NUKLEONIKA 2012;57(2):313–316

ORIGINAL PAPER

# Implementation of a Monte Carlo method to a two-dimensional particle-in-cell solver using algebraic meshes

**Nevsan Sengil,**
**Özgür Tümüklü,**
**Mehmed Cevdet Çelenligil**

**Abstract.** Particle-in-cell (PIC) technique is a widely used computational method in the simulation of low density collisionless plasma flows. In this study, a new two-dimensional (2-D) electrostatic particle-in-cell solver is developed that can be applied to non-rectangular configurations.

**Key words:** particle-in-cell (PIC) • plasma flows • Monte Carlo (MC) method • coordinate transformation • Poisson's equation • Boltzmann relation

N. Sengil✉
Civil Air Transportation Management Department,
University of Turkish Aeronautical Association,
Türkkuşu Campus, 06790 Etimesgut/Ankara, Türkiye,
Tel.: +90 506 908 5425, Fax: +90 312 342 8460,
E-mail: nsengil@thk.edu.tr

Ö. Tümüklü, M. Cevdet Çelenligil
Department of Aerospace Engineering,
Middle East Technical University,
06800 Ankara, Türkiye

Received: 10 October 2011
Accepted: 9 December 2011

## Introduction

Electric propulsion devices offer a good alternative to the chemically based thrusters. These devices expel high speed plasma into vacuum, and their high specific impulse characteristics make them very useful in space applications. In the study of the plasma behavior, computational simulations are used in addition to the analytical and experimental methods, and PIC technique is widely used to simulate collisionless low density plasmas [1]. In this study, the main aim has been to develop a new two-dimensional (2-D) electrostatic PIC solver that can be used on non-rectangular geometries.

In electrostatic PIC solvers, potentials and electric fields on the mesh points are calculated by solving Poisson's equation. While equations of motion of the ions are computed using algorithms such as Leap-Frog, electron distributions are modeled using the Boltzmann relation [3]. In the first part of this study, a 2-D non-rectangular physical domain (which consists of two walls and two open boundaries) is meshed using an algebraic grid, and the physical domain with non-uniform meshes are mapped to a rectangular computational domain with uniform meshes. Then, 2-D Poisson's equation is transformed to the computational domain and is solved using a MC method. The advantages of the MC method are: easy extension of the higher dimensional problems, easy implementation on parallel computers, a straightforward implementation of the complex boundary conditions with severe gradients, and availability of

local solutions without calculation of the whole field [4, 8, 10, 11]. In the second part of the study, ions are traced using equations of motion. Electron densities at mesh points are modeled using the Boltzmann relation. While ion movements are calculated in the physical domain, cell data are obtained from computational domain to increase efficiency [9].

### Poisson's equation in computational domain

Many methods are available to solve Poisson's equation [6], and in this study, a new Poisson solver based on a MC method is developed which is capable of handling both Neumann and Dirichlet boundary conditions. As for verification, a Poisson equation

$$(1) \qquad \frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2} = -8\pi^2 \sin(2\pi x)\sin(2\pi y)$$

is solved on a rectangular domain where Neumann boundary conditions are implemented on the left and right boundaries with zero derivatives. The results are

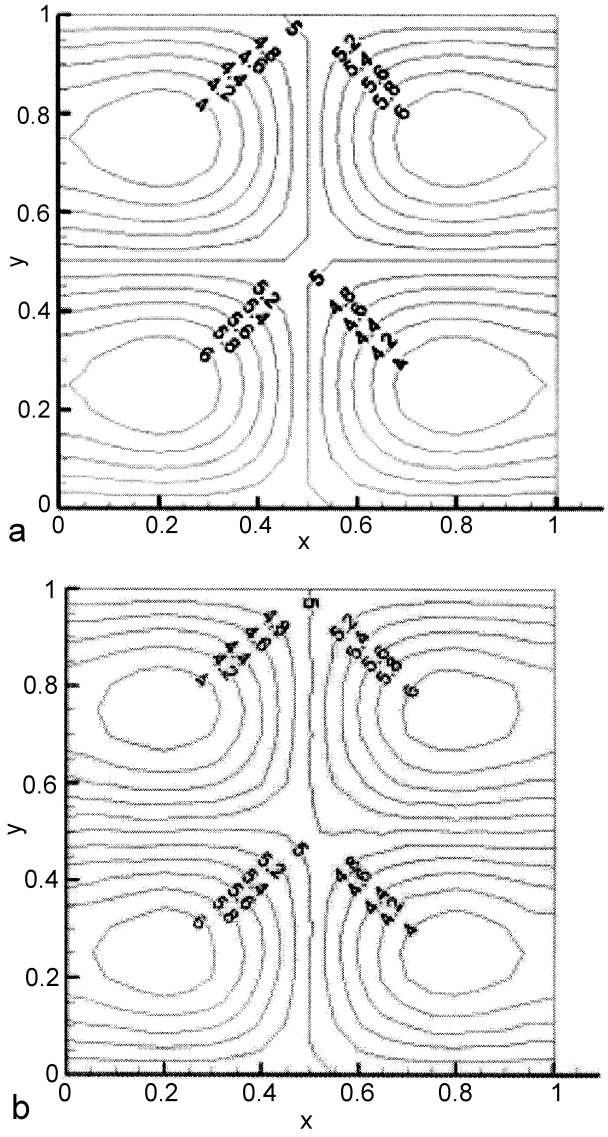


**Fig. 1.** Solution of a Poisson's equation (a) FFT solver, (b) Monte Carlo solver.

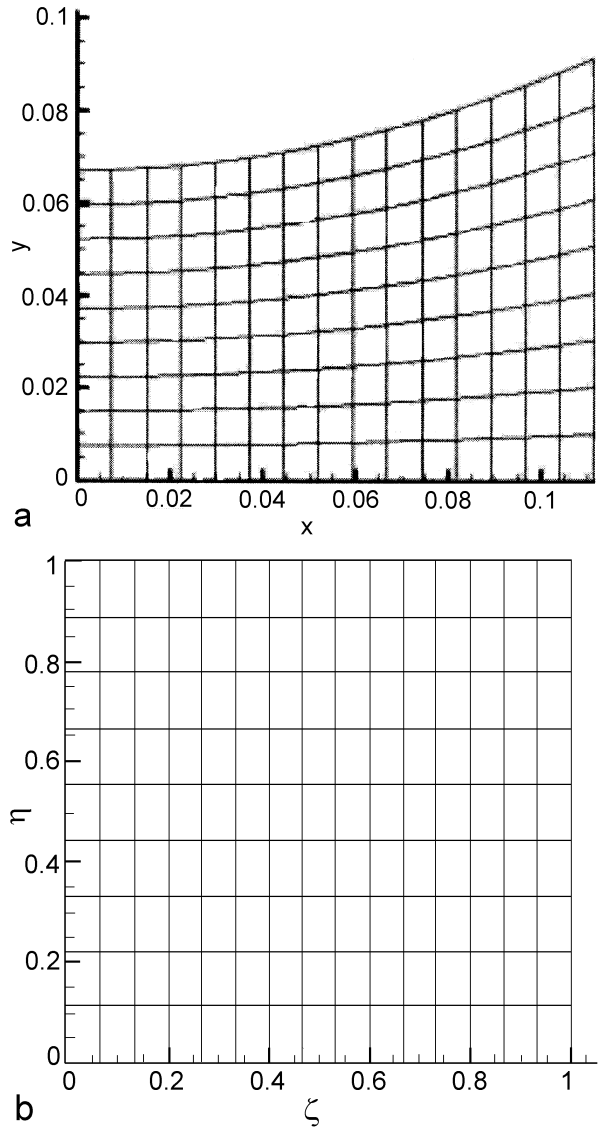


**Fig. 2.** (a) Physical domain. (b) Computational domain.

shown to be very close to those obtained by a different solver based on fast Fourier transforms (FFT) [7] as shown in Fig. 1. Less than 2 percent difference is observed locally throughout the domain.

Later, a 2-D non-rectangular physical domain is meshed using an algebraic grid generator (with 15 cells in the $x$-direction and 9 cells in the $y$-direction) and the physical domain $(x,y)$ is mapped to a square shaped computational domain $(\zeta,\eta)$ with uniform meshes using analytical relations $\zeta = x/d$ and $\eta = y/(ax^2 + bx + c)$ where $a$, $b$, $c$ and $d$ are constants (see Fig. 2).

The Poisson equation $\phi_{xx} + \phi_{yy} = -[\rho(x,y)/\varepsilon_0]$ is transformed to a new computational domain as follows:

$$(2) \quad \frac{1}{d^2}\phi_{\zeta\zeta} - \frac{2\eta(2a\cdot d\zeta + b)}{d(a\,d^2\zeta^2 + b\,d\zeta + c)}\phi_{\zeta\eta} - \frac{2\eta\left[a(a\,d^2\zeta^2 + b\,d\zeta + c) - (2a\cdot d\zeta + b)^2\right]}{(a\,d^2\zeta^2 + b\,d\zeta + c)^2}\phi_{\eta} + \frac{\left[1 + \eta^2(2a\,d\zeta + b)^2\right]}{(a\,d^2\zeta^2 + b\,d\zeta + c)^2}\phi_{\eta\eta} = -\frac{\rho(\zeta,\eta)}{\varepsilon_0}.$$

This equation can be discretized by finite differencing, as follows:

$$(3)\quad \left[-\left(\frac{2}{d^2(\Delta\zeta)^2}\right)+\left(\frac{2\eta(2a\,d\zeta+b)}{d(a\,d^2\zeta^2+b\,d\zeta+c)\Delta\zeta\,\Delta\eta}\right)-\left(\frac{2\left[1+\eta^2(2a\,d\zeta+b)^2\right]}{(a\,d^2\zeta^2+b\,d\zeta+c)^2(\Delta\eta)^2}\right)\right]\phi_{i,j}+\left[\left(\frac{1}{d^2(\Delta\zeta)^2}\right)\right]\phi_{i+1,j}+\left[\left(\frac{1}{d^2(\Delta\zeta)^2}\right)-\left(\frac{2\eta(2a\,d\zeta+b)}{d(a\,d^2\zeta^2+b\,d\zeta+c)\Delta\zeta\cdot\Delta\eta}\right)\right]\phi_{i-1,j}+\left[\left(\frac{\left[1+\eta^2(2a\,d\zeta+b)^2\right]}{(a\,d^2\zeta^2+b\,d\zeta+c)^2(\Delta\eta)^2}\right)-\left(\frac{\eta\left[a(a\,d^2\zeta^2+b\,d\zeta+c)-(2a\,d\zeta+b)^2\right]}{(a\,d^2\zeta^2+b\,d\zeta+c)^2\Delta\eta}\right)-\left(\frac{2\eta(2a\,d\zeta+b)}{d(a\,d^2\zeta^2+b\,d\zeta+c)\Delta\zeta\,\Delta\eta}\right)\right]\phi_{i,j+1}+\left[\left(\frac{\left[1+\eta^2(2a\,d\zeta+b)^2\right]}{(a\,d^2\zeta^2+b\,d\zeta+c)^2(\Delta\eta)^2}\right)+\left(\frac{\eta\left[a(a\,d^2\zeta^2+b\,d\zeta+c)-(2a\,d\zeta+b)^2\right]}{(a\,d^2\zeta^2+b\,d\zeta+c)^2\Delta\eta}\right)\right]\phi_{i,j-1}+\left[\left(\frac{2\eta(2a\,d\zeta+b)}{d(a\,d^2\zeta^2+b\,d\zeta+c)\Delta\zeta\,\Delta\eta}\right)\right]\phi_{i-1,j+1}=-\frac{\rho(\zeta,\eta)}{\varepsilon_0}.$$

After dividing each term by the coefficient of $\phi_{i,j}$, the coefficients in the front of $\phi_{i+1,j}$, $\phi_{i-1,j}$, $\phi_{i,j+1}$, $\phi_{i,j-1}$, $\phi_{i-1,j+1}$, are called as the sensitivity coefficients in this Monte Carlo method. They are related to the probability of moving in different directions within the algorithm. The summation of these coefficients should give unity, and none of them should be negative [5]. In the present calculations, these conditions are met and their values at various $(\zeta,\eta)$ locations (in the transformed domain) are presented in Table 1.

Note that Poisson's equation is a linear partial differential equation and can be solved in two steps using the superposition principle [12]. For the first solution, Laplace's equation, $\nabla^2\phi_1 = 0$, is solved with the boundary conditions of the original problem. Then, Poisson's equation, $\nabla^2\phi_2 = -(\rho/\varepsilon_0)$ with zero boundary conditions is solved and the two solutions are added, $\phi = \phi_1 + \phi_2$. The solution gives the discrete potential values at the grid points of the computational domain. Next, the electric field is calculated from

$$(4)\qquad \vec{E} = -\vec{\nabla}\,\emptyset$$

Note that one needs to use transformed form of this equation and it is solved using finite difference schemes.

## Tracing of plasma particles

In numerical studies, plasma ions are introduced to the physical domain from the open boundaries and finite numbers of ions enter and leave the physical domain at each time step. In this study, ions are sent into the flow region from ion reservoirs. The number of ions and their velocities are designated by the boundary conditions, and positions of ions in the reservoir are chosen randomly. A force is applied on each ion because of the electric field ($\vec{E}$). The equations of motion of these particles can be solved using different algorithms. In this study, the Leap-Frog algorithm is used to calculate particle advancing.

Electrons move much faster compared to ions, because of their small masses. It is assumed that electrons settle themselves instantly according to the electric fields. As a result of this, a kinetic-ions and fluid-electrons approach is used, and electrons are distributed in the physical domain using the Boltzmann distribution assuming isothermal electrons. The electron number density is given as

$$(5)\qquad n_e = n_0\exp\left(\frac{e\phi}{kT_e}\right)$$

In this equation $k$ is the Boltzmann constant, $n_0$ is the electron number density for $\phi = 0$, $T_e$ is the temperature and the potential is $\phi$. After solving equations of motion for a time step, ions are placed in their new positions in the physical domain. At this stage, boundary conditions are also taken into account. Next, using analytical relations between physical and computational domains, ion cell data are calculated in the computational domain and charge densities on the mesh points are found. These charge densities are used in the calculation of the potential values of the same mesh points. In the PIC method, this loop continues until a predetermined condition is met.

**Table 1.** Sensitivity coefficients at different locations of computational domain

| $(\zeta,\eta)$ | Probability going right ($\rightarrow$) | Probability going up ($\uparrow$) | Probability going left ($\leftarrow$) | Probability going down ($\downarrow$) | Probability going up and left ($\nwarrow$) | Total |
|---|---|---|---|---|---|---|
| (0,0) | 0.250 | 0.250 | 0.250 | 0.250 | 0.000 | 1.000 |
| (0.20, 0.56) | 0.261 | 0.224 | 0.234 | 0.255 | 0.026 | 1.000 |
| (0.40, 0.22) | 0.270 | 0.219 | 0.250 | 0.241 | 0.020 | 1.000 |
| (0.53, 0.44) | 0.289 | 0.183 | 0.235 | 0.239 | 0.054 | 1.000 |
| (0.80, 0.55) | 0.328 | 0.120 | 0.226 | 0.223 | 0.102 | 1.000 |
| (1,1) | 0.379 | 0.002 | 0.140 | 0.240 | 0.239 | 1.000 |

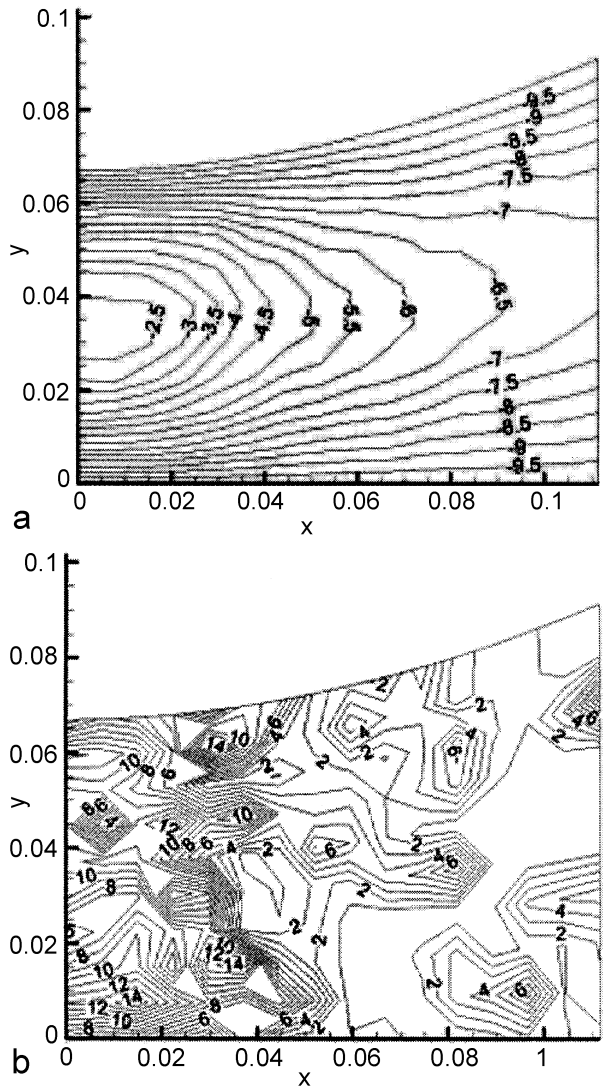


**Fig. 3.** Plasma flow properties in a 2-D nozzle. (a) – Field potentials in volts; (b) – number density ($\times 10^{11}$) in 1 $m^3$.

An example problem is solved with the present PIC solver for a (non-rectangular) 2-D nozzle flow. The calculated results of the field potentials and charge densities are shown in Fig. 3. In these calculations, Neumann boundary conditions (with zero derivatives) are used at the right and left boundaries, and Dirichlet boundary conditions (with –10 V potential) are stipulated at the top and bottom boundaries.

The present results represent the initial findings of an ongoing research work on plasma flow simulations and have some shortcomings. For example, the results for the non-rectangular flow configuration (as shown in Fig. 3) are not validated due to the lack of experimental data and simulation results of other researchers. It is observed that when positive potentials are stipulated at the top and bottom walls the solutions start to diverge. These instabilities may either arise from the nonphysical statistical noise of the PIC method or the nonlinear response of the plasma flow [2]. It is assessed that experimental verifications are needed to understand this issue.

## Summary and conclusions

In this study, a MC method is used to solve Poisson's equation which is the main equation of the electrostatic PIC solver. To validate the Monte Carlo computations, the solutions of Poisson's equation (for both Neumann and Dirichlet boundary conditions) on a rectangular domain are compared successfully with other programs. Then, a non-rectangular physical domain is meshed using an algebraic grid generation method. The physical domain is mapped to a non-dimensional square shaped computational domain with uniform cells. The computational domain provides fast computation of particle cell data in addition to the easy implementation of the MC method to solve Poisson's equation. Poisson's equation is also transformed to the new domain using analytical relations between physical and computational domains. Finally, PIC simulations are realized to calculate the plasma flows. The present calculations are the initial work of an ongoing research on plasma flow simulations and the results are found to be promising.

## References

1. Birsdall CK, Langdon AB (2005) Plasma physics via computer simulation. Taylor & Francis, New York
2. Buchner J (2007) Vlasov-code simulation. In: Usui H, Omura Y (eds) Advance methods for space simulations. Terrapub, Tokyo, pp 23–46
3. Chen FF (1984) Introduction to plasma physics and controlled fusion plasma physics. Plenum Press, New York
4. DeLaurentis JM, Romero LA (1990) A Monte Carlo method for Poisson's equation. J Comput Phys 90:123–140
5. Haji-Sheikh A (2009) Monte Carlo methods. In: Minkowycz WJ, Sparrow EM, Murthy JY (eds) Handbook of numerical heat transfer, 2nd ed. John Wiley & Sons, New York, pp 673–723
6. Hockney RW, Eastwood JW (1988) Computer simulation using particles. IOP Publishing, Bristol
7. Intel Corporation (2010) Intel®Math Kernel Library (Intel®MKL)
8. Rosenthal JS (2000) Parallel computing and Monte Carlo algorithms. Far East J Theor Stat 4:207–236
9. Sengil N, Edis FO (2011) Fast cell determination of the DSMC molecules in multi-stage turbo molecular pump design. Comp Fluids 45;1:202–206
10. Shentu J, Yun S, Cho NZ (2007) A Monte Carlo method for solving heat conduction problems with complicated geometry. Nucl Eng Technol 39;3:207–214
11. Thompson JJ, Chen PYP (1970) Heat conduction with internal sources by modified Monte Carlo methods. Nucl Eng Des 12:207–214
12. Vahedi V, Dipeso G (1997) Simultaneous potential and circuit solution for two-dimensional bounded plasma simulation codes. J Comput Phys 131:149–163