, they avoid spurious grid-based effects and may be especially appropriate for problems involving sharp gradients. Particle-in-cell methods are well-accepted in the plasma simulation community and techniques are available for reducing their grid-based effects, e.g., particle–particle/particle–mesh [2] and adaptive mesh refinement [16]. In future work we will assess the capability of the treecode algorithm in comparison with the particle-in-cell method.

References

[1] C.K. Birdsall, A.B. Langdon, Plasma Physics via Computer Simulation, IOP Publishing, Bristol, 1991.
[2] R.W. Hockney, J.W. Eastwood, Computer Simulation Using Particles, IOP Publishing, Bristol, 1988.
[3] J. Barnes, P. Hut, A hierarchical O($N \log N$) force-calculation algorithm, Nature 324 (1986) 446.
[4] S. Pfalzner, P. Gibbon, Many-Body Tree Methods in Physics, Cambridge Univ. Press, Cambridge, UK, 1996.
[5] L. Greengard, V. Rokhlin, J. Comput. Phys. 73 (1987) 325.
[6] V. Rokhlin, J. Comput. Phys. 60 (1985) 187.
[7] K. Nabors, F.T. Korsmeyer, F.T. Leighton, J. White, SIAM J. Sci. Stat. Comput. 15 (1994) 714.
[8] A. Grama, V. Kumar, A. Sameh, SIAM J. Sci. Comput. 20 (1998) 337.
[9] D.A. Schecter, D.H.E. Dubin, K.S. Fine, C.F. Driscoll, Phys. Fluids 11 (1999) 905.
[10] C.F. Driscoll, D.Z. Jin, D.A. Schecter, D.H.E. Dubin, Physica C 369 (2002) 21.
[11] A.J. Christlieb, R. Krasny, J.P. Verboncoeur, IEEE Trans. Plasma Sci. 32 (2004) 384.
[12] K. Lindsay, R. Krasny, J. Comput. Phys. 172 (2001) 879.
[13] F.F. Chen, Plasma Physics and Controlled Fusion, Plenum Press, New York, 1984.
[14] P. Koumoutsakos, J. Comput. Phys. 138 (1997) 821.
[15] P. Meunier, U. Ehrenstein, T. Leweke, M. Rossi, Phys. Fluids 14 (2002) 2757.
[16] J.-L. Vay, P. Colella, P. McCorquodale, B. Van Straalen, A. Friedman, D.P. Grote, Laser and Particle Beams 20 (2002) 569.