REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
15-12-2003	Technical Paper	

4. TITLE AND SUBTITLE

Simulation of Plasma Expansion Using a Two-Timescale Accelerated Particle-in-Cell Method

5a. CONTRACT NUMBER
F04611-99-C-0025
5b. GRANT NUMBER
5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

M.R. Gibbons (ERC, Inc.); M. Santi (Advatech Pacific, Inc.); D.B. VanGilder and J.M. Fife (AFRL/PRSS)

5d. PROJECT NUMBER
4847
5e. TASK NUMBER
0052
5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Engineering Research Corporation (ERC), Inc.
10 E. Saturn Blvd.
Edwards AFB CA 93524-7680

Air Force Research Laboratory (AFMC)
AFRL/PRSS
1 Ara Drive
Edwards AFB CA 93524-7013

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Air Force Research Laboratory (AFMC)
AFRL/PRS
5 Pollux Drive
Edwards AFB CA 93524-7048

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S NUMBER(S)
AFRL-PR-ED-TP-2003-324

12. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for public release; distribution unlimited.

13. SUPPLEMENTARY NOTES

For presentation at the 42nd AIAA Aerospace Sciences Meeting and Exhibit in Reno, NV, taking place 5-8 January 2004.

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
					Leilani Richardson
a. REPORT	b. ABSTRACT	c. THIS PAGE	A	7	19b. TELEPHONE NUMBER *(include area code)*
Unclassified	Unclassified	Unclassified			(661) 275-5015

Simulation of Plasma Expansion Using a Two-Timescale Accelerated Particle-in-Cell Method

M.R. Gibbons
ERC, Inc.
Edwards AFB, CA

M. Santi
Advatech Pacific, Inc.
Edwards AFB, CA

D. B. VanGilder and J. M. Fife
U.S. Air Force Research Laboratory
Edwards AFB, CA

COLISEUM is an application programming interface which performs calculations of plasma propagation and interaction with arbitrary 3-D surfaces. The applications of COLISEUM are wide-ranging, but include simulating ion source configurations inside vacuum chambers and predicting sputtering and re-deposition on surfaces. COLISEUM allows users to easily define complicated 3-D geometries using off the shelf CAD software, then select from a set of plasma expansion models of varying fidelities and numerical complexity to perform the solution. Once the object surfaces are created, the user can run different types of simulations for the same geometry. With this system, low fidelity models can be used to verify the geometry and boundary conditions, and to obtain first-order predictions. Higher fidelity models are then used to obtain more accurate predictions with greater cost in computation time. Detailed plasma expansion calculations are performed with a particle-in-cell (PIC) algorithm which includes wall collisions and wall recombination. This paper presents an acceleration scheme which temporarily decouples the ion and neutral propagation loops to speed convergence.

Introduction

The Air Force Research Laboratory is leading development of a software package named COLISEUM, which is capable of self-consistently modeling plasma propagation and interactions with arbitrary 3-D surfaces. Three important requirements have been placed on COLISEUM: It must be USABLE, FLEXIBLE, and EXPANDABLE.

USABLE means a typical engineer is able to set up and run a typical low-fidelity case in less than one day with less than three days training.

FLEXIBLE means COLISEUM is able to simulate plasma sources and vacuum test facilities with accurate definition of their 3-D geometries. In addition to being able to simulate multiple geometries, COLISEUM is flexible in its use of plasma simulation algorithms. Problem set-up and geometry definition is preformed once. Then, the user may select from a set of interchangeable plasma simulation algorithms to perform the solution. If fast execution is desired, a low-fidelity technique can be selected such as ray tracing. For higher fidelity (at the cost of longer run-time), something like Particle-In-Cell (PIC) can be used.

EXPANDABLE means COLISEUM can be easily expanded to incorporate new plasma simulation algorithms, new capabilities, or improved efficiency. Furthermore, as new plasma simulation algorithms are added, old ones will continue to function.

This paper describes an acceleration scheme for the PIC simulation modules. Ion beam calculations present difficulties because of the significantly different species speeds. Ions can be accelerated to many times the speed of neutrals. Neutrals are expelled from the ion source because of incomplete ionization, and neutrals are created during the interation of ions with vacuum chamber walls. The species are still collisionally coupled. Extremely long convergence times are typically required for the neutral species to relax using an ion time step. The acceleration scheme temporarily decouples the ion and neutral propagation loops to speed convergence by alternating two algorithms. The first algorithm is the normal PIC simulation. While this algorithm operates, the ion flux to the walls is accumulated. After the ion flux reaches a quasi-steady state, the neutrals are advanced using a large time step with the ion species frozen. Neutrals are created at the walls in accordance with the stored ion flux. Once the neutrals reach a steady state, the code returns to the full algorithm which again accumulates the ion flux at the walls. The algorithms are alternated until the various particle densities reach a global steady state.

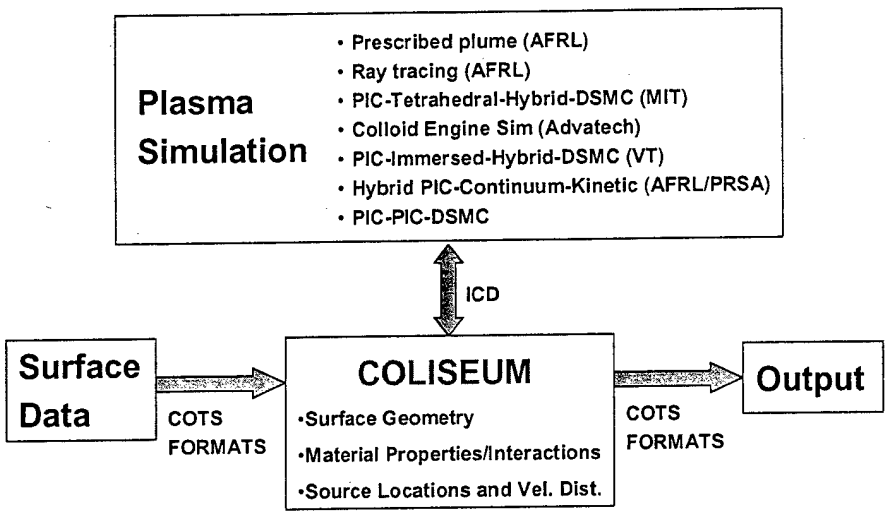

Fig. 1. Architecture for using various interchangeable plasma simulation techniques with the same 3-D surface geometry.

Code Architecture

Fig. 1 shows how the COLISEUM Application Programming Interface (API) works with a set of various interchangeable plasma simulations (applications). In general, the COLISEUM API can be viewed as a framework in which 3-D plasma simulations can be quickly integrated. Common calculations (such as those related to surfaces, material properties, and flux sources) are standardized, grouped, and provided as a resource library (data and subroutines) to each simulation. This resource library takes the form of a `.lib` file that users link with their set of plasma simulation routines.

Plasma simulation modules are the primary functional components of COLISEUM. They calculate plasma propagation of matter on the volume domain. They contain algorithms, such as ray tracing, fluid, PIC, DSMC, or hybrids thereof, which perform a solution subject to pre-set boundary conditions. Plasma simulation modules are interchangeable. They all conform to the COLISEUM API which is formalized in the Interface Control Document (ICD).

The COLISEUM resource library functions support tasks common to all types of plasma simulations. They handle boundary conditions, and provide support to plasma simulation modules.

The purpose of this modular design is to give COLISEUM flexibility and expandability. A large number of plasma simulation modules are desired to allow flexibility in solving a variety of different problems. The ICD is, therefore, very important because it describes for authors of plasma simulation modules a) what inputs and boundary conditions must be recognized, b) what outputs are expected, and c) what COLISEUM resource functions are available. The ICD and COLISEUM resource library may be distributed to outside groups so that COLISEUM can be expanded through addition of new plasma simulation modules.

The COLISEUM API standardizes the definition of a 3-D plasma problem by providing strict specifications on three categories of information: surface geometry, surface properties, and sources. The manners in which these three categories of information are defined, input, stored, and accessed are described below.

Surfaces

Surfaces are modeled in finite-element fashion as contiguous triangular elements joined at the vertices (nodes). COLISEUM does not generate 3-D geometries or surfaces; instead, it imports them from other software.

Users create custom geometries using almost any mainstream commercial 3-D solid modeling package. Then, they use finite element analysis software to mesh the surface of their geometry as if they were going to perform a structural analysis using thin shells. The user then saves the meshed surface file in one of several formats (ANSYS, ABAQUS, NASTRAN, MGEN), which are readable by COLISEUM.

This concept of separating the surface geometry definition from the plasma calculation has proven very successful. It greatly reduced development time and cost by eliminating the need for a separate surface definition module. It allows users to choose which software to use in defining geometries. Also users can import into COLISEUM geometries that have already been defined for other reasons (structural, thermal, etc.).

Surface Properties

The user may provide three databases in conjunction with a surface geometry: a component database, a material database, and a material interaction database.

The component database associates specific surfaces with component names and material names. These associations are established by using a component number which is specified in the ANSYS file using integer values in the elastic modulus field. For example, the component database may specify component number 34 as component name "chamber_wall" and material "stainless_steel."

The material database associates component names with material names and material properties. The plasma simulations RAY and PRESCRIBED_PLUME require, in addition to material name, molecular weights, and charges (in the case of ions).

The material interaction database contains the sputter yield coefficients and sticking coefficients of one material interacting with the other, e.g. between Xe^+ and Aluminum.

Sources

Sources are modeled as having a specific velocity distribution, $f_s(\vec{r},\vec{v},t)$, that is a function of position on the surface, of three-dimensional velocity space, and of time:

$$\dot{m}_s = \int_S \int_V f_s(\vec{r},\vec{v},t) d^3v ds \qquad (1)$$

Rather than specify $f_s(\vec{r},\vec{v},t)$ directly, however, three COLISEUM resource library functions are provided for each source type. These a) give the distribution of velocities at some point P in space due to the source, b) provide a random sample from the source velocity distribution at the surface, or c) update the source to be valid at some new time, t.

This method is extremely descriptive and general. Plasma simulation modules may use the three source functions to treat the source distribution function in various ways. For example, using the first function, a plasma simulation module can be written to treat the source element as a source for geometric ray tracing. Alternately, particle methods can use the second sample from the velocity distribution and introduce particles randomly over the full element surface. Therefore, this choice of source definition methods gives COLISEUM great flexibility by enabling a wide variety of plasma simulation techniques with the same source definition.

Plasma Simulation Modules

Four plasma simulation modules are being incorporated into COLISEUM. PRESCRIBED_PLUME allows the user to import a previously calculated or measured plume field. The plume is superimposed over the user's surface geometry. RAY uses ray tracing to calculate the flux from all sources onto all surface nodes. These modules then calculate sputter and re-deposition rates at each surface node. AQUILA is a PIC-DSMC module with an unstructured tetrahedral-mesh being developed by MIT. In addition, Virginia Polytechnic Institute (VPI) is developing a PIC-DSMC simulation which uses immersed mesh techniques. Details of these modules have been presented before. [1,2,3]

Execution Sequence

To run a COLISEUM case, three steps are typically required: geometry definition, surface meshing, and execution of the simulation.

Geometry Definition

Users generate the 3-D geometry using any suitable software. AFRL uses SolidWorks™. Enclosed geometries such as vacuum chambers are typically created as hollow objects. In these cases, it is convenient for the geometry definition software to have cutaway-capability so that the interior of the enclosure can be visualized.

Surface Meshing

The surface meshing process may also be performed by a variety of software packages. The only requirements are that the software must produce output in one of the following formats: ANSYS, MGEN, and NASTRAN. AFRL is currently using COSMOS DesignStar, which is a finite element analysis package that supports ANSYS output format.

Execution of the Simulation

COLISEUM executes batch commands that the user provides in a text input file. The commands are executed sequentially as they appear in the input file. Each command may have some number of parameters separated by spaces or commas.

Geometry definition typically takes approximately 6 hours for medium-complexity geometries. Typical run times for low-fidelity cases (using PRESCRIBED_PLUME or RAY) take approximately 20 minutes on a 2 GHz Intel Pentium 4 workstation. Once more detailed physics are incorporated with plasma algorithms such as PIC-DSMC, run times are expected to be between 20 minutes and 20 hours, depending on the level of fidelity and on the initial conditions.

This illustrates a key feature of COLISEUM. From scratch, a user can define a complete three-dimensional problem and generate a first order solution all in less than one workday. Then, for higher fidelity solutions, the problem does not have to be redefined. Since the plasma simulation modules are interchangeable, a higher-fidelity algorithm may be immediately started for an overnight run.

Acceleration of Particle Simulations

One universal difficulty modeling ion sources in vacuum chambers is the long computation time required for the system to relax to a steady state. We are incorporating several techniques into COLISEUM to mitigate this problem.

Subcycling

There are three main phenomena that require temporal resolution during the plasma simulation of the thruster: the particle velocity, the electric field, and the collisions. We must resolve the particle crossing of spatial gradients.

Although these simulations are relaxing to an equilibrium state, we must ensure that the field solution does not suffer from numerical error or instability. We must also resolve the collision frequencies of the various particles. In a previous paper we showed the savings in computation time afforded by subcycling some particle species.

The ion flow velocities are in the range of $2 \cdot 10^4$ m/s, and the neutral thermal velocity is near 200 m/s. The smallest simulation volume elements are 0.01 m on a side. To properly sample spatial gradients the particles should traverse less than one third of an element during a particle move. Thus, the particle motion time limit is 10^{-7} for ions and 10^{-5} for neutrals.

In the PIC-DSMC calculations the plasma is assumed to be quasi-neutral with electrons subject to the Boltzmann equation. The finite difference leap-frog method is used to advance the position and velocity of the ion particles. We can use linear theory to determine the stability and accuracy of this scheme as a function of numerical discretization in space and time[4]. Consider Xe with T_e = 2 eV and Δx = 1 cm. The algorithm remains stable and accurate for Δt less than 1 μsec.

The mean time between collisions, τ, is given by the inverse of $n\sigma\langle v_{rel}\rangle$. The maximum neutral density in these simulations is approximately $10^{18}/m^3$. Table 1 lists the collision times given the typical relative velocities between species. A simulation time step of less than 10 μsec is necessary to properly resolve collisions.

Table 1: Collision properties

Collision	<Vrel> (m/s)	σ (m2)	τ (sec)
Xe – Xe elastic	200	5.9e-19	8.4e-3
Xe – ion elastic	20000	8.3e-20	1.2e-3
Xe – Xe+ CEX	10000	1.14e-18	8.8e-5
Xe – Xe++ CEX	20000	6.8e-19	7.4e-5

The preceding analysis indicates a situation where the phenomena to be modeled operate on two disparate time scales. The field solution and ion motion require a time step near 0.1 μsec while collisions and neutral motion require a time step near 10 μsec. For computational speed we have chosen to subcycle the ion motion and field solve. Savings in collision computations and the neutral particle push lead to significant improvements in computation time.

A flow chart of the subcycle algorithm is shown in Fig. 2. The main loop encompasses the procedures to advance all particles one full time step about 10 μsec for these simulations. Within each main loop the fast particles, typically the ions, are subcycled. For the results shown below there are 100 subcycle steps.

The subcycle loop includes the time advance of the particle velocities and positions. Fast particles are injected from sources. The E-field is also updated each iteration. Since the code presently assumes quasi-neutrality with Boltzmann electrons, the potential is obtained directly from the ion density. Since this procedure is computationally fast, it does not significantly impact the speed of the subcycle iterations. In the future when the potential is calculated from Poisson's equation, the solution of the elliptic equation on the mesh will modify the results presented here.

After completion of subcycling, positions and velocities of the slow particles are advanced one full time step. Slow particles are injected from sources. Finally, all particles participate in DSMC collisions.

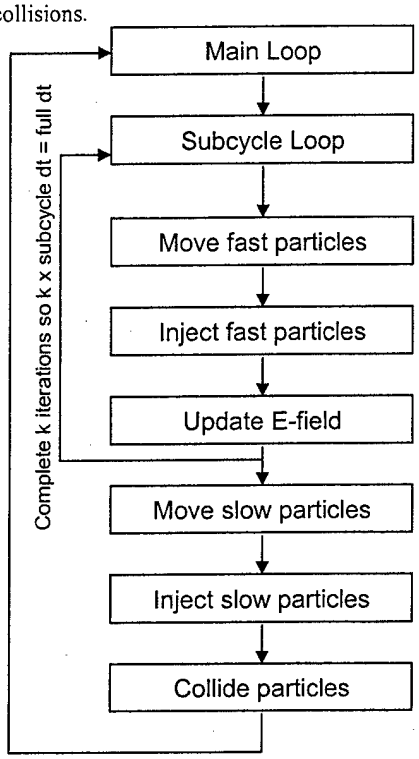

Fig. 2. Flow diagram for the time advance with subcycling.

Surface Source of Neutrals

Ions impinge on all surfaces within a vacuum chamber and are neutralized. The process of reflection and accommodation significantly decreases the speed of the particles. The neutralized particles can be treated as a source with the flux determined by the incoming flux of ions. The simulation can be accelerated by using only the neutral sources, including no ion dynamics, and running on the long time scale of the neutrals. The ions exist as a background available for CEX and elastic collisions. These background particles can be used directly with DSMC collisions. Once the neutrals reach a steady state, the data is stored in a restart file.

A flowchart of the surface source algorithm is shown in Fig. 3. The algorithm alternates between what we have defined as the "normal" algorithm and the "SurfS" algorithm. The "normal" algorithm is the same as the subcycling algorithm shown in Fig. 2. This is the fully self-consistent algorithm. The "SurfS" algorithm has no subcycling and only advances the neutral particles. This algorithm utilizes the wall sources that create neutrals based on the neutralization of the ion flux. The main loop time step for the "normal" algorithm and the time step for the "SurfS" algorithm are the same and

correspond to the neutral time scale. The "normal" algorithm is run for relatively few time steps in order to bring the ions to an intermediate steady state. The "SurfS" algorithm is run for many time steps similar in number to a simulation which does not use this acceleration technique. The total number of time steps to reach steady state will be about the same for either a standard simulation or for the surface source simulation. Since the surface source simulation is dominated by "SurfS" time steps, the CPU time is much less.

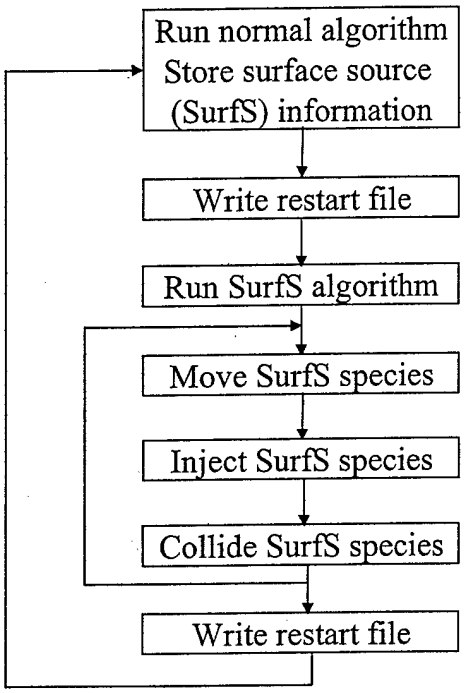

Fig. 3. Flow diagram for the surface source algorithm.

In the surface source simulation there is one change to the "normal" algorithm from the standard subcycling algorithm. The flux of particles to each surface element is accumulated for each particle move step. The accumulation of the flux provides variance reduction over many time steps. Also the particle velocity information is stored for the particles arriving at each surface element. This provides the velocity distribution information for the incoming flux. Because of memory limitations only a finite number of particles are stored (typically 100) in a First In First Out (FIFO) list. We sample from this list to generate the surface source neutrals during implementation of the "SurfS" algorithm.

The velocity distribution for the incoming particles is needed for the diffuse surface scattering algorithms. At present we are using Lord's extension of the Cercignani-Lampis model[5,6]. This model satisfies detailed balance and transitions from diffusive scatter with no accommodation to complete accommodation. The tangential components satisfy the probability scattering kernel

$$P(v_i, v_r) = \frac{1}{\sqrt{\pi a(2-a)}} \exp\left[-\frac{[v_r - (1-a)v_i]^2}{a(1-a)} \right]$$

Here v_i is the incident velocity. Notice that these components can be sampled in the standard manner for a Gaussian distribution.

The distribution for the normal component of the velocity, u, is modified to be appropriate for flux from a surface. This is similar to the diffuse scattering where the distribution is multiplied by the normal velocity. The scattering kernel becomes

$$P(u_i, u_r) = \frac{2u_r}{a} I_0\left[\frac{2(1-a)^{1/2} u_r u_i}{a} \right] \exp\left[-\frac{u_r^2 - (1-a)u_i^2}{a} \right]$$

Here I_0 is the modified Bessel function of order zero. Figure 4 shows the energy distribution for reflected particles. The incident particles had kinetic energy equivalent to ten times the wall temperature, and the accommodation coefficient was 0.5.

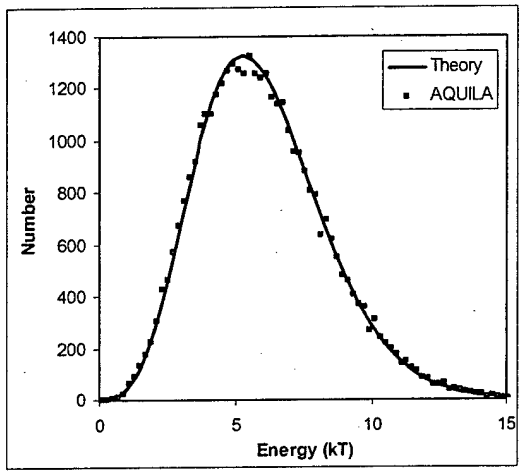

Fig. 4. Energy distribution of neutrals after reflection and accommodation.

After the "normal" algorithm is run for a user specified number of time steps, the particle information is written to a restart file. The particle information and surface element information is then read from the restart file to begin the "SurfS" algorithm. During the "SurfS" loop only the surface source particles, usually neutrals, are moved. The surface source particles are injected from wall elements due to the stored ion flux as well as from actual sources. The surface source particles then undergo collisions. Note that the results of collisions only affect the surface source species. As an example consider a CEX collision between an ion with velocity v_i and a neutral with velocity v_n. Normally, after the collision the ion has the velocity v_n and the neutral has the velocity v_i. However the "SurfS" algorithm changes this. After the collision the neutral will still be given the velocity v_i, but the ion will not change. It retains the velocity v_i. Thus the ions are treated as a background that is unaffected by

collisions. Of course, collisions between neutral operate normally. After a user specified number of time steps, the code writes a restart file.

At present we estimate the number of time steps required to reach steady state for the "normal" and "SurfS" algorithms. This is done by extrapolating the convergence rate of a standard simulation along with estimates of the time for typical ions and neutrals to cross the simulation domain. In the future we shall automate this process so the code will determine the optimal times to switch from one algorithm to the other. It will also determine when the simulation has reached steady state.

Results and Discussion

A set of simulations were run to quantify the effect of the surface source simulation on the computation time. Fig. 5 shows a simulated vacuum chamber with a set of ion sources. Xe neutrals and singly charged ions are injected with drifting Maxwellian distributions from one of the sources. The neutrals had a mass flow rate of $1 \cdot 10^{-7}$ kg/s, a drift velocity of 200 m/s and a temperature of 700 K. The ions had a mass flow rate of $5 \cdot 10^{-7}$ kg/s, a drift velocity of 20000 m/s and a temperature of 100 eV.

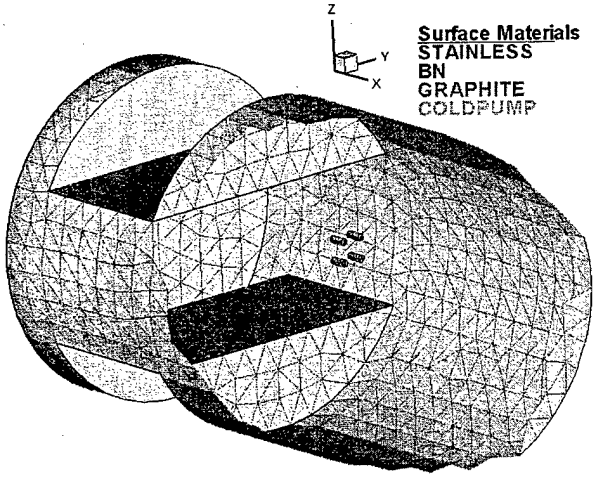

Fig. 5. Cutaway of the vacuum chamber geometry used in this simulation.

We use the particle number and particle energy as checks of the simulation accuracy. Fig. 6 is a plot of the particle number versus iteration. The black curves are the result for the standard simulation while the blue curves are the result for the surface source simulation. The neutrals slowly build up over time. At early time there is a sharp increase in the neutral particle number when the beam ions initially hit the chamber wall and begin to neutralize. The ion number rises in 20 iterations to approximately 30000. These are the beam ions. After 20 iterations the ion number increases slowly as the neutral density causes the creation of CEX ions.

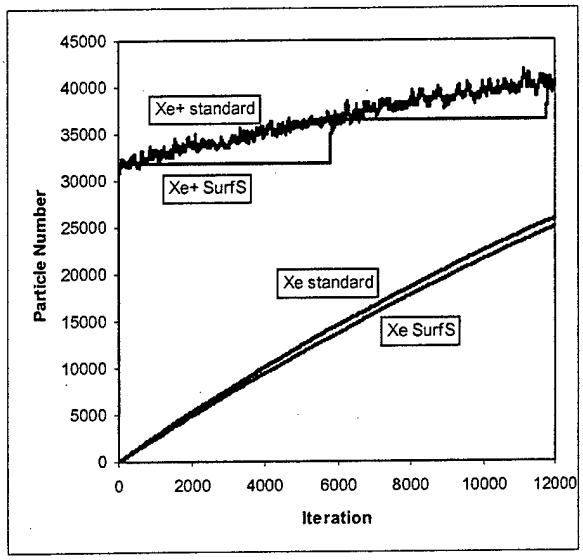

Fig. 6: Comparison of particle number for the different simulation types.

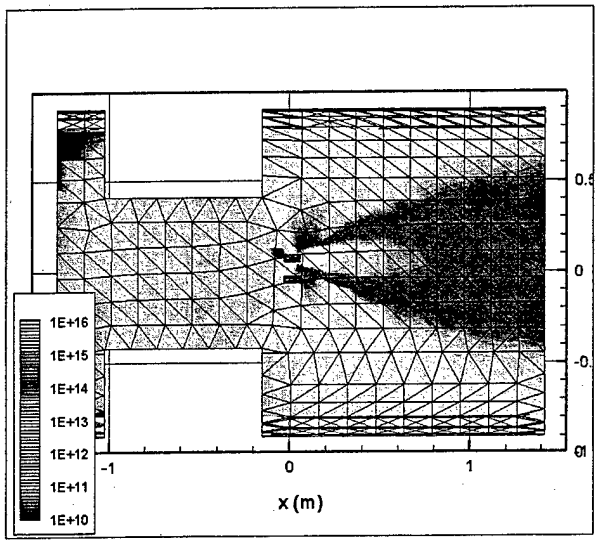

Fig. 7: Contour plot of the ion density in the vacuum chamber (#/m^3).

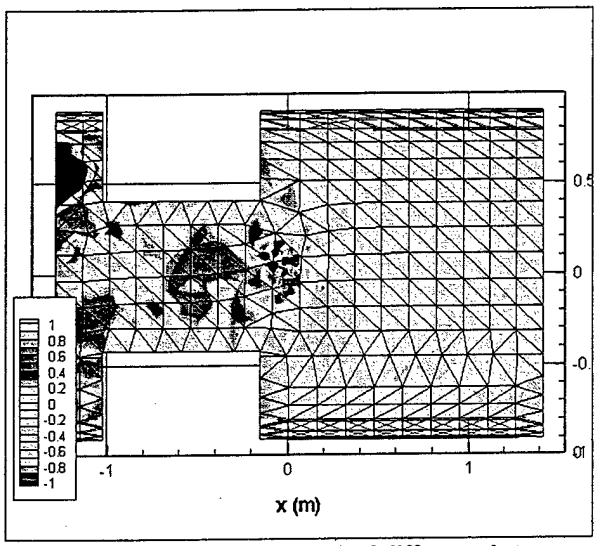

Fig. 8: Contour plot of the normalized difference between the ion densities of standard and the surface source simulations.

The ion density in the vacuum chamber is shown in Fig. 7. The highest densities are in the beam cone to the chamber wall and the CEX lobes perpendicular to the beam direction. We compare the results from the standard simulation and the surface source simulation in Fig. 8 by taking the difference and normalizing. In front of the source the difference between the simulation is less than 10%. Behind the source and near the pump inlets the variation becomes large. This is due to the high noise in the macro-particle count.

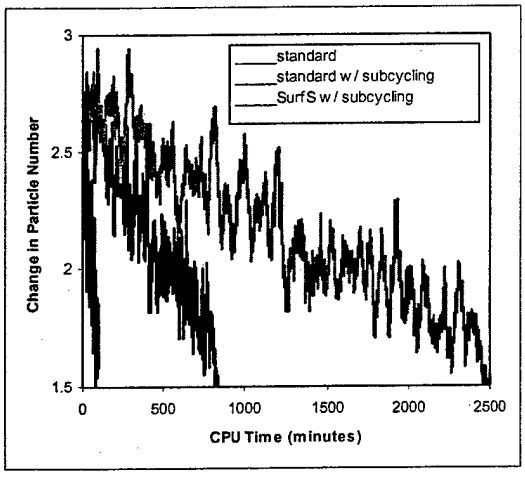

Fig. 9: Comparison of the change in particle number as a function of CPU time.

We track the change in particle number, ΔN, from one iteration to the next in order to check the convergence of the simulation to steady state. Once this quantity has an average of zero, we have reached a steady state. Fig. 9 is a moving average plot of ΔN with an averaging interval of 200 iterations. The moving average is used to suppress the noise in the data. The horizontal axis is the CPU time on a 3 GHz Intel Xeon. The standard simulation without subcycling is approximately 3 times slower than the standard simulation with subcycling. The standard simulation without subcycling is approximately 24 times slower than the surface source simulation with subcycling.

Conclusions

COLISEUM's modular architecture is allowing rapid expansion of its capabilities, and giving users flexibility to design their own geometries and choose their preferential plasma simulation method. This allows the user to combine modules to optimize the tradeoff between speed and accuracy. With the addition of the surface source algorithm it is now feasible to run vacuum chamber simulations including self-consistent treatment of the neutrals. This is important both for the determination of the CEX ion distribution and for calibration of simulation with experimental measurements such as pressure.

Additional work for the future includes introduction of new physics capabilities into the PIC-DSMC modules, determination of the most appropriate models for sources and wall interactions, implementation of more acceleration algorithms, and improvement of the capability to feed results from one module to another.

References

[1]Fife, J. M., et al., "3-D Computation of Surface Sputtering and Redeposition Due to Hall Thruster Plumes," 28th International Electric Propulsion Conference, 17-21 March 2003.

[2]Santi, M., S. Cheng, M. Celik, M. Martinez-Sanchez, and J. Peraire, "Further Development and Preliminary Results of the AQUILA Hall Thruster Plume Model," AIAA-2003-4873, 39th Joint Propulsion Conference, 2003.

[3]Wang, J., R. Kafafy, L. Brieda, "An IFE-PIC Simulation Model for Plume-Spacecraft Interactions," AIAA-2003-4874, 39th Joint Propulsion Conference, 2003.

[4]Gibbons, M.R., D.E. Kirtley, D.B. VanGilder and J.M. Fife, "Flexible Three-Dimensional Modeling of Electric Thrusters in Vacuum Chambers," AIAA-2003-4872, 39th Joint Propulsion Conference, 2003.

[54]Lord, R.G., "some extensions to the Cercignani-Lampis gas-surface scattering kernel," Phys. Fluids A 3 (4), 706-710 (1991).

[6]Cercignani, C. and M. Lampis, in "Rarefied Gas Dynamics," Academic Press, NY, p. 361 (1974).