JOURNAL OF COMPUTATIONAL PHYSICS 122, 323–334 (1995)

# Combining the Monotonic Lagrangian Grid with a Direct Simulation Monte Carlo Model

BOHDAN Z. CYBYK

*Wright Laboratory, Wright-Patterson Air Force Base, Ohio 45433*

ELAINE S. ORAN AND JAY P. BORIS

*Naval Research Laboratory, Washington, DC*

AND

JOHN D. ANDERSON, JR.

*University of Maryland, College Park, Maryland*

Received August 11, 1994; revised April 10, 1995

Using the monotonic Lagrangian grid (MLG) as a data structure in the direct simulation Monte Carlo (DSMC) methodology produces an approach that automatically adjusts grid resolution to time-varying densities in the flow. The MLG algorithm is an algorithm for tracking and sorting moving particles, and it has a monotonic data structure for indexing and storing the physical attributes of the particles. The DSMC method is a direct particle simulation technique widely used in predicting rarefied flows of dilute gases. Monotonicity features of the MLG ensure that particles close in physical space are stored in adjacent array locations so that particle interactions may be restricted to a "template" of near neighbors. The MLG templates provide a time-varying grid network that automatically adapts to local number densities within the flowfield. Computational advantages and disadvantages of this new implementation are demonstrated by a series of test problems. 

## 1. INTRODUCTION

In a rarefied gas flow, the motions and interactions of individual particles are important. To numerically model such a flow, a microscopic particle-dynamics approach based on the Boltzmann equation can be used. In this approach, a gas is treated as a collection of particles whose positions and velocities are individually tracked. It is the only approach that is valid when the freestream Knudsen number Kn,

$$\mathrm{Kn} = \lambda/L, \tag{1}$$

is of the order of unity or greater (where $\lambda$ is the mean free path and $L$ is the system characteristic length). A fluid cannot be accurately modeled as a continuous media for Knudsen numbers of 0.2 or greater.

Another approach to high Kn flows is the direct simulation Monte Carlo (DSMC) method [1], a direct particle simulation technique based on kinetic theory. The fundamental idea of DSMC is to track a very large number of test particles, representing one or more actual gas molecules of physically correct molecular size, through representative collisions and boundary interactions and then to modify their positions and velocities appropriately in time. The core of the DSMC algorithm consists of four primary processes: move particles; index and cross-reference particles; simulate collisions; and sample the flowfield. The simplicity of the algorithm allows for straightforward incorporation of higher-order physical models and for application to complex geometries. The primary approximation of DSMC is to uncouple the molecular motions and intermolecular collisions over a sufficiently small time increment. Particle motions are modeled deterministically, while collisions are treated statistically. It is the probabilistic treatment of the collision process that restricts the applicability of the method to dilute gas flows. For monatomic gases undergoing binary collisions, DSMC has been shown rigorously to be equivalent to solving the basic Boltzmann equation [2]. The statistical error of a DSMC solution is inversely proportional to the square root of the total number of simulated particles, or sample size $N$, while the computational work involved is proportional to $N$.

While the DSMC technique has been extremely successful in predicting rarefied flows, it is limited by the large computational requirements of practical systems. Significantly greater computational resources are required, compared to numerical methods based on the Navier–Stokes equations, for problems in which both approaches are valid (i.e., near-continuum flows with slip effects). Further reductions in DSMC computational times are



0021-9991/95 $12.00

needed to enable tractable analyses of complex geometries with complex physical processes such as chemical reactions, wall catalysis, radiation effects, and ionization effects. Such reductions are possible through using parallel computers and efficient parallel algorithms. The parallelization of the DSMC technique has been the focus of many recent research efforts, such as the work of Wilmoth, Carlson, and Bird [3], Wong and Long [4], and Dietrich and Boyd [5].

In this paper, we report on our efforts to combine the DSMC with a method for particle tracking and sorting, the monotonic Lagrangian grid (MLG). The MLG data structure stores the locations and other attributes of particles in computer memory in a manner consistent with particle locations in physical space and maintains this order as the computation evolves. Using the MLG as the database structure for DSMC allows a time-varying network of nearest-neighbor "templates" to be used in lieu of the space-fixed cells required by DSMC for collision modeling and macroscopic property evaluation. The combination of DSMC and MLG has several significant benefits, namely, an automatically adapting grid, improved prediction accuracy (for a given grid size), reduced user effort, and decreased computational requirements through efficient parallelization [6]. Extensions of the DSMC-MLG to massively parallel computers offer truly significant advances in what can be computed.

## 2. MONOTONIC LAGRANGIAN GRID

The MLG [7–10] is a method of constructing and maintaining data structures for large particle simulations that is optimal for use on parallel and vector computers. Since the MLG maintains a direct correspondence between the indexing and the position of the particle, it is well suited for problems involving substantial local density variations. Recent applications of the MLG have been to problems in molecular dynamics, granular flow, and battle management [11–15].

The MLG data structure requires minimal memory and uses monotone arrays for indexing geometric positions and other physical attributes of the moving particles. The monotonicity feature ensures that particles close in physical space are stored in adjacent array locations. For a three-dimensional grid, the monotonicity constraints are

$$\begin{aligned} x(i,j,k) &\le x(i+1,j,k) && \text{for } 1 \le i \le N_x - 1, \text{ all } j, \text{ all } k \\ y(i,j,k) &\le y(i,j+1,k) && \text{for } 1 \le j \le N_y - 1, \text{ all } i, \text{ all } k \\ z(i,j,k) &\le z(i,j,k+1) && \text{for } 1 \le k \le N_z - 1, \text{ all } i, \text{ all } j \end{aligned} \quad (2)$$

where the set $(N_x, N_y, N_z)$ defines the data structure and $(i, j, k)$ represents the grid indices. Each particle is assigned three indices in the MLG data arrays for its three spatial coordinates. Other particle attributes, such as velocity components, are grouped together and stored as a single object in a record of fixed length (i.e., words or bytes). Objects are then arranged in computer memory by axis. Efficient mapping to memory locations depends on the computer memory architecture; optimal MLG libraries are available for various computer platforms.

Figure 1 gives an example of a simple two-dimensional MLG linking 16 particles [16]. The conditions defined by Eq. (2) are not sufficient to define a unique ordering of the particles within the data structure. Previous molecular dynamics applications have demonstrated that some rather undesirable MLG orderings can occur. The question of quality of an MLG and the issues involved in constructing an MLG with the best properties are addressed in a recent study [17].

The sorting algorithm used to construct an MLG from a random distribution of particles has an operation count which scales as $N \log N$. The basic MLG procedure is to sort all particle record data according to the monotonicity requirements of the first spatial direction, partition the result into subsets, and then sort each subset according to the next spatial direction, and so on. When all constraints are satisfied, particle data is considered in "MLG order." An illustrated example of this construction process is given by Picone *et al.* [18]. In time-dependent applications, the local monotonicity conditions are usually not met after individual molecules move and interact within a time interval. For instance, Fig. 2a shows a $4 \times 4$ MLG with corresponding velocity vectors at time $t$, while Fig. 2b shows the same 16 particles no longer in MLG order at time $t + \Delta t_g$. To reestablish the instantaneous correspondence between the indexing and the spatial locations of the particles, the MLG uses an algorithm that interchanges particle data in computer memory until the monotonicity conditions are reestablished. Such restructuring is discrete, reversible, efficient (faster than $N \log N$), and corrects local monotonicity violations without visible global changes. A resorted $4 \times 4$ MLG at time $t + \Delta t_g$ is shown in Fig. 2c.

Within a well-structured MLG, a particle's most influential neighbors are at most two or three indices away in data memory for most of the interaction laws we use. Using this concept,

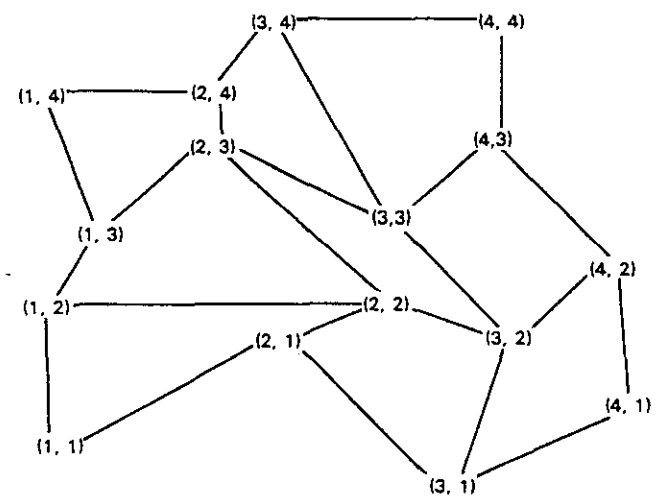


FIG. 1. Example of a two-dimensional monotonic Lagrangian grid.

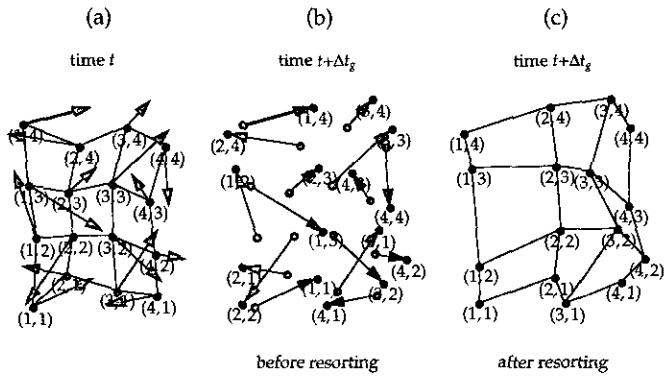


**FIG. 2.** A 4 × 4 MLG before (a) and after (b) the particles are moved, and after resorting (c). Arrows represent instantaneous velocity vectors in (a), identify initial molecular positions in (b).

we define the nearest-neighbors template by those particles that may have significant interactions with the center particle. The size of this template is controlled by specifying $\Delta_{io}$, the maximum index offset. Particles whose indices are offset from a given particle's indices by an amount less than or equal to $\Delta_{io}$ comprise the template. If the maximum index offsets for $i$, $j$, and $k$ have the same value $\Delta_{io}$, then the number of near neighbors, $N_{nn}$, is given by

$$N_{nn} = (2\Delta_{io} + 1)^3 - 1. \tag{3}$$

Hence, interacting particles are located within a small contiguous portion of the computer memory spanned by $2\Delta_{io}$, providing a substantial computational cost savings over methods which must search for near neighbors [18]. The programming logic is natural for massively parallel machines, where data for adjacent particles are stored in adjacent processors.

### 3. MLG IMPLEMENTATION IN DSMC

In a typical DSMC application, all simulated test particles are individually tracked as they move through a physical space that has been subdivided into a number of cells. This space-fixed cell network provides the geometric areas and volumes required to evaluate macroscopic flow properties. It is also used by the collision process, in which only particles located within the same cell at a given time are allowed to interact. To maintain an accurate solution, cell sizes must be small in regions of large macroscopic gradients [19]. Specifically, the dimensions of the cells must be much smaller than $L$, the scale length of the macroscopic gradients [20]. Using local values of density $\rho$, pressure $p$, temperature $T$, or volume $V$, this scale length is given by

$$L = \frac{\varphi}{|\partial\varphi/\partial x|} \quad \text{for } \varphi = \rho, p, T, \text{ or } V. \tag{4}$$

Maintaining this requirement is difficult for complex problems whose flowfield properties are not predictable.

It is possible to eliminate the cell dependency of the DSMC collision process by defining a local subset of possible collision partners in a specified cutoff distance, $R_c$. This approach permits only neighboring particles to interact. However, it is computationally impractical for large sample sizes because the cost scales as the square of $N$, and so directly conflicts with the DSMC requirement on the size of $N$ for acceptable statistical accuracy.

The DSMC algorithm used here is based on Bird's original implementation [1]. A monatomic gas is modeled as a collection of elastic hard-sphere molecules. The collision routine, referred to as the "no time counter" or NTC approach [19], is a probabilistic scheme that relies on acceptance–rejection statistics to collide particles before they are randomly scattered. The acceptance–rejection technique uses random numbers to determine whether a randomly selected pair of molecules will interact. The probability of the two-body collision is given by the product of the total collision cross-section $\sigma_T$ and relative velocity $c_r$ divided by the product's maximum value within the cell. Bird's original indexing scheme [1] is retained to serve as a benchmark for both the one- and two-dimensional MLG-based implementations.

Incorporating the MLG into the DSMC test program requires three primary steps:

1. Eliminating space-fixed cells in lieu of nearest-neighbor templates;
2. Replacing the standard indexing scheme with MLG particle tracking and sorting routines;
3. Calculating areas and volumes of the time-varying templates.

These modifications result in a number of procedural changes to the standard DSMC method, as highlighted in the DSMC-MLG flowchart presented in Fig. 3. The details of this process are described here for a two-dimensional problem; it is straightforward to extend these ideas to three dimensions.

The first step, to eliminate space-fixed cells, requires two additional operations immediately after the zero-time state of the gas is set (Fig. 3): construct the MLG, and then subdivide the MLG into a specified number of nearest-neighbor templates. Taken together, these are analogous to, and effectively replace, the grid generation required prior to any standard DSMC application.

The MLG construction algorithm uses a shell sort scheme [21] to sort the random distribution of molecules into MLG order. Compared to subsequent sorting steps that reestablish MLG monotonicity, this initial step typically involves the most computational work and therefore is important to optimize. Several sorting controls are available for this purpose; optimal settings, however, must be determined for different problems.

The indices that identify individual molecules within the

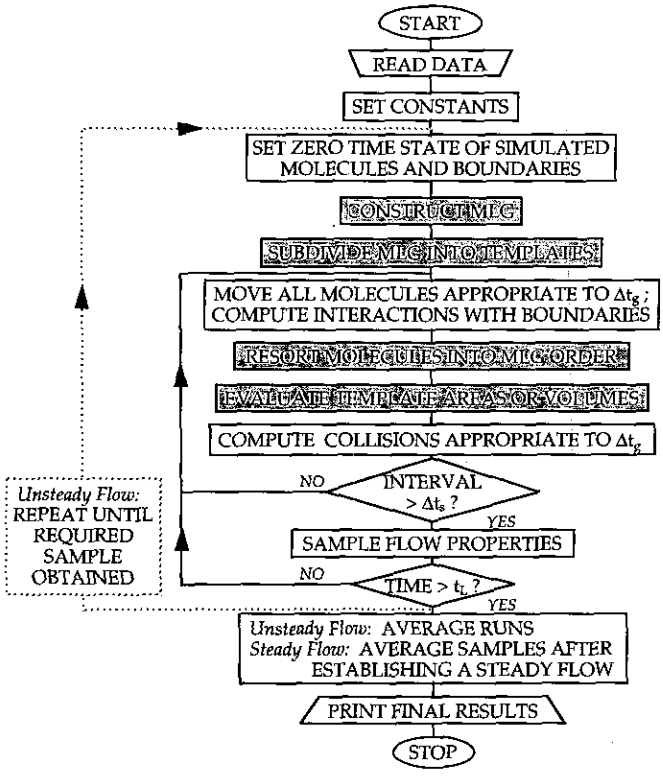


**FIG. 3.** DSMC-MLG flowchart.

MLG are next grouped into subsets based on computer memory location. Since there is a one-to-one correspondence between the indexing and spatial locations of the molecules, this grouping process also defines the grid, which consists here of a network of templates in physical space. The current MLG implementation specifies the grid size by choosing the number of templates along each axis. The same grid size is used for each independent computation of an ensemble. Figure 4 shows a schematic of a 15 × 9 MLG that has been subdivided into fifteen 3 × 3 templates. Note that every molecule belongs to one and only one template. The relative positions of the templates do not change during the course of a simulation; neither do the index pairs $(i, j)$ of each template. For instance, template 1 is always made up of particles whose indices $(i, j)$ are given by $1 \le i \le 3$ and $1 \le j \le 3$. However, the actual particle associated with an index pair may vary from timestep to timestep. For simplicity, templates are assumed to have equal population counts (that is, the same number of particles in each template) and to be square in index space.

The second step requires replacing the standard DSMC molecular indexing with MLG-based routines (Fig. 3). Sorting and tracking algorithms are used to resort particles into MLG order. The resorting process may change a template's physical size, shape, and individual particle makeup; this is depicted in Fig. 5, where a 3 × 3 template is carried through the same procedures described in Fig. 2. It is this resorting step that automatically adapts the grid in physical space to local number densities.

The third step is to evaluate individual template areas, which are required by the collision modeling and sampling steps (Fig. 3). For instance, the collision counter calculation in the NTC model, given by

$$N_{cp} = \frac{N_t^2 S(\sigma_T c_r)_{max} \Delta t_g}{2A_{t_{i,j}}}, \tag{5}$$

depends on instantaneous template areas $A_{t_{i,j}}$ in a two-dimensional simulation. In this equation, $N_t$ represents the template population, $\sigma_T$ the total collision cross-section, $c_r$ the relative velocity, and $S$ the ratio of actual-to-simulated molecules. Similarly, macroscopic properties generated during the sampling step are valid at the centers-of-mass of each of the templates, which in turn are functions of $A_{t_{i,j}}$.

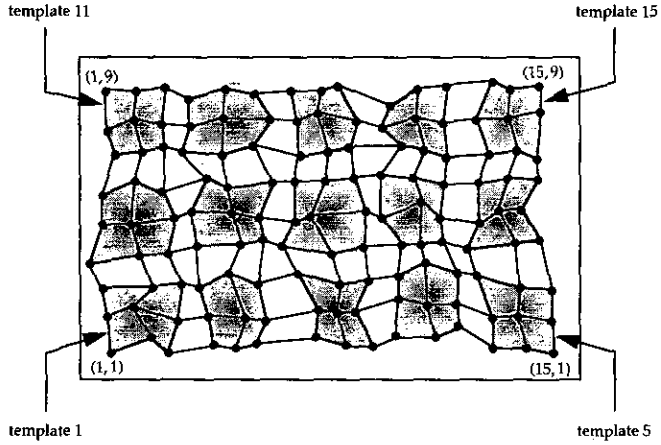


**FIG. 4.** Subdivision of a 15 × 9 MLG into a grid of 15 nearest-neighbor templates. Shaded regions delineate template areas.

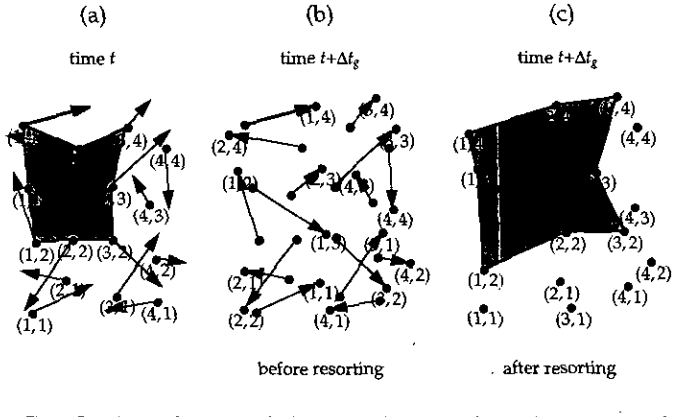


**FIG. 5.** A 3 × 3 template before (a) and after (b) the particles are moved, and after resorting (c). Arrows represent instantaneous velocity vectors in (a), identify initial molecular positions in (b). Shaded regions delineate template areas.

Two area-evaluation methods were tested here. Method I approximates the area of each template by the area of a rectangular region assigned to the template. The sides of this rectangular region pass through the midpoints of segments A-B, A-C, A-D, and A-E, as shown schematically in Figure 6. Collectively, these nonoverlapping rectangles account for the entire computational domain. In equation form, $A_{t_{i,j}}$ is given by

$$A_{t_{i,j}} = \tfrac{1}{2}[(x_{t,cm})_{i+1,j} - (x_{t,cm})_{i-1,j}][(y_{t,cm})_{i,j+1} - (y_{t,cm})_{i,j-1}], \quad (6)$$

where the subscript cm indicates a center-of-mass value. Initial DSMC-MLG simulations showed that the local accuracy of method I decreases as density variations increase locally. This is because the method is first-order accurate; large changes in local number density in either direction that are nonlinear are not accounted for in Eq. (6).

Method II was proposed as an improvement on I for simulations with large density variations. Here a template area is computed by first subdividing the template into pie-shaped slices, as sketched in Fig. 7. Each of these triangular sections is made up of two boundary molecules and the template center-of-mass. The individual triangular areas are evaluated and summed to give the area of the template. This summation process leads to a compact expression which does not involve the location of the center-of-mass [22]. With boundary molecules labeled $P_1$, $P_2$, ..., $P_n$ in a counterclockwise fashion, $A_{t_{i,j}}$ is given by

$$A_{t_{i,j}} = \tfrac{1}{2}[(x_1y_2 + x_2y_3 + \cdots + x_ny_1) - (y_1x_2 + y_2x_3 + \cdots + y_nx_1)], \quad (7)$$

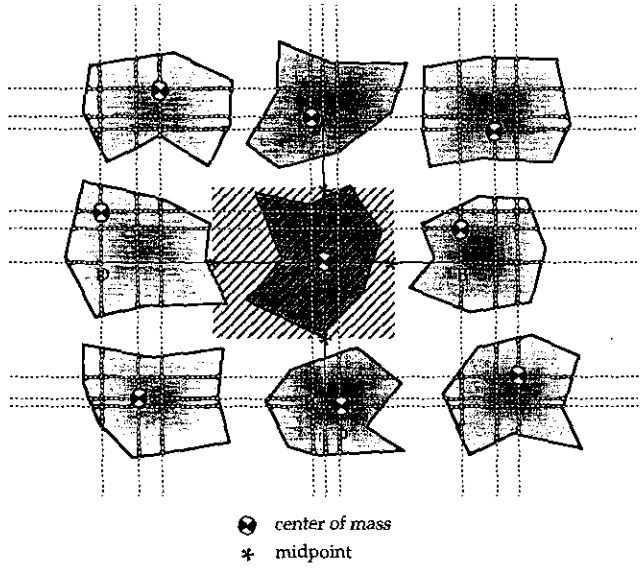


FIG. 6. Schematic for template area evaluation method I. Shaded regions delineate template areas.

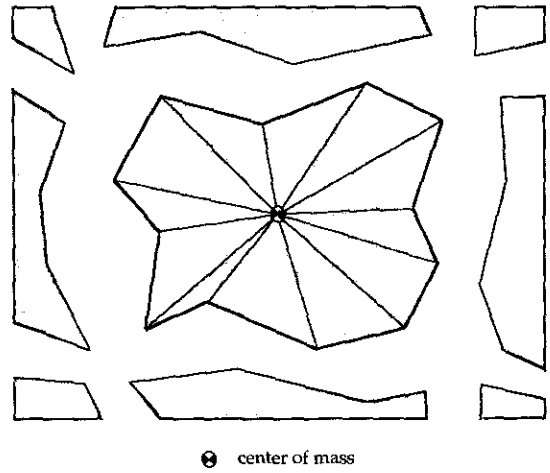


FIG. 7. Schematic for template area evaluation method II. Shaded regions delineate template areas.

where $x_l$ and $y_m$ are coordinates of two different boundary molecules. However, as seen in Fig. 4, the summation of all individual template areas accounts for only 75–80% of the computational domain. To account for the rest of the domain, weighting factors based on relative areas are computed for the templates. These weighting factors are then used to proportionally increase individual template areas until the total computational area is accounted for, preserving relative template sizes. In a computational load study using the Rayleigh test problem [6], the evaluation of template areas using method II accounted for 0.6% of the computational work associated with a single independent calculation. For the DSMC-MLG results presented in this paper, this was the method of choice.

Regions of high molecular concentrations and extremely large gradients may pose a problem to using method II to evaluate areas. For example, Fig. 8 presents a schematic of a

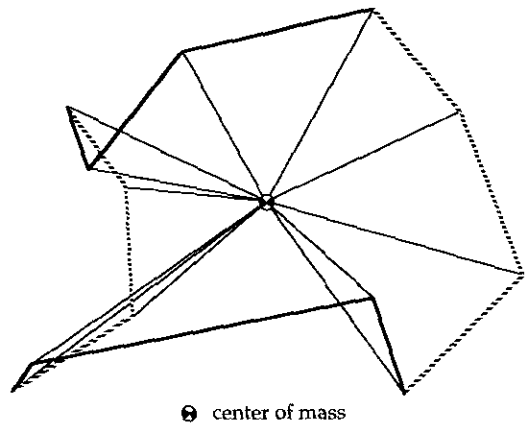


FIG. 8. Schematic of a 4 × 4 template that may cause area evaluation difficulties. Dark solid and dashed lines represent $x$ and $y$ links, respectively.

highly skewed 4 × 4 template whose particles are in MLG order, but whose irregular shape may cause inaccuracies that necessitate a larger ensemble size for a given statistical accuracy. A MLG regularization technique [17] has recently been developed that minimizes the occurrence of such highly skewed templates during MLG resorting procedures. In a subsequent study [22], this regularization technique was incorporated into DSMC-MLG and successfully demonstrated for this purpose.

## 4. TEST PROBLEMS

The initial development and testing of the DSMC-MLG was carried out in a series of steps on Sun Sparc workstations. These steps included numerically validating one- and two-dimensional DSMC-MLG codes and demonstrating the adaptive gridding in several test problems.

### *Rayleigh Test Problems*

Variations of the classical Rayleigh problem, a standard DSMC validation problem [4, 23], were used to test one- and two-dimensional versions of the DSMC-MLG, and to highlight the advantages of the new algorithm. Only two-dimensional results are presented here. Nondimensional variables are denoted by a tilde.

The first 2D Rayleigh test problem, shown schematically in Fig. 9, consists of a flat plate instantaneously heated, and then accelerated to a constant speed through an undisturbed gas. The plate speed, $u_{tw}$, is twice the most probable molecular speed of the undisturbed gas, $V_{mp}$, and the plate temperature $T_{tw}$, is 1.6 times that of the undisturbed gas, $T_{ug}$. The plate surface is modeled as a fully diffuse boundary, where the physical attributes of the reflected molecules are distributed according to the half-range Maxwellian distribution based on wall temperature. The remaining boundaries are specularly reflecting; that is, the normal velocity component of an incident molecule is reversed while the parallel component remains unchanged. The

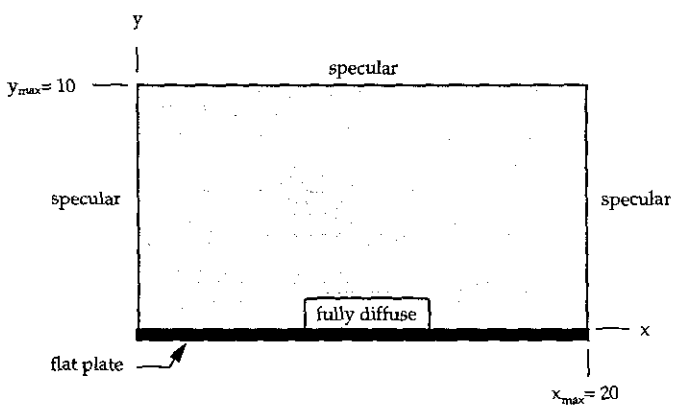


**FIG. 9.** Schematic of a two-dimensional version of the Rayleigh validation problem. Boundary conditions: $u(x, y, 0) = 0$; $u(x, 0, t) = 2V_{mp}$ for $t > 0$; $T(x, 0, t) = 1.6T_{ug}$ for $t > 0$.

**TABLE I**

Computational Parameters for the Rayleigh Validation Problem

| | DSMC | DSMC-MLG |
|---|---|---|
| $(N_c)_{ug}$, $N_t$ | 50 | 49 |
| $N_{ic}$ | 40000 | 39200 |
| $\Delta x/\lambda_{ug}$ | 0.5 | 0.5[a] |
| $\Delta y/\lambda_{ug}$ | 0.5 | 0.5[a] |
| $\Delta\tilde{t}_g$ | 0.1772 | 0.1772 |

[a] At $\tilde{t} = 0$.

gas initially consists of a uniform distribution of hard-sphere molecules. The one-dimensional Rayleigh solution is reproduced at $\tilde{x} = \tilde{x}_{max}/2$ at early times in the calculation, before disturbances from the left, right, and upper boundaries have had a chance to disrupt the one-dimensional nature of the flow.

Simulations were performed using both the standard DSMC and the new DSMC-MLG. The parameters describing the calculations are listed in Table I. Ensembles consisted of 1000 independent computations, each using 800 cells or templates in a 40 × 20 arrangement. The physical size of each cell and template are the same only at time zero. Throughout the simulation, each space-fixed cell initially contains 50 molecules, while every MLG template is 7 × 7. The global timestep, $\Delta t_g$, is 20% of the mean collision time of the undisturbed gas. All variables are nondimensionalized [1]; one unit of length is a freestream mean-free path, and a unit of time is equivalent to the freestream mean collision time.

Comparisons of DSMC-MLG and DSMC axial velocity, number density, and temperature profiles at three different sampling time intervals are shown in Figs. 10a–c. The DSMC results shown here agree with tabulated results [1] given by Bird. The location of each symbol in the curves represents the ensemble-averaged coordinates (the grid distribution) of the center-of-mass of a particular template or cell. At time zero, the symbol locations for both simulations are identical, but they vary at later times because the MLG grid is changing in time. Figure 10d compares the shear stress acting on incident molecules at the wall.

To demonstrate the advantages of the DSMC-MLG, the two-dimensional Rayleigh validation problem was modified to produce steeper gradients and stronger recirculation regions in the flow. Specifically, both the lower and upper boundaries are instantaneously heated to 1.6 $T_{ug}$, and then set in motion at a speed of 10 $V_{mp}$. The upper and lower boundaries are modeled as fully diffuse, while the left and right boundaries remain specularly reflecting. The global timestep, $\Delta\tilde{t}_g$, is set to 0.03545 (e.g., 4% of the freestream mean collision time), and the ensemble consists of 200 independent computations. All other computational parameters remained the same as in Table I.

The high-gradient test problem was run using both the NTC collision model, and Bird's original "time counter" or TC

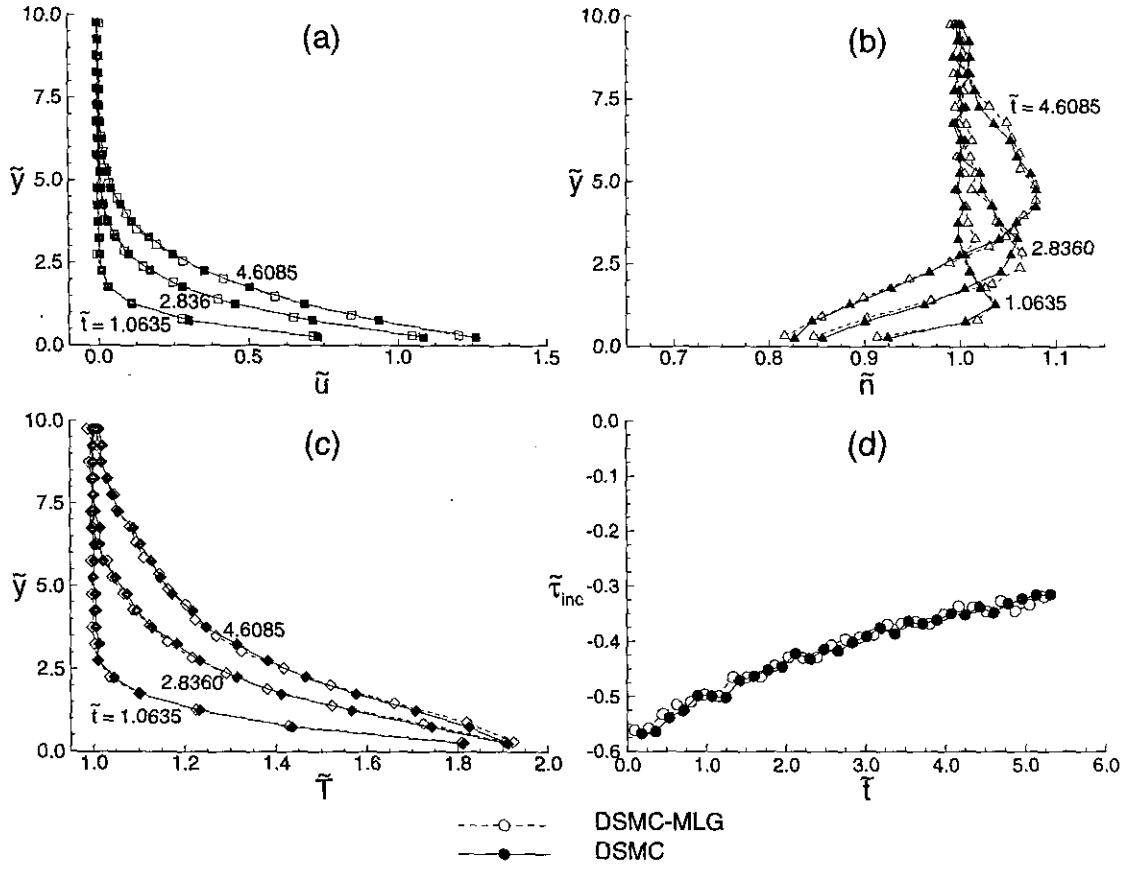


**FIG. 10.** Comparison of DSMC-MLG and DSMC axial velocity (a), number density (b), and temperature (c) profiles at various sampling times, and time histories of shear stress acting on incident molecules at the wall (d). Rayleigh validation problem, $\tilde{u}_{tw} = 2.0$, $\tilde{T}_{tw} = 1.6$. (*Note.* Results sampled at $\tilde{x} \approx \tilde{x}_{max}/2$.)

model [1]. Although DSMC-MLG results were identical using both collision models, DSMC results were not. Only TC model results are discussed here.

A qualitative comparison of the TC-based DSMC-MLG and the DSMC solutions across the symmetry line of Fig. 11 reveals little difference between the two. The top half of the figure shows 10 representative streamtraces traversing the DSMC flowfield at a nondimensional time of 3.722, or after 105 timesteps. Dotted lines connecting the centers of the 800 square cells which make up the space-fixed grid are overlaid for reference. The bottom half of Fig. 11 shows comparable DSMC-MLG streamtraces. In this case, the overlaid, dotted lines connect the average center-of-mass locations of the nearest-neighbor templates. Figure 12 shows contours of the number density in the interior of the computational domain at $\tilde{t} = 3.722$. DSMC predictions are displayed in the top half, and DSMC-MLG predictions in the bottom half; grids are again overlaid for reference.

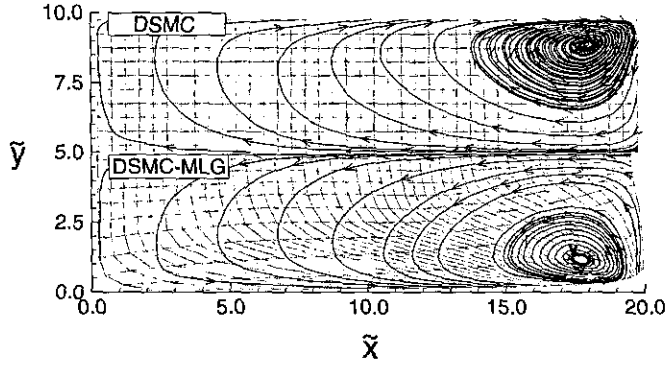


**FIG. 11.** DSMC-MLG and DSMC streamtraces at $\tilde{t} = 3.722$. Rayleigh problem with $\tilde{u}_{tw} = \tilde{u}_{uw} = 10.0$, $\tilde{T}_{tw} = \tilde{T}_{uw} = 1.6$. (*Note.* Grids overlaid for reference.)

A quantitative comparison of the results of the two cases shows significant differences in the density field predictions, especially near the hot moving boundaries. In an attempt to explain the differences between the two solutions, the DSMC simulation was rerun and individual cell populations were monitored. At later times in the simulation (e.g., $\tilde{t} > 2$), the populations of many cells near the upper and lower boundaries fell below 20 particles per cell, and sometimes even below 10. As a consequence, instantaneous collision rates predicted by the

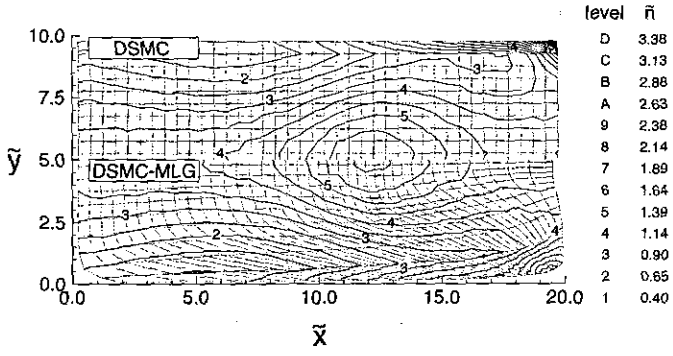


**FIG. 12.** Comparison of DSMC-MLG and DSMC number density contours at $\tilde{t} = 3.722$. Rayleigh problem with $\tilde{u}_{1w} = \tilde{u}_{uw} = 10.0$, $\tilde{T}_{1w} = \tilde{T}_{uw} = 1.6$. (*Note.* Grids overlaid for reference.)

TC method within these sparsely populated cells were suspect. The large differences in collision rates predicted by the two algorithms are shown in Fig. 13. Incorrect collision rates near the upper and lower walls of the DSMC simulation have a large effect on the results in the bulk of the computational domain. Two other factors also contribute to the quantitative differences between the DSMC-MLG and the DSMC solutions. The DSMC calculation violates the fundamental requirement of small cell sizes in regions of large macroscopic gradients (such as near the moving boundaries and in the downstream corner) at later sampling times. In addition, the DSMC-MLG simulation benefits from automatic grid restructuring. In the test problem shown here, on a small scalar computer, the cost of the DSMC-MLG improvement in the results is a factor of four in the runtime.

The purpose of this high-gradient problem was to highlight certain features of the adaptive gridding automatically provided by the MLG-based algorithm. Conditions were specifically chosen that are known to produce inaccurate DSMC results if they are not anticipated and appropriately handled. The TC-based simulation was particularly susceptible due to the well-documented tendency of the collision model to produce inaccurate collision rates in highly nonequilibrium flows. The DSMC-MLG method did not experience problems with the TC model because of the basic definition of a template in the MLG, which is tantamount to cell populations remaining constant. In addition, the adaptive gridding fulfilled DSMC cell-size requirements in most of the computational domain by providing higher resolution in regions of high densities. This was achieved without any additional user interference.

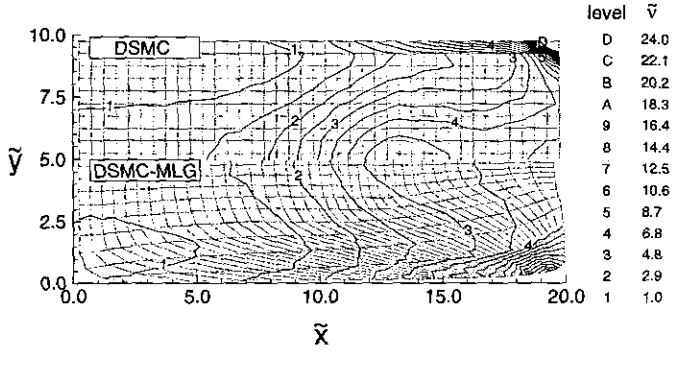


**FIG. 13.** Comparison of DSMC-MLG and DSMC collision rate contours at $\tilde{t} = 3.722$. Rayleigh problem with $\tilde{u}_{1w} = \tilde{u}_{uw} = 10.0$, $\tilde{T}_{1w} = \tilde{T}_{uw} = 1.6$. (*Note.* Grids overlaid for reference.)

**TABLE II**

Normalized Scalar Runtimes for the Rayleigh Test Problems

| | Validation problem | High-gradient problem[a] |
|---|---|---|
| DSMC | 1.0 | 2.8 |
| DSMC-MLG | 3.1 | 11.0 |

[a] Extrapolated to an equivalent $N$.

Table II summarizes the relative scalar performance of the two Rayleigh problem simulations; cpu times have been scaled to reflect equivalent ensemble sizes and normalized by the smallest value. These values are indicative of either collision model for both algorithms.

*Circular-Diaphram Test Problem*

A series of shock tube problems was simulated to further highlight the adaptive gridding capabilities of the DSMC-MLG. The test problem discussed here models a circular diaphram that initially separates the driver and driven gases at time zero. The starting geometry is shown in Fig. 14, where a 90° section

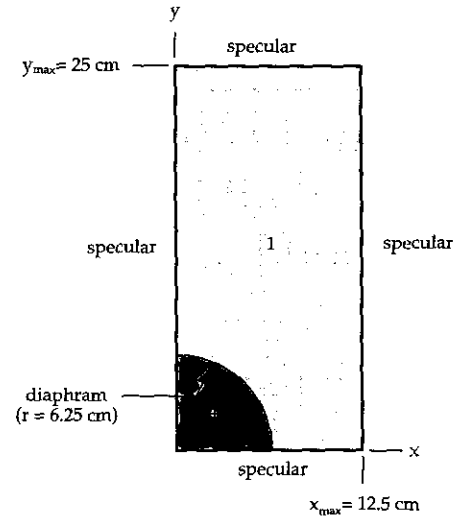


**FIG. 14.** Schematic of the circular diaphram shock tube problem. Initial conditions: $n_4 = 10n_1$; $\lambda_4 = 0.1\lambda_1$; $T_4 = T_1$.

**TABLE III**

Computational Parameters for the Circular Diaphram Shock Tube Problem

| | Driver section | Driven section |
|---|---|---|
| $N_t$ | 25 | 25 |
| $N_{tc}$ | 16287 | 14963 |
| $\lambda_{ug}$ (cm) | 1.0 | 10.0 |
| $[(\Delta x)_t/\lambda]_{ug}$ | 0.2 | 0.05 |
| $[(\Delta y)_t/\lambda]_{ug}$ | 0.2 | 0.1 |
| $\Delta t_g$ (s) | 3.3E − 07 | 3.3E − 07 |
| $n_{ug}$ (cm$^{-3}$) | 4.1E + 14 | 4.1E + 13 |

of the diaphram is positioned in the lower left corner of a 12.5 cm by 25.0 cm rectangular domain. Both gases, initially at the same temperature of 273 K, are modeled as hard-sphere helium atoms. The driver-to-driven number density ratio is 10 : 1. The mean-free path of the undisturbed gas, $\lambda_{ug}$, is 1.0 cm in the driver section and 10.0 cm in the driven section. The ratio of actual-to-simulated molecules, $S$, is set to $7.81 \times 10^{12}$ in both sections. Artificially high values of number density are used to keep computational requirements reasonable. All computational boundaries are modeled as specularly reflecting.

An ensemble of 500 independent computations is sufficient to show the adaptive grid performance. A total of 1250 templates, arranged in a 25 × 50 grid, were used in each computation. Each template contains 25 atoms in a 5 × 5 mesh. The global timestep, $\Delta t_g$, is set to 50% of the mean collision time of the undisturbed driver gas. Values of various computational parameters used in each independent calculation are listed in Table III. The simulation used the NTC collision model.

The evolution of the mass density is shown by the sequence of contours in Fig. 15. The time-dependent variations in $\rho$ caused by the expansion of the high-density helium into the driven section after the diaphram is removed results in changes and readjustments of the DSMC-MLG grid. The evolution of this adaptive grid, represented by lines that connect the average center-of-mass locations of the time-varying templates, is presented in Fig. 16. Initially, 52% of the MLG templates were distributed throughout the driver section. The shapes and areas of the templates gradually change as the driver gas propagates radially outward from the lower left corner of the computational domain.

To achieve comparable accuracy using the standard DSMC algorithm without adaptive gridding, more than 1250 space-fixed cells would have to be used. For instance, the maximum dimensions of each cell in a space-fixed grid are dictated by the minimum local values of $\lambda$ encountered during a simulation. Local property changes as the flow evolves must be anticipated (possibly using a preliminary coarse-grid simulation) to ensure that DSMC requirements are met. If adaptive gridding is not available, the DSMC grid in the driven section would require

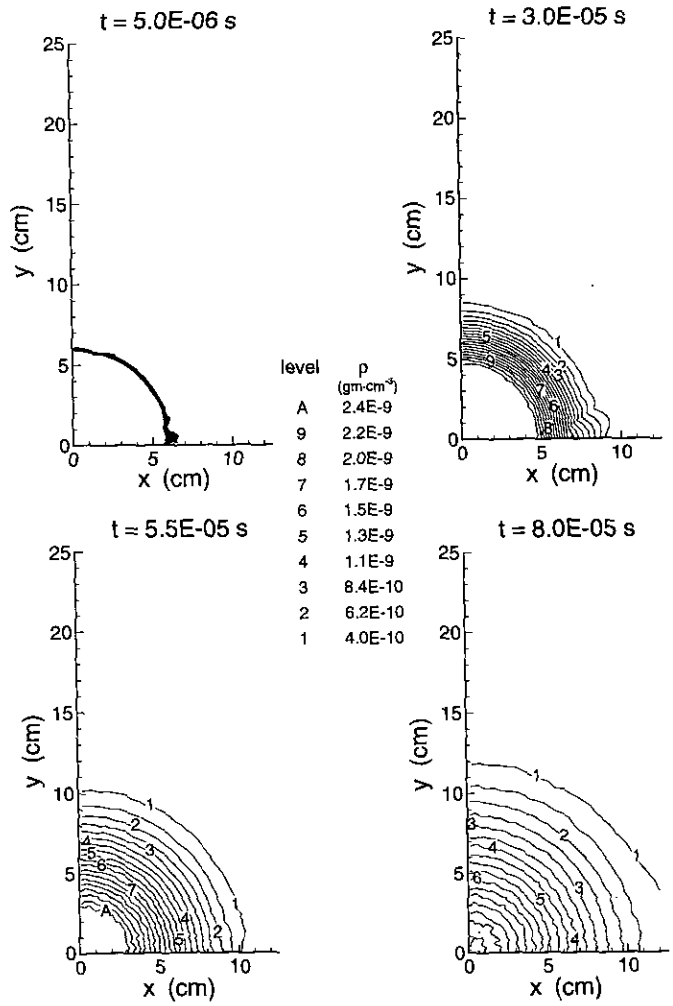


**FIG. 15.** DSMC-MLG density contours at four sampling times. Circular diaphram shock tube simulation. (*Note.* Densities along the computational boundaries are not shown.)

substantially more cells than the 599 templates used in the present DSMC-MLG simulation. The best flow resolution for a given grid size is achieved automatically by the DSMC-MLG algorithm.

*Effectiveness of Grid Adaption*

The changes in the grid in the circular-diaphram test problem raise several questions. How effective is the adaptive gridding? Are local cell-size requirements met at all times? Can the adaptation process be improved? These questions can be addressed through a quantitative analysis of the time-dependent gridding process.

To assess the effectiveness of the grid adaption, diagnostic quantities are required. For accurate DSMC simulations, linear cell dimensions must be smaller in each spatial direction than the scale length $L$ of the local macroscopic gradients [20]. For

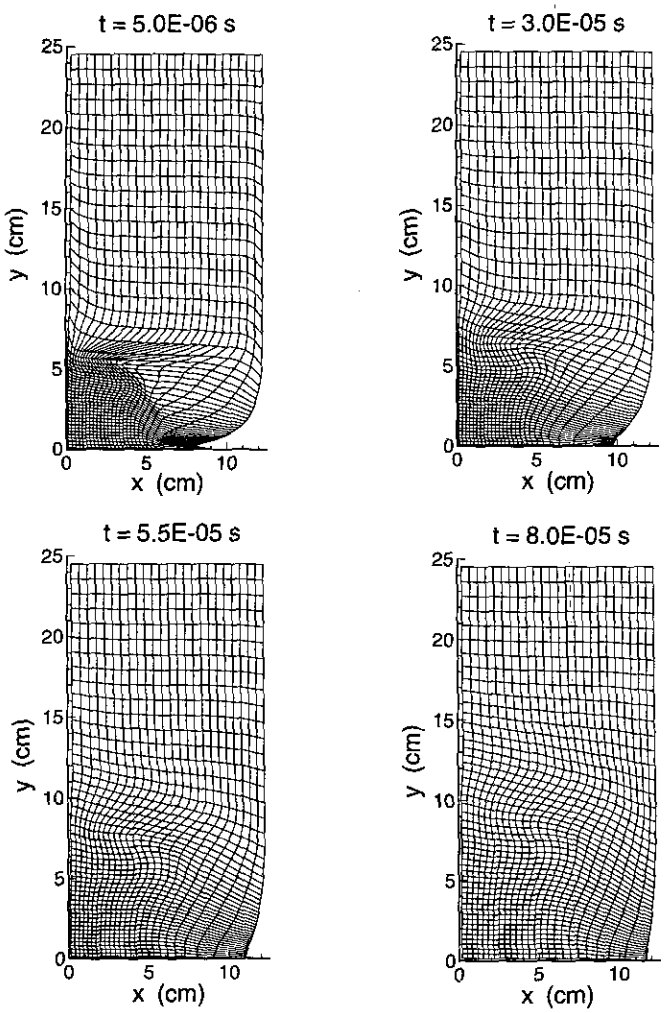


**FIG. 16.** Time evolution of the DSMC-MLG adaptive grid. Circular diaphram shock tube simulation.

a two-dimensional simulation, these requirements, with the help of Eq. (4), can be expressed as

$$\Delta x \le L_x = \frac{\varphi}{|\partial\varphi/\partial x|} \quad \text{for } \varphi = \rho, p, T, \text{ or } V \tag{8}$$

and

$$\Delta y \le L_y = \frac{\varphi}{|\partial\varphi/\partial y|} \quad \text{for } \varphi = \rho, p, T, \text{ or } V. \tag{9}$$

Since DSMC-MLG macroscopic properties are valid at the average center-of-mass of each MLG template, finite difference expressions can be used to approximate the derivatives. Using second-order accurate central differences for interior templates, the derivatives become

$$\left(\frac{\partial\varphi}{\partial x}\right)_{i,j} = \frac{\varphi_{i+1,j} - \varphi_{i-1,j}}{2\Delta x_{i,j}} \tag{10}$$

and

$$\left(\frac{\partial\varphi}{\partial y}\right)_{i,j} = \frac{\varphi_{i,j-1} - \varphi_{i,j-1}}{2\Delta y_{i,j}}. \tag{11}$$

For templates adjoining computational boundaries, second-order accurate one-sided differences are available. For example, the $y$-derivative term for templates located just above the lower boundary of the computational shock tube becomes

$$\left(\frac{\partial\varphi}{\partial y}\right)_{i,j} = \frac{-3\varphi_{i,j} + 4\varphi_{i,j+1} - \varphi_{i,j+2}}{2\Delta y_{i,j}}. \tag{12}$$

Second-order accurate approximations for other derivatives near the boundaries are similar. Finally, the diagnostic quantities for assessing the DSMC-MLG adaptive gridding are

$$\frac{\Delta x}{L_x} = \frac{\Delta x}{(\rho/|\partial\rho/\partial x|)} \tag{13}$$

and

$$\frac{\Delta y}{L_y} = \frac{\Delta y}{(\rho/|\partial\rho/\partial y|)}, \tag{14}$$

where local density values are used to represent the macroscopic property $\varphi$ and appropriate second-order accurate finite differences are used to represent the derivatives of $\rho$.

Figure 17 presents variations of $\Delta x/L_x$ and $\Delta y/L_y$ for the

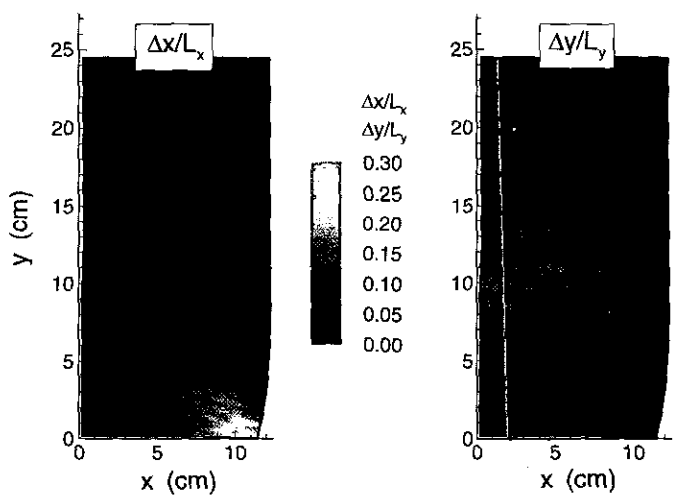


**FIG. 17.** DSMC-MLG $\Delta x/L_x$ and $\Delta y/L_y$ distributions at $t$ = 8.0E − 05 s. Circular diaphram shock tube simulation.

circular-diaphram test problem $8.0 \times 10^{-5}$ s into the simulation. Small differences exist in the two distributions. This directional bias in the grid adaption is caused by the use of a *fixed* grid size from one time interval to the next. Sorting procedures of the current DSMC-MLG algorithm require specification of an "$x$ by $y$" grid size at time zero that does not change during the course of a simulation. Hence, the effectiveness of the current implementation of the DSMC-MLG adaptive gridding is directionally dependent. Nevertheless, values of the diagnostic quantities are significantly less than one everywhere, indicating that DSMC cell-size requirements have been met throughout the flowfield, even though there is an order-of-magnitude variation in the mean-free path.

## 5. DISCUSSION AND CONCLUSIONS

Several advantages of the DSMC-MLG algorithm have been demonstrated in the previous section using various test problems. First, the MLG adds a time-varying grid that automatically adapts to the local number densities in the flowfield. Second, the MLG routines for sorting are already optimized for two- and three-dimensional applications on parallel computer architectures. Finally, the TC collision model can be used with confidence in DSMC-MLG simulations, if so desired.

The primary feature of the DSMC-MLG algorithm is the automatic grid adaption provided by the Lagrangian data structure. A number of computational benefits arise as a direct consequence, including: automatic changes in grid resolution according to properties of the flow; relaxation of the DSMC requirement that cell dimensions be very small in directions of large macroscopic gradients [19]; higher accuracy for a given grid size; and simplified representation of complex geometries. All of these help to improve the performance of the original DSMC algorithm and increase user-friendliness.

Modeling more complex geometries is possible by distributing molecules along stationary or moving solid boundaries that define the geometry itself. A flag added to the physical attributes stored by the MLG data structure identifies these molecules as boundary points so that they can be treated appropriately.

The sub-cell technique [20] is highly recommended by Bird for use within collision modeling procedures. The purpose of the sub-cell technique is to improve particle interaction accuracy by ensuring that collisions occur only between near neighbors. The basic principle of this technique is inherent to the Lagrangian data structure employed by DSMC-MLG. That is, collision partners in a DSMC-MLG simulation are chosen from within time-varying MLG templates which, by definition, are comprised of near neighbors. Specifying a maximum index offset, $\Delta_{io}$, for collision pair selection further restricts interactions to nearest neighbors.

Previous applications of the MLG have used optimized two- and three-dimensional algorithms on parallel machines. Furthermore, MLG performance does not depend on problem size because of the direct scalability of the MLG algorithms on parallel architectures. Massively parallel computers offer computing speeds that are potentially orders of magnitude faster than vector and scalar equivalents. Implementing the new DSMC-MLG algorithm on massively parallel computers will significantly lower computational costs, allowing the simulation of large-scale transition regime flows that were previously not feasible. In fact, a recent DSMC-MLG implementation on a Connection Machine achieved a two orders-of-magnitude decrease in computing time with minimal optimization over the serial computer counterpart [22].

The incorporation of the MLG in direct simulation Monte Carlo also alleviates load balancing issues raised in a recent parallel DSMC application [4]. For instance, the number of collision pairs during a given timestep (Eq. (5)) involves the square of the local template population $N_t$. Constant and equal template sizes used by the DSMC-MLG reduce processor load variations.

One drawback to the DSMC-MLG algorithm is that the cost of scalar processing appears more expensive (Table II). However, coupling a Lagrangian data structure to the DSMC method provides significant improvements, such as automatic grid restructuring, to a proven high-Kn tool. Optimizing the MLG sorting and tracking procedures for a given scalar application minimizes the computational penalties. The most efficient DSMC-MLG implementations will occur on parallel computer architectures.

## ACKNOWLEDGMENTS

This research is jointly supported by the Advanced Research Projects Agency, the Office of Naval Research through the Naval Research Laboratory, and the Air Force Palace Knight Program. The authors thank R. S. Sinkovits and D. E. Fyfe of the Naval Research Laboratory for help with various aspects of the MLG algorithms, and G. A. Bird of GAB Consulting Pty Ltd. for his comments and suggestions.

## REFERENCES

1. G. A. Bird, *Molecular Gas Dynamics* (Clarendon Press, Oxford, 1976).
2. K. Nanbu, "Theoretical Basis of the Direct Simulation Monte Carlo Method," in *Proceedings, 15th Int. Symp. Rarefied Gas Dynamics*, Vol. 1, edited by V. Boffi and C. Cercignani (Teubner, Stuttgart, 1986), p. 369.
3. R. G. Wilmoth, A. B. Carlson, and G. A. Bird, AIAA Paper No. 92-2861, 1992 (unpublished).
4. B. C. Wong and L. N. Long, AIAA Paper No. 92-0564, 1992 (unpublished).
5. S. Dietrich and I. D. Boyd, AIAA Paper No. 94-0355, 1994 (unpublished).
6. B. Z. Cybyk, E. S. Oran, J. P. Boris, and J. D. Anderson, Jr., AIAA Paper No. 94-0354, 1994 (unpublished).
7. J. P. Boris, *J. Comput. Phys.* **66,** 1(1986).
8. J. P. Boris and S. G. Lambrakos, *The Free-Lagrange Method*, edited by M. J. Fritts *et al.* (Springer-Verlag, New York, 1985), p. 158.
9. J. P. Boris and S. G. Lambrakos, in *Proceedings, 1985 Summer Computer Simulation Conference, Chicago, Illinois, 1985* (North-Holland, Amsterdam, 1985), p. 206.
10. S. G. Lambrakos and J. P. Boris, *J. Comput. Phys.* **73,** 183 (1987).
11. S. G. Lambrakos, M. Peyrard, E. S. Oran, and J. P. Boris, *Phys. Rev. B* **39,** 993 (1989).

12. S. G. Lambrakos, J. P. Boris, R. H. Guirguis, M. Page, and E. S. Oran, *J. Chem. Phys.* **90,** 4473 (1989).

13. R. L. Kolbe, J. P. Boris, and J. M. Picone, NRL Memorandum Report 6705, 1990 (unpublished).

14. L. Phillips, R. S. Sinkovits, E. S. Oran, and J. P. Boris, NRL Memorandum Report 4440-92-6998, 1992 (unpublished).

15. J. H. Dunn and S. G. Lambrakos, *J. Comput. Phys.* **111,** 15 (1994).

16. E. S. Oran and J. P. Boris, *Numerical Simulation of Reactive Flow* (Elsevier Science, New York, 1987), p. 388.

17. R. S. Sinkovits, J. P. Boris, and E. S. Oran; *J. Comput. Phys.* **108,** 368 (1993).

18. J. M. Picone, S. G. Lambrakos, and J. P. Boris, *SIAM J. Sci. Stat. Comput.* **11,** 368 (1990).

19. G. A. Bird, "Perception of Numerical Methods in Rarefied Gasdynamics," in *Progress in Astronautics and Aeronautics,* Vol. 118, edited by E. P. Muntz (AIAA, Washington, DC, 1989).

20. G. A. Bird, *Molecular Gas Dynamics and the Direct Simulation of Gas Flows* (Clarendon Press, Oxford, 1994).

21. W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vettering, *Numerical Recipes* (Cambridge Univ. Press, Cambridge, UK, 1989), p. 244.

22. C. K. Oh, R. S. Sinkovits, B. Z. Cybyk, E. S. Oran, and J. P. Boris, AIAA Paper No. 95-0409, 1995 (unpublished).

23. R. G. Wilmoth, AIAA Paper No. 89-1666, 1989 (unpublished).