JOURNAL OF COMPUTATIONAL PHYSICS **128,** 6–18 (1996)
ARTICLE NO. 0192

Three-Dimensional Plasma Particle-in-Cell Calculations of Ion Thruster Backflow Contamination

ROBIE I. SAMANTA ROY,*[,1] DANIEL E. HASTINGS,* AND STEVEN TAYLOR†

*Space Power and Propulsion Laboratory, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139; †Scalable Concurrent Programming Laboratory, Department of Computer Science, California Institute of Technology, Pasadena, California 91125

Received May 12, 1995; revised January 30, 1996

A fully three-dimensional hybrid plasma particle-in-cell model for multi-computer environments was developed to assess the spacecraft backflow contamination of an ion thruster. Results of plume backflow are presented for a 13-cm xenon ion thruster operating with a current level of 0.4 A on a model spacecraft. The computational domain was over 40 m^3 in volume, and used over 35 million particles representing charge-exchange (CEX) xenon ions produced in the plume. Results obtained on a massively parallel 256-node Cray T3D clearly show the plasma density enhancement around the spacecraft due to the CEX ions. Three-dimensional results are compared with the results of a two-dimensional axisymmetric model to explore the three-dimensionality of the backstreaming flow field.

I. INTRODUCTION

The potential problems of spacecraft contamination by the effluents of electric propulsion (EP) thrusters have been known for some time [1, 2]. However, ground-based experiments produce estimates of thruster contamination that are questionable due to vacuum chamber facility effects such as chamber wall sputtering and the presence of residual chamber gases. Only recently has massively parallel computing technology enabled large-scale fully three-dimensional computational assessments of thruster backflow contamination over realistic spacecraft geometries. The prediction of backflow contamination is of increasing importance now that EP is earnestly being considered for a variety of applications, including station keeping on commercial geostationary communications satellites [3]. Among the various types of EP thrusters, ion thrusters have reached a relatively high state of maturity and, thus, have received much attention in regards to spacecraft contamination and integration issues.

The evaluation of the thruster-induced environmental effects that could degrade the performance of spacecraft subsystems and sensors is very important. For example, in ion thruster plumes, a low-energy plasma is created by charge-exchange (CEX) processes and can expand around a spacecraft leading to a current drain on high-voltage surfaces. The enhanced plasma density due to a thruster plume can also lead to attenuation and refraction of electromagnetic wave transmission and reception. In addition, ion thrusters emit heavy metal species, both charged and uncharged, due to grid and discharge chamber erosion which can easily adhere to sensitive spacecraft surfaces and decrease the operational lifetime of the satellite. It is vitally important to understand and predict the backflow transport of these species from the plume unto a spacecraft.

We have developed a model of an ion thruster plume, based on the hybrid plasma particle-in-cell (PIC) technique, to simulate the backflow contamination from the CEX plasma that is created from CEX collisions between fast beam ions and thermal neutrals that leak out of the discharge chamber. Due to the large scale nature of the problem that involves spatial domains of meters to encompass a realistic spacecraft, and the large numbers of computational particles that are necessary for reasonable statistics, we have developed our model to take advantage of state-of-the-art massively parallel architectures. These computers enable the solution of problems that were previously inconceivable. The model is applied to estimate the backflow from a 13-cm xenon ion thruster located on a model spacecraft.

In Section II we formulate our approach to the problem and describe the physical model. Numerical methods are discussed in Section III, followed by a presentation of the concurrent implementation in Section IV. Results of a large-scale application are presented and discussed in Section V, and the concurrent performance of the model is discussed in Section VI. Last, conclusions and future work are offered in Section VII.

[1] Present address: Institute for Defense Analyses, Alexandria, VA 22311.

0021-9991/96 $18.00

II. PHYSICAL MODEL

The ion thruster plume model used in this study accounts for four of the five major thruster effluents: (1) fast (>10 km/s) propellant beam ions that provide the thrust, (2) unionized propellant neutrals with thermal energies that flow from both the discharge chamber and the neutralizer, (3) slow (initially thermal) propellant ions created predominantly from (CEX) collisions between the beam ions and neutrals, (4) non-propellant efflux (NPE) that consists mainly of eroded grid material, typically molybdenum, of which a fraction is charged due to either CEX or electron bombardment ionization, and (5) neutralizing electrons. The NPE efflux is not considered in this paper. In this section, brief descriptions of the other effluents are given based on the work of Samanta Roy *et al.* [4]. Some of the elements of the model are given in terms of axisymmetric coordinates for simplicity, but they are implemented as fully three-dimensional.

2.1. *Beam Ions*

The collimated beam ions of mass m_i that provide the thrust are accelerated electrostatically by a grid system to velocities typically on the order of 20,000–40,000 m/s. The beam ion velocity, $\mathbf{v}_{bi}$, is related to the beam voltage, Φ_b, by conservation of energy, $\mathbf{v}_{bi} = (2e\Phi_b/m_i)^{1/2}$. The current density of the beam ions, j_{bi}, is approximated by a parabolic axisymmetric profile given in cylindrical coordinates (r, x), where x is the axial coordinate,

$$j_{bi}(r,x) = \frac{2I_b}{\pi r_b^2}\left(1 - \frac{r^2}{r_b^2}\right) \tag{1}$$

which is subject to the normalization that at any downstream location in the beam,

$$I_b = \int_0^{2\pi}\int_0^{r_b} j_{bi} r\, dr\, d\theta, \tag{2}$$

where I_b is the ion current emitted from the thruster. The beam has a constant divergence angle, α, which is usually 15–20°, and thus the beam radius is $r_b = r_T + x\tan\alpha$, where r_T is the thruster radius. The beam current is assumed to be predominantly axial, with the beam velocity remaining approximately constant over the length scale of interest of several meters and, hence, the beam ion density, n_{bi}, is

$$n_{bi}(r,x) = \frac{j_{bi}(r,x)}{e\,\mathbf{v}_{bi}}. \tag{3}$$

The parabolic profile is in good agreement with experimental measurements on modern ion thrusters [5].

2.2. *Neutral Efflux*

Due to operating constraints, not all the propellant is ionized within the thruster. Unionized propellant that effuses out from the discharge chamber, exits in free-molecular flow. Only the flow on the macroscale is modelled and the effects of the multi-hole structure of the grids are neglected. A simple point source model gives the neutral density, n_n,

$$n_n(r,x) = \frac{n_{no}}{4}\frac{r_T^2(x + r_T)}{[(x + r_T)^2 + r^2]^{3/2}}. \tag{4}$$

The flux of neutrals is given by the Knudsen efflux, $n_{no}\overline{C}/4$, where the mean neutral speed $\overline{C} = \sqrt{8kT_w/\pi m_i}$ is based on the temperature of the thruster walls, T_w. The neutral density at the thruster exit, n_{no}, is given as a function of the beam current and the propellant utilization efficiency by the relation

$$n_{no} = \frac{4I_b}{e\overline{C}A_n}\left(\frac{1 - \eta_p}{\eta_p}\right), \tag{5}$$

where the propellant utilization efficiency, η_p, is based on the total mass flow rate (discharge + neutralizer),

$$\eta_p = \frac{I_b}{\dot{m}_{\text{total}}}\left(\frac{m_i}{e}\right). \tag{6}$$

The neutral flow through area, A_n, is close to the geometrical open area of the grids.

2.3. *CEX Propellant Ions*

Slow propellant ions are created inside the beam due to CEX reactions of the following type between the fast beam ions and the slow thermal neutrals,

$$\mathrm{Xe}^+_{\text{fast}} + \mathrm{Xe}^0_{\text{slow}} \rightarrow \mathrm{Xe}^+_{\text{slow}} + \mathrm{Xe}^0_{\text{fast}}.$$

The result is a fast neutral that travels in a line of sight manner and a slow ion that is affected by the local radial electric fields in the beam. The volumetric production rate of these CEX ions is given by

$$\dot{N}_{cex}(\mathbf{x}) = n_n(\mathbf{x}) n_{bi}(\mathbf{x}) \mathbf{v}_{bi} \sigma_{cex}(\mathbf{v}_{bi}), \tag{7}$$

where the relative collision velocity is taken to be the beam ion velocity. For a beam ion velocity of 33,200 m/s—the value used in the present simulation of the 13-cm xenon thruster—the CEX cross section [6], σ_{cex}, is 3.9×10^{-19} m^2. Volumetric production rates using Eq. (7) and the simple neutral model of Eq. (4) compare well with estimates based on measurements [4]. However, it must be

TABLE I

Gyroradii for Xenon Ions

	LEO ($B = 0.2G$)	GEO ($B = 0.001G$)
Thermal CEX ion ($T = 500$°K)	12 m	2 km
Beam ion $V > 10$ km/s	>680 m	>136 km

pointed out that there is a limit to the validity of this method of CEX ion creation. If the CEX ion production rate is too high, then the beam ion density cannot be taken to be a fixed quantity. If we define a CEX ion current, $I_{cex} = 4/3\pi r_T^3 e n_{no} n_{bio} \mathbf{v}_{bi} \sigma_{cex}(\mathbf{v}_{bi})$, then it must be that the ratio, $I_{cex}/I_b \sim r_T n_{no} \sigma_{cex} \ll 1$. In our results, typically this ratio is less than 3%.

An important consideration for the transport of the slow ions is the ambient and thruster-induced magnetic fields. Table I shows the gyroradii for thermal and beam ions in various magnetic field strengths corresponding to a range of orbital altitudes. The thermal speed of the CEX ions is the minimum speed and represents ions that have not been accelerated through the potential drop of the beam. For the length scales that we are interested in currently (<5 m), the ions can be considered unmagnetized. In addition, thruster-induced fields outside of modern ion thrusters are weak.

2.4. *Electrons*

Ion thrusters have a device called a neutralizer that emits an electron current to balance the ion current of the beam. Essentially, the neutralizer produces a quasi-neutral plasma cloud that acts as a "bridge" for the passage of electrons to the beam. In ion thruster beam plasmas, the thermal velocity of the electrons ($T_e = 1$–5 eV $\rightarrow 4 \times 10^5 - 9 \times 10^5$ m/s) is much higher than the ion beam velocity (2–4×10^4 m/s) and, thus, the electrons can diffuse rapidly to provide neutralization. It is beyond the scope of this work to include a detailed model of the physics of an electron emitting a hollow cathode, which is still not completely understood today. Our approach is to develop and use a simple model that treats the electrons as an isothermal neutralizing fluid with a drift velocity of that of the ions. Preliminary efforts have been made recently to model the electrons kinetically [7].

The general momentum balance for the electrons including electric (**E**) and magnetic fields (**B**), pressure forces, and collisional drag terms is

$$\frac{\partial \mathbf{v}_e}{\partial t} + \mathbf{v}_e \cdot \nabla \mathbf{v}_e = -\frac{e}{m_e}(\mathbf{E} + \mathbf{v}_e \times \mathbf{B}) - \frac{\nabla p_e}{m_e n_e} + \mathbf{R}_e. \quad (8)$$

A scaling analysis [4, 5] of the momentum equation leads to significant simplification. Since we are interested in the motion of the CEX ions, the time scales involved are much larger than electron time scales. Thus, the electron unsteady and inertia terms can be neglected.

The electron drift velocity is assumed to be on the order of the ion drift velocities. Physically, the electrons cannot overexpand the ions expanding in the plume, since a large charge imbalance would result. With this estimate for the electron drift velocity and taking typical beam potentials of 10 V or more, the ratio of the electric to magnetic forces is at least 10^2, even with magnetic field strengths of 3×10^{-5} T in a low earth orbit.

The collisional term consists of electron–ion and electron–neutral collisions. The electron–ion term is zero since it is proportional to the difference between the ion and electron drift velocities, and the electron–neutral term is small, compared to the electric field term and the pressure term. Hence, Eq. (8) can be simplified to a balance between the pressure, $p_e = n_e k T_e$, and the electric potential, ϕ, gradient,

$$e\nabla\phi = \frac{k\nabla n_e T_e}{n_e}. \quad (9)$$

While experiments show that the electron temperature decreases radially and axially due to cooling as the plume expands, the variation is much less than that of the plasma density since the electron thermal conductivity is very high [4, 5]. In this work, we consider the case where the electron temperature is constant, and Eq. (9) upon integration becomes

$$n_e = n_{e\infty} \exp\left(\frac{e\phi}{kT_e}\right). \quad (10)$$

Note that with this model, the electron density is a specified background density when the potential reaches zero, or the reference space potential far from the beam. The isothermal Boltzmann relationship, often referred to as the "barometric equation," has been studied and compared with a variable temperature model [5]. Comparisons show that the isothermal model yields slightly higher values of the backflowing CEX ion densities. However, comparisons of both models with experimental data show good agreement within experimental error [5]. The simplicity of the isothermal model outweighs the complexity of implementing the full fluid model in a multi-computer environment.

III. NUMERICAL MODEL

To model the expansion of an ion thruster plume, we employ the hybrid electrostatic plasma (PIC) method [8].

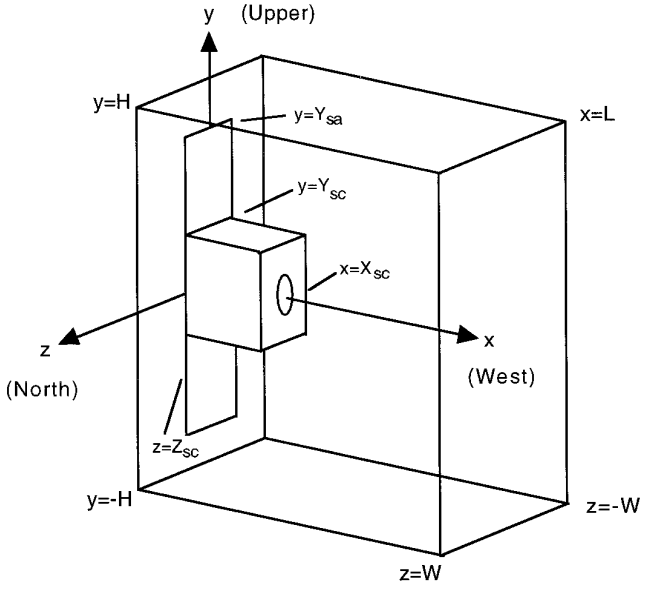

FIG. 1. Three-dimensional domain geometry.

In the electrostatic PIC technique, ions and electrons in a plasma are treated as macro-particles, where each macro-particle represents many actual particles. The charge of the simulation particles is deposited onto a computational grid and a charge density is computed. From this density, Poisson's equation for the electrostatic potential is solved, and the particles are moved under the influence of this self-consistent electric field. A major shortcoming of explicit fully kinetic PIC codes where electrons are treated as particles, is the very small time step that is required to resolve the electron motion. Since we are interested in the ion motion, we adopt the hybrid approach where the ions are treated as particles, but the electrons are treated as a fluid. In this manner, the time step is now on the ion time scale, which for xenon ions, is about 490 times larger than the electron time scale.

The model is fully three-dimensional. Figure 1 is a schematic of the general computational domain, and Fig. 2 shows the Cartesian grid in the $x - y$ plane for $y > 0$. The grid is non-uniform to more efficiently handle the highly non-uniform density distribution in the plume. Since the grid cell size should be on the order of the Debye length which scales with plasma density, n, as $1/n^{1/2}$, the grids are linearly stretched in the x, y, and z directions from the thruster exit to follow the increase in Debye length due to the density decrease which is assumed to decay as $1/R^2$ from the exit.

The slow CEX ions are treated as particles, with the real to macro-particle ratio around 10–100 million. Particles are created each time step in each grid cell based on the volumetric CEX production rate given by Eq. (7). The velocities are those of a Maxwellian distribution with a temperature corresponding to the wall of the discharge chamber (usually around 500°K). Particles that reach the simulation boundaries and spacecraft surfaces are removed, and steady-state is reached when the loss of particles at the boundaries balances the production rate in the beam. The bulk of CEX ions are produced within 2–3 beam radii downstream.

A standard linear Cartesian weighting function [8] is used to weight the charge of the particles to the grid. Within a cell, the charge is weighted to the eight neighboring grid points, and the volumetric charge density is computed by summing over all the particles. Once the charge density is computed, the electrostatic potential is determined by solving Poisson's equation,

$$\nabla^2\phi = \frac{e}{\varepsilon_0}\left(n_e - \sum_{\text{species}} n_i\right), \tag{11}$$

where n_e is given by Eq. (10). Note that the summation over the ion species allows multiple species: the beam ions, CEX ions, and ambient ions. With the Boltzmann distribution for the electron density, the Poisson equation for the electric potential becomes highly nonlinear. This equation is solved with a Newton–Raphson successive-over-relaxation (SOR) scheme. For large meshes, grid relaxation techniques are the methods of choice [9]. Either fixed po-

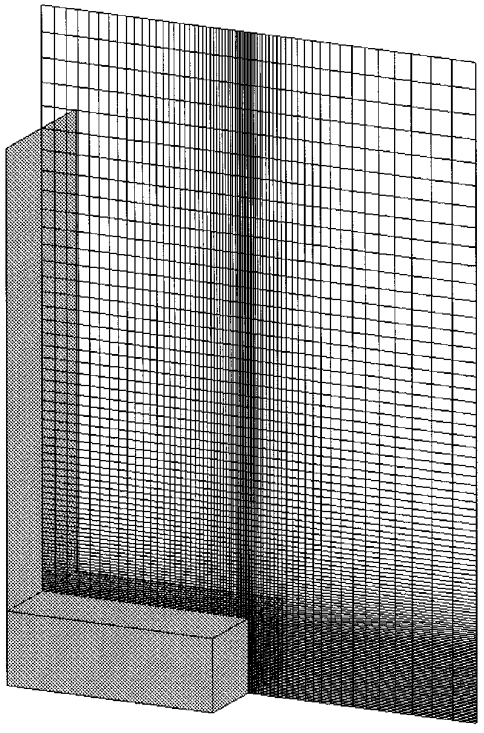

FIG. 2. Computational grid in x-y plane.

tentials are imposed on the spacecraft surfaces, or the spacecraft potential can float with respect to the ambient plasma, and Neumann boundary conditions are held on all exterior domain boundaries. A solar array panel is attached to the spacecraft, and a potential distribution along its length is specified.

Given the potential, the electric field, **E**, is computed from $\mathbf{E} = -\nabla\phi$, and the equations of motion of each ion macro-particle are integrated,

$$\frac{d\mathbf{v}_i}{dt} = \left(\frac{q}{m}\right)_i [\mathbf{E} + \mathbf{v}_i \times \mathbf{B}], \quad \frac{d\mathbf{x}_i}{dt} = \mathbf{v}_i, \tag{12}$$

where q/m is the charge to mass ratio of the ion. The capability to include a fixed magnetic field, **B** (i.e., the geomagnetic field), is incorporated in our model. However, for these calculations, the role of the magnetic field is neglected.

IV. PARALLEL IMPLEMENTATION

4.1. *Partitioning*

The approach to parallel implementation is similar to the first general concurrent PIC algorithm developed by Liewer and Decyk [10]. Data decomposition is used for the partitioning of the problem. The computational domain is partitioned into blocks according to a simple rule that each block face must be of a single type, i.e., a face is completely a spacecraft surface or an interior interblock face—it should not be a mixture in order to maintain simplicity in programming. The methodology of dividing a problem into blocks actually simplifies programming structure and enables very general geometries to be handled, because each block can be individually initialized with or without particles, and with various boundary conditions.

Figure 3 shows a coarse decomposition of the domain in the x-y plane. The domain is partitioned into a minimum of two blocks in the x-direction: one behind the thruster plane and another in front. There are seven in the y-direction for a full plane simulation and four for a half-plane simulation in order to accommodate thruster, spacecraft, solar array, and exterior boundary surfaces. Last there are five in the z-direction. Thus, for a half-plane simulation, there are a minimum of 40 blocks in the primary decomposition. Each block can then be decomposed into smaller blocks by a secondary decomposition, so that, depending on the total domain size, there are two to three blocks per available processor. The blocks are numbered as $N = i + jN_x + kN_xN_y$, where N_x and N_y are the number of blocks in the x and y directions and i, j, k are the block indices which start from (0, 0, 0) in the lower southeast corner of the domain.

The faces of each block must be categorized for computing and communications purposes. It is important to know what types of boundary conditions to enforce for the Poisson solver and how to treat particles that cross the boundaries of each block. The face types are identified in the following manner: (a) hole (interior of the spacecraft—no computation or communication); (b) interior interblock cut face; (c) exterior boundary—Dirichlet or Neumann potential boundary condition; (d) spacecraft surface—potential specified; (e) thruster front—potential specified; (f) top solar array—potential specified; (g) bottom solar array—potential specified; and (h) reflecting boundary condition on $y = 0$ plane for upper half-plane calculations (Neumann condition on potential).

The reflecting boundary condition is used for the particles, i.e., a particle that hits this surface is reflected—just as on the plume centerline in an axisymmetric model. Physically, for every particle that leaves the upper half-plane, there is another one that is entering from the lower half-plane. On all other exterior and spacecraft boundaries, particles are absorbed. Particles are passed between processors if they share an interior interblock cut face.

The way that the global computational grid is divided at the boundary between two blocks is illustrated in Fig. 4 (which is the same for every axis). Particles in Block N whose position is greater than or equal to x(IXB) are passed to Block $N + 1$. Similarly, particles that are less than $x(1)$ in Block $N + 1$ are passed to Block N.

4.2. *Parallel PIC Algorithm*

A preprocessing geometry code generates the global computational domain for a given problem and interactively decomposes the domain into N_b blocks. Generally, if M is the number of processors available, $N_b \approx (2\text{–}3)M$ to allow for overlapping of communication and computation. An input file for each block is created that consists of the local grid, boundary conditions, and face types. The N_b input files are mapped onto the M processors, and the same PIC code operates on each block. Each block executes the algorithm shown in Fig. 5.

Each block in the entire domain is solved independently and appropriate boundary conditions are used to signify what should happen at the interface between blocks. At block interfaces, there is a non-physics boundary condition representing a ''cut'' in the domain. This boundary condition represents the fact that communication must occur for both particle and grid quantities. Note that the algorithm basically has two parts. Within each block where the CEX ion production rate is non-zero, particles are created and are moved. If a particle exits a block, it is communicated to an appropriate neighboring block either via a face, edge, or corner. After all particle movement is conducted, the electric field is solved self-consistently in the inner loop of the algorithm. This requires only face exchanges between

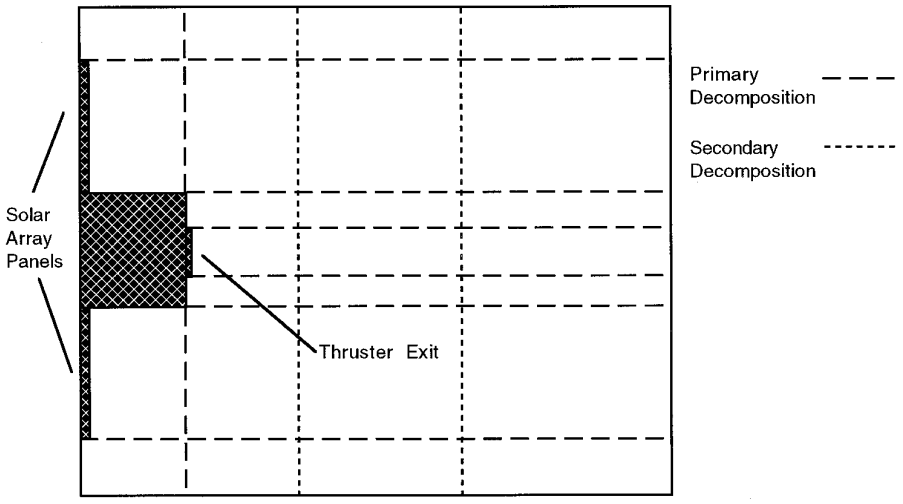

FIG. 3. Example of domain decomposition in x-y plane.

adjacent blocks. After an SOR sweep in all the blocks, the potentials must be passed between blocks that have common interior interblock faces. In essence, continuity of the potential must be enforced. The potential at grid points IXB in Block N are passed to grid points 1 in Block $N + 1$, while grid points 2 in Block $N + 1$ are passed to grid points IXB + 1 in Block N. Thus, each block solves the potential from points 2 to IXB, holding points 1 and IXB + 1 fixed for each iteration. The process of an SOR sweep followed by boundary cell exchange is continued until global convergence is met.

In addition to the particles being passed between blocks, the densities at the grid points on the boundaries between neighboring blocks must be superimposed to get the right values. For instance, referring to Fig. 4, a particle in Block N between x(IXB $-$ 1) and x(IXB) is weighted to those two points. However, Block $N + 1$ must know about the charge of that particle in order to have the proper density at its grid point $x(1)$. Thus these boundary density values must be added up between blocks.

The biggest challenge to parallel computing for simulations where the computational load is dynamically changing is the load-balancing of all the processors. In our simulations, the CEX ions are created within the beam and expand outwards to surround the spacecraft. Initially, all the particles are within the beam, and processors that have

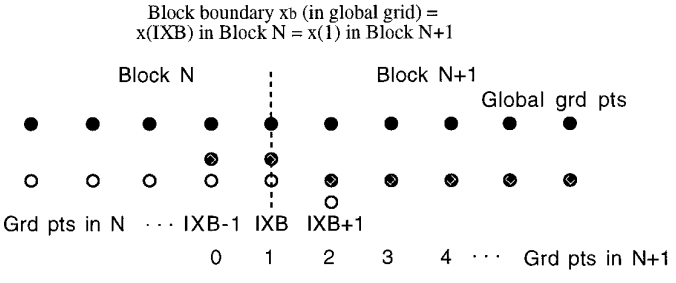

FIG. 4. Detail of splitting computational grid between two neighboring blocks.

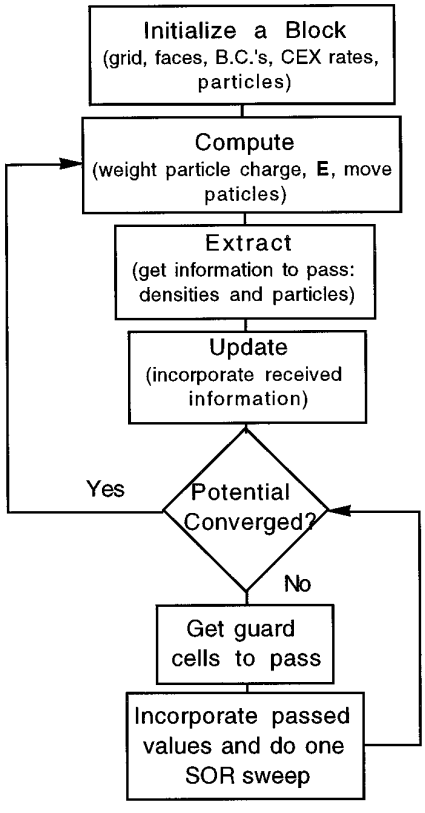

FIG. 5. Flowchart of concurrent algorithm.

blocks in that region are doing all the work. Processors outside of the beam do not have any particles and are only solving the potential field in their respective blocks. Thus, initially, there is a severe load imbalance on the machine. However, as the simulation progresses, the particles travel into other processors and the load becomes more distributed. To alleviate this problem somewhat, the blocks that are within the beam (where the particle densities are the highest) are made smaller than blocks outside, so that more processors have portions of the beam. It may be noted that blocks outside are larger and, hence, have more grid points for the potential solver. Thus, processors outside of the beam may not have any particles, but they will be busy solving the field. However, the field must be solved before the particle operations, so that there is no gain in this respect.

The most efficient approach is to balance the load dynamically while the simulation is in progress. There are two main approaches: one is to change dynamically the size of the blocks (and, hence, the number of particles within a block), and another is to move the blocks that have the most particles to processors that have the least work. Ferraro *et al.* [11] have applied the former technique to a two-dimensional PIC simulation. By load balancing dynamically, there is always the trade-off between gains in computational efficiency and the cost associated with the repartitioning. The results of the three-dimensional calculations in this paper do not incorporate any form of dynamic load balancing. A static decomposition was used, based on sizing the blocks, so that those in the beam were about half the size of those outside. Large-scale three-dimensional plasma simulations on massively parallel computers have been performed (without any dynamic load balancing) before on unbounded plasmas [12] and, recently, on spacecraft charging [13]. The results in this paper are the first of their kind in three-dimensional ion thruster plume contamination simulations, but they are quite elementary in terms of the complexity of the computer science issues. There is much room for increases in parallel efficiency, and it is one area of future improvement beyond this work.

V. RESULTS

5.1. *Three-Dimensional Plume Structure*

In this section, we present and discuss the three-dimensional simulation results. The main goal of the three-dimensional model in this paper is to investigate the CEX ion backflow structure and to see whether geometrical effects due to a fully three-dimensional spacecraft are important. We will also compare the three-dimensional results with results from an axisymmetric model [4]. In the case considered in this paper, it is shown that the two-dimensional model gives an upper bound on the CEX plasma density, and hence, the two-dimensional model can be used for estimates in situations that are not purely axisymmetric. This is highly desirable since the two-dimensional model offers vast computational savings over the three-dimensional model.

We examine the backflow from an ion thruster on a large-scale model spacecraft. Following the geometry of Fig. 1, the size of the model spacecraft in the simulation is 1.5 m long in the x-direction. This is only the distance from the solar array panels to the thruster exit; the entire spacecraft is not modelled. The model spacecraft has a half-height of 0.5 m in the y-direction, and a width of 1 m in the z-direction. A solar array panel extends 3.1 m from the top of the spacecraft 1.5 m behind the thruster exit and is biased. The potential drop from the spacecraft to the end of the array is 28 V, typical of most current systems, with the spacecraft grounded to the positive end. The simulation domain is a half-domain in the y-direction. The ion thruster simulated is a 13-cm beam diameter xenon thruster, and the operating conditions are a beam current of 0.404 A, a propellant utilization efficiency of 0.84, and a beam ion velocity of 33,200 m/s. A constant electron temperature of 1 eV is used, and no geomagnetic field was included. To encompass the spacecraft and the solar array panel, the dimensions of the computational domain are 3.2 m $\times$ 4.5 m $\times$ 3 m in the x, y, and z directions, respectively. The computational grid has 139 grid points in the x-direction, 241 in the y-direction, and 281 in the z-direction—a total of 9,413,219 grid points.

The problem was run initially on the massively parallel Intel Touchstone Delta at Caltech and, subsequently, on the Cray T3D at the Jet Propulsion Laboratory. Inaugurated in 1991, the Delta consists of 512 processors each with 16 MB of memory and a peak speed of 80 Mflops. However, the amount of user available memory per node is closer to 12 MB, bringing the total machine capacity to over 6 GB. The processors are connected via a scalable two-dimensional mesh. The Cray T3D, installed in 1994, consists of 256 nodes, each with about 55 MB of usable memory—a total of 14 GB. The speed of each processor is about 150 Mflops peak, and they are connected via a three-dimensional torus topology which offers much better interprocessor communications. Message latency (time to send a message) on the Delta is about 150 μs and the bandwidth (rate of information being transmitted) is about 10 MB/s. In contrast, the latency is around 10 μs, and the bandwidth is 120 MB/s on the T3D.

Since the problem was initially targeted for the Delta, the domain was partitioned into 1575 blocks so that there would be three or more blocks per processor. The same partitioning was kept for the T3D, hence six or more blocks were allocated to each processor. It was found that the simulation code performed about six times faster on the T3D due to its more powerful processors and faster com-

munications network. The simulation was run until steady state was reached with the particle population reaching over 35 million. Due to the lack of dynamic load balancing at the start of the simulation, only about 10% of the processors are utilized for particle operations. However, as the particles expand from the beam region, the number of processors utilized increases. The ratio of the time spent in interprocessor communications to computation, is about 1.4%, and the overall speed of the code is about 30 s/timestep. Due to the high demand of the machines and the availability of only four-hour slots on weekdays, the simulation took nearly four months to complete. The calculation required a total of approximately 128,000 node-hours.

Figure 6 displays a picture of the spacecraft (gray surfaces) and two planes in the *x-y* (vertical) and *x-z* (horizontal) displaying the potential in the plume. A potential isosurface is also rendered. Since the isothermal Boltzmann model is used for the electrons, isopotential surfaces are also isodensity surfaces, and we can clearly see the expansion of the CEX plasma around the model spacecraft. The plume structure can be seen more clearly in Fig. 7 which is a contour plot of the total ion density in the *x-y* plane cut directly through the center of the plume ($z = 0$). The plasma density decays from 10^{14} m^{-3} in the beam at the thruster exit, to below 2×10^{10} m^{-3}. The background plasma density is fixed at 10^{10} m^{-3}, typical of the LEO environment. In the region directly above the spacecraft, the CEX ions expand over the top of the spacecraft, and the density there is between 10^{10}–2×10^{10} m^{-3}. The CEX ions propagate out of the beam and around the spacecraft due to the electric field which acts normal to the potential contours shown in Fig. 8.

The geometrical effect of the rectangular spacecraft on the backflow is apparent though, when we examine the *z-y* plane that is perpendicular to the plume axis. Figure 9 is a *z-y* plane that is 3 cm behind the thruster exit plane. In Fig. 9, we can clearly see that the CEX ions flowing back around the spacecraft do not do so in a completely axisymmetrical fashion due to the rectangular geometry of the spacecraft. The CEX ions are essentially expanding around a plate that is the spacecraft face on which the thruster is located, and the backflow is concentrated on the top and both sides. However, at the corners (45° to the *y* axis), the density is much less.

A clearer presentation of the asymmetry in the *z-y* plane is shown in Fig. 10, where the total ion density along an arc 0.75 m from the center of the plume is shown for three *z-y* planes located 3, 24, and 95 cm upstream of the thruster exit plane. In the plane 3 cm upstream, the density falls by almost an order of magnitude at 45° and 135°—angles corresponding to the corners of the spacecraft. However, at planes further upstream, the density becomes more axisymmetric, until 95 cm upstream, there is no azimuthal distinction. At distances further from the plume center though, the structure of the CEX density still retains a distinctive asymmetric nature due to the rectangular spacecraft. Thus, this calculation demonstrates that the three-dimensional geometry of the spacecraft influences the structure of the plume backflow.

5.2. *Comparison of Two- and Three-Dimensional Results*

To see how substantial the three-dimensional effects are in comparison, an axisymmetric model [4] was applied to the same spacecraft in the *x-y* plane, including the solar array panel. The same number of grid points (139×241) and the same thruster operating conditions were used. Various numbers of particles were used ranging from 60,000 to 300,000. The appeal of the axisymmetrical model is that low noise results could be obtained in two dimensions with less than 200,000 particles. The two dimensional simulation could run 24 h/day on a workstation and, depending on the number of particles, the overall performance was between 4–20 s/timestep. Thus, the axisymmetric code was much more cost effective in terms of producing results in a relatively short amount of time (<1–3 days).

Figure 11 compares the total ion density along a radial cut (90° to the plume centerline) 22 cm downstream of the thruster in the *x-y* plane from both the three-dimensional and axisymmetric simulations. We can see that the comparison is very good. The plume radial expansion is axisymmetric as we expect. It is interesting to see what the decay rate in the density is, due to the plume expansion. Simple backflow models [5] treat the CEX plasma with a constant velocity spherical expansion, and the density decays as $1/R^2$. From the density profile in Fig. 11, we see that the density decays roughly with a rate of $1/R^{2.4}$, which is quite close. However, a difference is to be expected since the CEX ions are not expanding at constant velocity, but they are still slowly gaining speed as they fall down the potential hill from the plume. It must be kept in mind that, with the isothermal Boltzmann electron model, the plasma density and potential follow each other, and hence, the potential is not completely flat outside of the beam.

It is important to see how the CEX plasma behaves, not only radially to the beam, but also along rays at angles greater than 90° that penetrate into the backflow region. In Fig. 12, the total ion density along a ray 120° from the plume centerline and from a point 0.75 m in front of the thruster exit is shown. Again, we see that the agreement between the two models is very good, with the three-dimensional results falling slightly below the axisymmetric results at distances greater than 2 m. This may possibly be due to an insufficient number of particles in the three-dimensional model, but nevertheless, the axisymmetric model yields larger densities. We note that the density decay in the backflow region

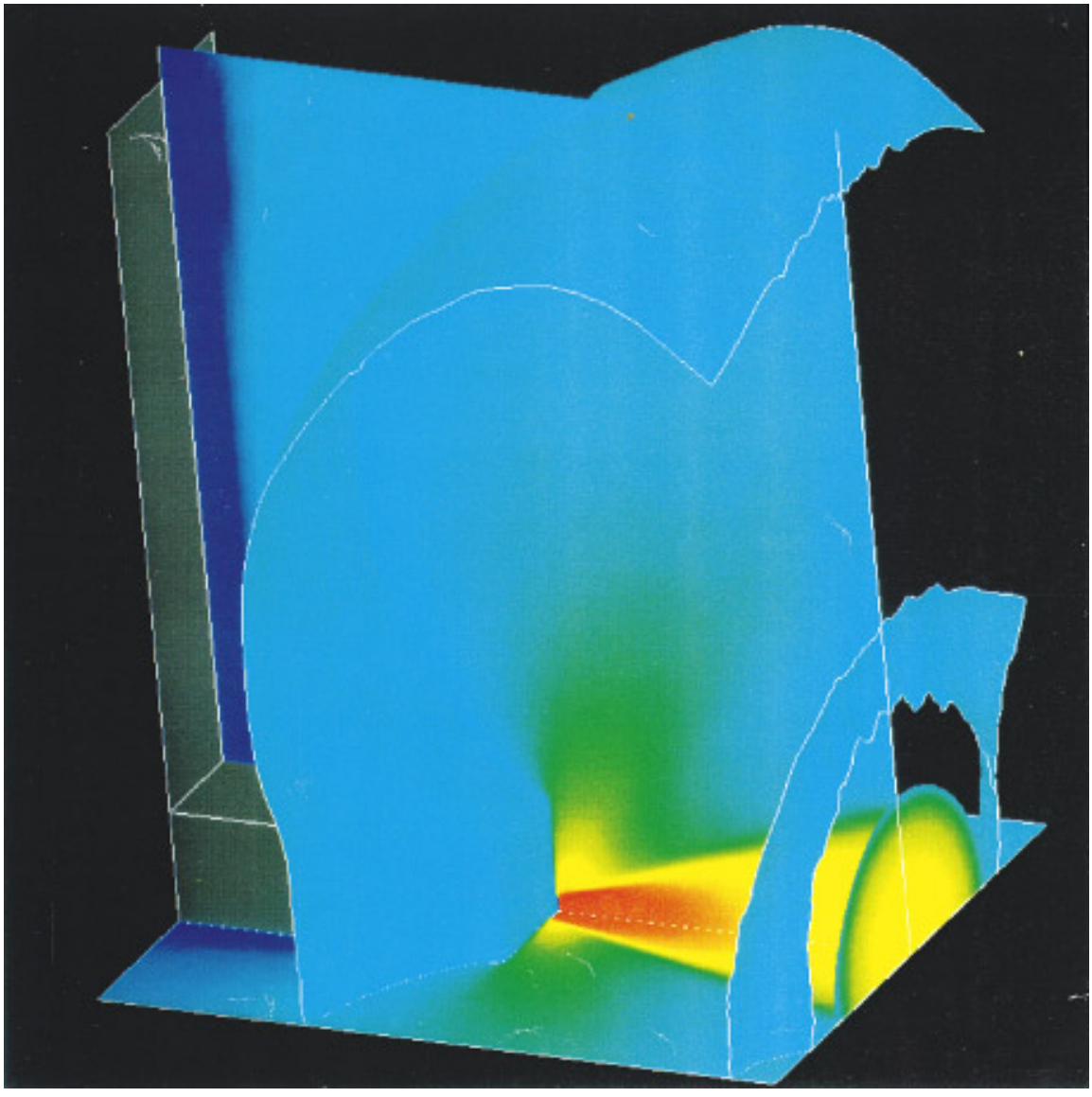

FIG. 6. Three-dimensional plot of potential field surrounding spacecraft.

along this ray is also about $1/R^{2.23}$, very similar to the behavior along a ray perpendicular to the plume axis.

In addition to radial comparisons, we compare the models along an arc of constant radius. In Fig. 13, the total ion density along an arc a distance of 1.5 m from a point 5 cm in front of the thruster is shown. In the backflow region at angles greater than 90°, the axisymmetric results are higher than the three-dimensional results again. Thus, we see that the axisymmetric model gives an upper bound on the CEX plasma density in the backflow regions.

From the comparisons of this relatively simple geometry case, we conclude (at least for asymmetric cases similar to one explored here) that the axisymmetric model can be used to provide an upper bound on the CEX backflow in situations that are not too strongly asymmetric. However, for more complex geometries, the fully three-dimensional model would have to be used. There are many computer science issues that need to be addressed such as dynamic load balancing that will improve the performance of the code. Nevertheless, routine three-dimensional calculations of this nature may have to wait for the next generation of massively parallel computers.

VI. PARALLEL STUDY

Figure 14 provides a baseline performance evaluation of the simulation and is taken from an associated load-balancing study [14]. The horizontal axis shows a timeline

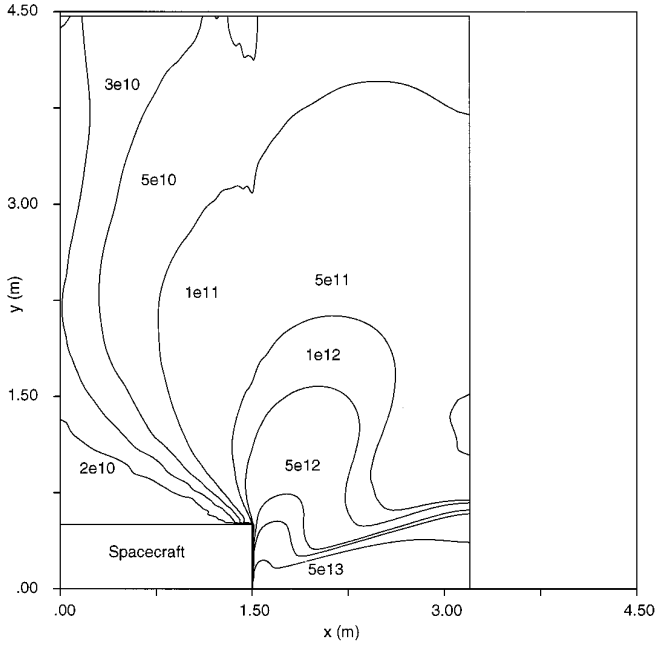

FIG. 7. Total ion density in *x-y* plane.

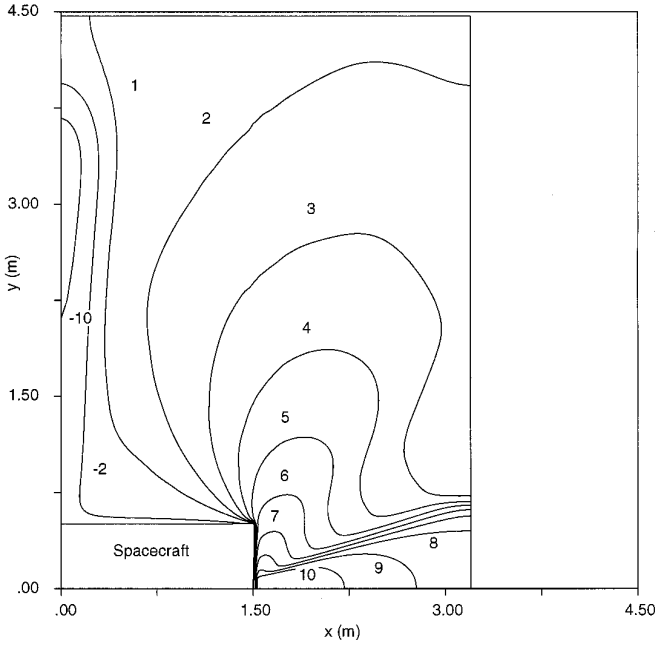

FIG. 8. Potential in *x-y* plane.

of the number of iterations used to generate the solution, while the vertical scale shows the time taken to execute 100 iterations. The time is broken down into three categories representing computation, communication, and idle time. The communication time represents only the time to place and extract messages into and out of the network. Idle time waiting for communication has been removed by virtue of multiprocessing multiple blocks within each computer. On the Intel Delta machine the calculation was primarily communication bound. In moving to the T3D, that communication time was reduced to insignificance.

The results indicate that using a carefully hand-coded decomposition, the aggregate utilization varies during the course of the calculation between 42% and 58%. The computation time per timestep is, in essence, a function of the time to solve individual steps of the field and a cost proportional to the number of particles that are moved. Although the field calculation can be predicted a priori, the particle distribution varies dynamically. Initially, the field solver dominates the computational cost and is reasonably balanced, even though the particle distribution is relatively localized. As particles begin to dominate, the utilization drops. As a result, to improve this overall distribution, a dynamic load-balancing mechanism is required. The code includes a scalable load-balancing technique based on diffusion [15] and a study of its effects on performance are the subject of ongoing research [14].

The area of the graph indicated by dividers represents changes to the basic constants in the simulation to improve accuracy. Typically these represent an increase in particle production to gain more accurate solutions. Between the final two measurements the total number of particles doubled. Notice that the overall utilization remained constant, indicating that the principal blocks in the decomposition-controlling performance eventually reach saturation in terms of the number of particles, as one would expect in

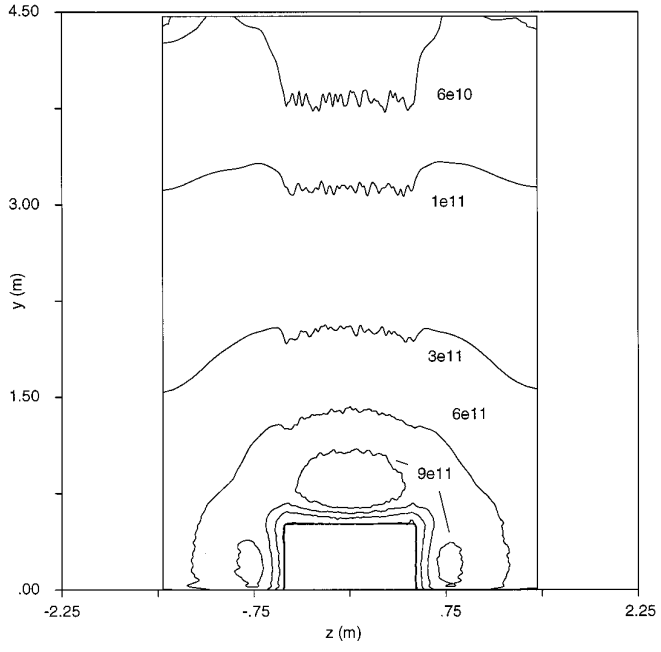

FIG. 9. Total ion density in *z-y* plane.

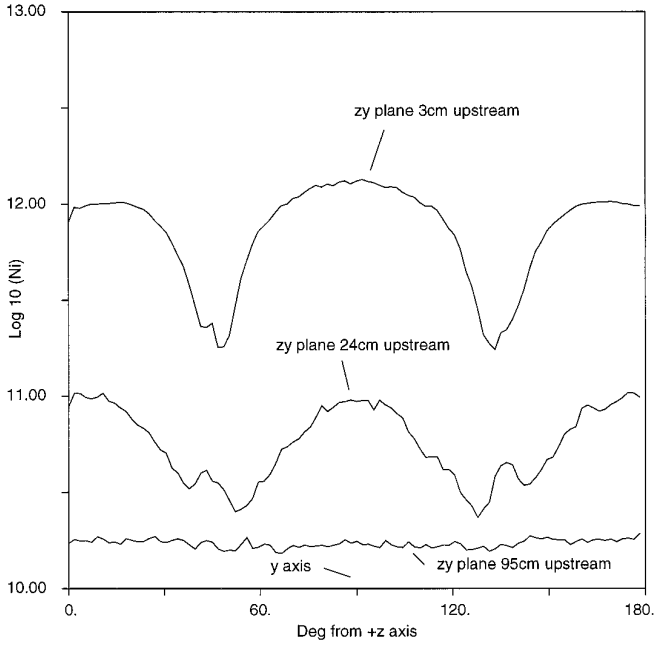

FIG. 10. Comparison of density in z-y plane along arc for distances 3, 24, 95 cm upstream of thruster exit.

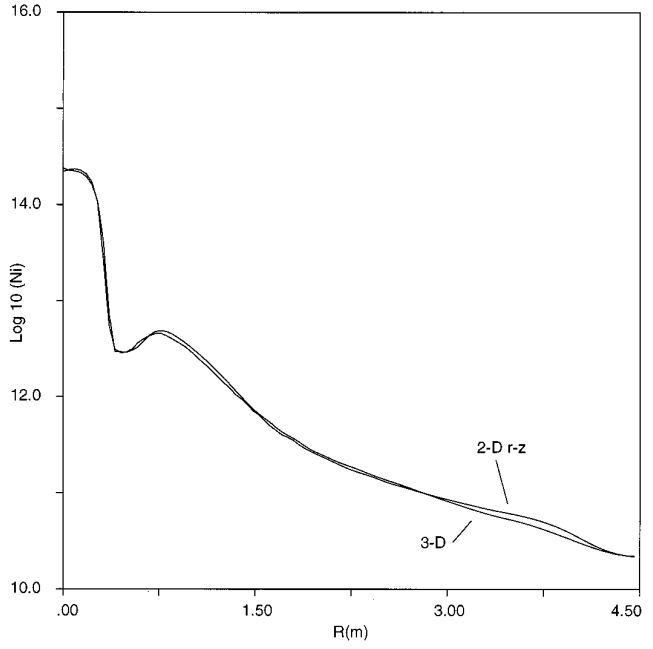

FIG. 12. Comparison of 3D and 2D ion density along cut 120° from plume axis.

the steady state. Thus, this study provides an accurate global picture of the overall performance of the algorithms; the utilization would not reduce further were the computations to be continued.

VII. CONCLUSIONS

In order to investigate the geometrical effects on the backflow structure due to the three-dimensional space-

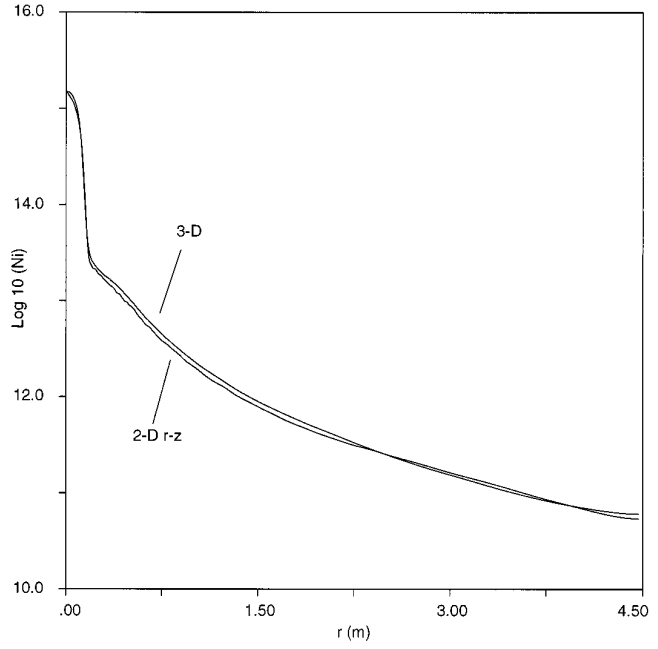

FIG. 11. Comparison of 3D and 2D ion density along radial cut 22 cm downstream.

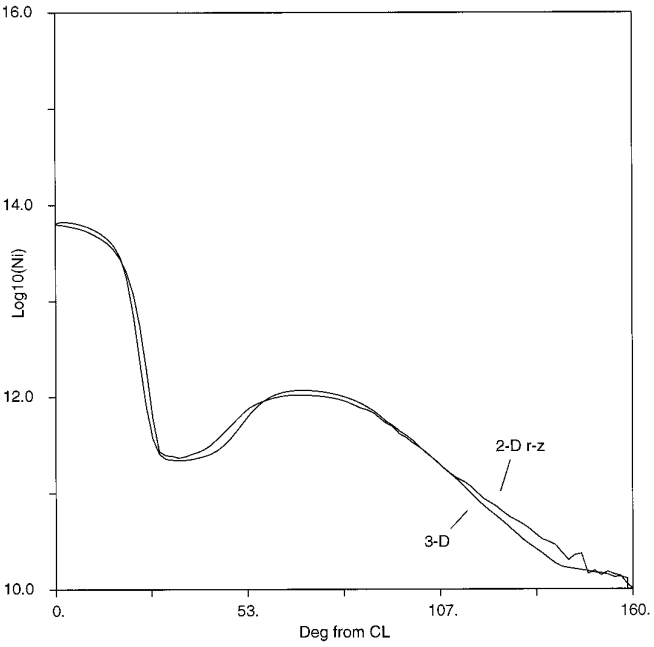

FIG. 13. Comparison of 3D and 2D ion density along arc 1.5 m from point 5 cm downstream.

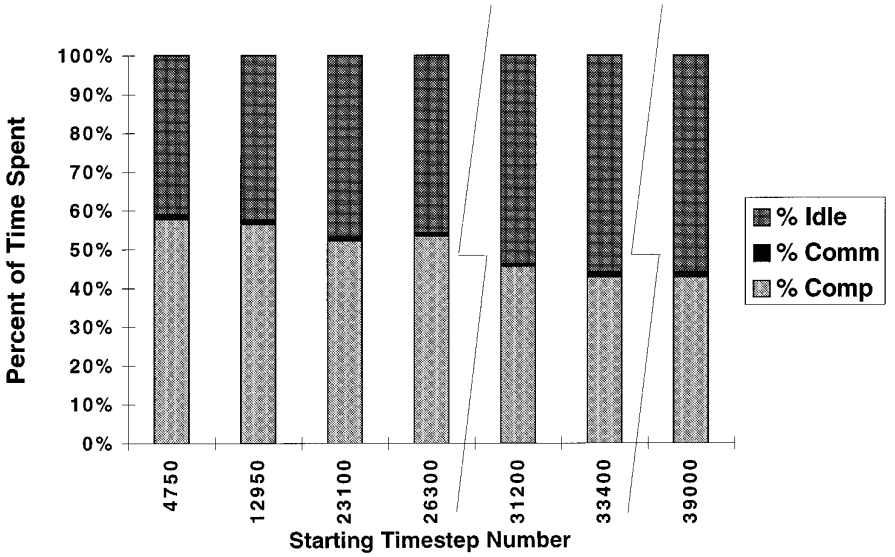

FIG. 14. PIC code runtime breakdown.

craft, a fully three-dimensional numerical model was developed. Due to the extremely large computational resources required for the large spatial domains involved encompassing a realistic spacecraft, the use of massively parallel computers was necessary and enabling. A PIC algorithm for a message-passing multi-computer environment was developed and implemented on two massively parallel computers that had sufficient memory to handle up to 3 GB memory requirements.

The propellant CEX ion backflow from a 13-cm xenon ion thruster was computed on a large-scale model spacecraft.The computational domain contained over 9.4 million grid points, and the simulation employed over 35 million particles. This is the first calculation ever conducted of ion thruster plume backflow on such a large scale. This code employs a variety of realistic spacecraft characteristics, including floating potentials on spacecraft surfaces, potential drops across solar array panels, and self-consistent electric fields. In addition, the computational model created can be used as a testbed tool with which many computer science issues such as dynamic load balancing can be explored.

The plume backflow was examined for three-dimensional effects since the spacecraft's geometry could not be captured with an axisymmetric model. Since the spacecraft was a box shape, a ''corner-effect,'' where the CEX plasma flowed over the front face of the spacecraft was identified. This led to an asymmetry in the backflow around the spacecraft, that could produce significant decreases in CEX ion density up to an order of magnitude, particularly, close to the thruster exit plane.

The three-dimensional results were compared with an axisymmetric model applied in a plane through the plume. Comparisons of CEX ion densities along radial and angular cuts throughout the backflow region away from the ''corner-effect'' regions did not show significant differences. It was shown that the axisymmetric model can be used to give a conservative upper bound on the backflow, at least for geometries that are not highly asymmetric, as was the case.

Future work will address the use of unstructured grids for arbitrary geometries and thruster positions. Multiple thrusters will also be examined, and the effect of the CEX plasma cloud discharging charged spacecraft surfaces will be investigated.

ACKNOWLEDGMENTS

The first author was supported in part under a National Science Foundation Graduate Fellowship. Any opinions, findings, conclusions, or recommendations expressed in this publication are those of the author and do not necessarily reflect the views of the National Science Foundation. Support for the first two authors was also provided by the U.S. Air Force Office of Scientific Research under Contract F49620-93-1-0317. The third author was supported by the Advanced Research Projects Agency, ARPA Order 8176, monitored by the Office of Naval Research under Contract N00014-91-J-1986. The authors are most grateful to Jerrell Watts, a graduate student at Caltech, for conducting the T3D production runs. The performance summary provided here was produced by Jerrell as part of his ongoing work on load-balancing methods.

REFERENCES

1. D. C. Byers, *J. Spacecraft Rockets* **16**, 289 (1979).
2. W. D. Deininger, AIAA Paper 85-2046, 1985.
3. J. E. Pollard, D. E. Jackson, D. C. Marvin, A. B. Jenkin, and S. W. Janson, AIAA Paper 93-2221, 1993 (unpublished).

4. R. I. Samanta Roy, D. E. Hastings, and N. A. Gatsonis, *J. Spacecraft Rockets* **33,** in press (1996).
5. R. I. Samanta Roy, Ph.D. dissertation, Dept. of Aeronautics and Astronautics, MIT, 1995 (unpublished).
6. D. Rapp and W. E. Francis, *J. Chem. Phys*. **37**, 2631 (1962).
7. J. Wang, J. Brophy, P. Liewer, and G. Murphy, AIAA Paper 95-0596, 1995 (unpublished).
8. C. K. Birdsall and A. B. Langdon, *Plasma Physics via Computer Simulation* (Hilger, Bristol, 1991).
9. R. W. Hockney and J. W. Eastwood, *Computer Simulation Using Particles*, (Hilger, Bristol, 1988).
10. P. Liewer and V. Decyk, *J. Comput. Phys.* **85**, 302 (1989).
11. R. D. Ferraro, P. C. Liewer, and V. K. Decyk, *J. Comput. Phys.* **109**, 329 (1993).
12. J. Wang, P. Liewer, and V. Decyk, *Comput. Phys. Commun.* **87**, 35 (1995).
13. J. Wang, AIAA Paper 96-0147, 1996 (unpublished).
14. J. Watts and S. Taylor, California Institute of Technology, Dept. of Computer Science, Technical Report TR-95-04, 1995 (unpublished).
15. A. Heirich and S. Taylor, in *Proceedings, 24th Int. Conf. on Parallel Programming, 1995*, Vol. 3 (CRC Press, 1995), p. 192.