JOURNAL OF THERMOPHYSICS AND HEAT TRANSFER
Vol. 11, No. 2, April–June 1997

Comparison of Grid-Definition Schemes for Monte Carlo Simulations

Robert P. Nance*
North Carolina State University, Raleigh, North Carolina 27695-7910
Richard G. Wilmoth†
NASA Langley Research Center, Hampton, Virginia 23681-0001
and
H. A. Hassan‡
North Carolina State University, Raleigh, North Carolina 27695-7910

This paper examines spatial-discretization approaches for the direct simulation Monte Carlo method for axisymmetric and three-dimensional flows. Computations using three schemes, 1) a uniform Cartesian grid, 2) a structured body-fitted grid, and 3) an unstructured tetrahedral grid, are presented for the hypersonic flow past a blunted cone for two angles of attack. The results indicate that the approaches return very similar flowfield and surface results and that these results show fair agreement with available experimental data. Moreover, it will be demonstrated that the Cartesian scheme is significantly more efficient on a per-particle basis, but yields overall solution times significantly greater than those for the other two algorithms.

Nomenclature

BCE	=	benchmark per-particle computational efficiency
c_r	=	relative speed between particles
F_{num}	=	ratio of real particles to simulated particles
N	=	number of surface-element corners, number of simulated particles in a cell
n	=	number density
T	=	temperature
V	=	cell volume, velocity magnitude
Δt_m	=	simulation time step
$\boldsymbol{\sigma}_T$	=	collision cross section

Subscripts

i	=	cell-face index
l	=	cell index, triangular surface-element subface index
m	=	cell index
w	=	surface value
∞	=	freestream value

Introduction

THE direct simulation Monte Carlo (DSMC) method of Bird[1] has become a widely used tool for the solution of thermochemical nonequilibrium flows in the transition regime. Unfortunately, the technique is considerably more expensive than conventional continuum calculations; this additional expense arises from the statistical nature of the method and the large number of simulated particles often required to achieve a meaningful solution.

Attempts to reduce the computational cost of the method have resulted in several different grid schemes, particularly in the case of three-dimensional flows. Three of the contemporary approaches are 1) a uniform Cartesian grid, 2) a structured-body-fitted grid, and 3) an unstructured, body-fitted grid. Each of these approaches has clear advantages and disadvantages; for instance, the former offers a significantly less expensive movement phase, and the latter two choices allow grid clustering in regions of large flow gradients. However, it is not immediately apparent which scheme offers the best overall performance. Moreover, the performance of each method may be dependent on the degree of rarefaction present in the flow.

This work is an attempt to quantify some of the issues previously described. Results for flowfield and surface quantities will be compared for the three grid schemes just mentioned; comparisons will also be made to experimental data. In addition, simulation performance data for each of the cases will be used to determine which approach is more efficient for the problems considered.

Flow Physics

The algorithms examined here utilize fundamentally similar physical models to help localize differences in solutions to the grid definitions. The physical processes are limited to thermal nonequilibrium through rotational and vibrational energy relaxation; chemical reactions are not included because of the relatively low stagnation temperatures for the cases examined here.

For the most part, the three codes employ models that are commonly used within the DSMC community. Bird's variable hard sphere model[1] is used to define the collision cross section, with a viscosity temperature exponent equal to 0.75. Internal energy transfer is accounted for by the Borgnakke and Larsen method,[2] with constant rotational and vibrational relaxation numbers of 5 and 50, respectively. The gas–surface interaction is modeled assuming diffuse reflection with full thermal accommodation.

Although the flow physics are quite similar, there are some discrepancies that should be noted. The first notable difference in flow physics lies in the procedures for calculating vibrational relaxation. The unstructured grid scheme utilizes the classical Borgnakke and Larsen phenomenological model[2] for inelastic collisions, implicitly assuming a continuous vibra-

Presented as Paper 96-0604 at the AIAA 34th Aerospace Sciences Meeting and Exhibit, Reno, NV, Jan. 15–19, 1996; received April 10, 1996; revision received Oct. 30, 1996; accepted for publication Dec. 16, 1996.

*Research Assistant, Mechanical and Aerospace Engineering. Student Member AIAA.

†Research Scientist, Aerothermodynamics Branch, Gas Dynamics Division. Senior Member AIAA.

‡Professor, Mechanical and Aerospace Engineering. Associate Fellow AIAA.

tional energy distribution. On the other hand, the Cartesian grid and structured, body-fitted grid algorithms use the quantum Borgnakke and Larsen method[2] proposed by Bergemann and Boyd[3] to compute vibrational energy exchange. For the flow conditions under consideration here, the degree of vibrational excitation is not great, and it is expected that there will not be a large difference in the results.

Another difference lies in the procedures used to compute the number of candidate collision pairs per cell. All three algorithms utilize Bird's no time counter method[4] for this determination; however, to maintain efficient parallel computation, the Cartesian scheme uses a slightly modified version. In the original method, the number of candidate collision pairs is given by

$$N_c = \tfrac{1}{2}\bar{N}_m n(\sigma_T c_r)_{\max}\Delta t_m \tag{1}$$

where

$$n = N_m F_{\text{num}}/V_m \tag{2}$$

Because the surface generation used in the Cartesian scheme can lead to the construction of cells whose volumes are many orders of magnitude smaller than those of neighboring cells, the previous definition of n can result in the calculation of an inordinately, and aphysically, large number of candidate collisions, leading to excessive local compute time, which can significantly reduce parallel efficiency. To remedy this problem, the Cartesian scheme instead uses a smoothed number density

$$n_s = \frac{(N_m + N_l)F_{\text{num}}}{V_m + V_l} \tag{3}$$

in cells where the density is greater than 100 times the density in adjacent cell l. This modification has no appreciable effect on the final solution. Note that, since the large density difference is driven by the large difference in cell volumes, one could equivalently base the smoothing on cell volume instead of number density.

The final difference between the algorithms is also tied to the difference in grid definitions. The Cartesian algorithm uses constant values of F_{num} and Δt_m throughout the computational domain; this convention helps reduce the overhead associated with the particle movement process. On the other hand, the other two schemes use local time-stepping; however, the value of F_{num} is also varied so that the ratio $F_{\text{num}}/\Delta t_m$ is constant throughout the flowfield, obviating the need to clone or destroy particles at cell boundaries.

Geometry Definition

Uniform Cartesian Grid

The Cartesian grid algorithm employed here is developed from Bird's DSMC3 demonstration code.[5] In this case, the body surface is not well defined within the grid. Thus, the volume grid is generated first, then the intersections between the body and the grid cells are computed. These intersections are then used to determine the properties of the surface elements and the grid cells cut by the body.

In the present Cartesian implementation, the body is represented by an analytic function of the form $f(x, y, z) = 0$. This function is used to determine which grid cells contain surface elements, as well as the intersections between the body and the grid cells.

Once the intersection points are determined, the geometry of the redefined cell (surface area, volume, normal vector) may be determined. Figure 1 shows a typical cell containing a surface element; in general, these elements will have differing numbers of sides, and need not be planar. Thus, a fairly general method is required for determining the properties of the element; such a method may be found in the work of Gaffney et al.[6]

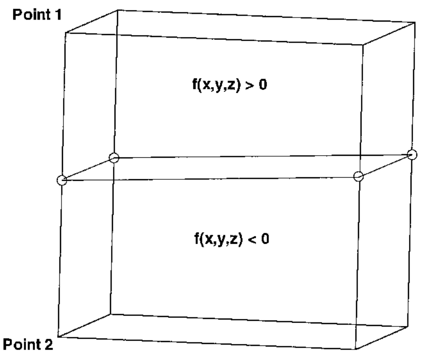

Fig. 1 Cartesian cell containing a surface element.

A surface element with more than three corners can be divided into N triangles, with N being equal to the number of corners. A center point may be defined for the face by averaging the corner points, and this center point is used to divide the face into triangles. The area of each triangle may be found through Heron's formula:

$$A_l = \sqrt{s(s - a)(s - b)(s - c)} \tag{4}$$

where a, b, and c are the side lengths of triangle l and s is equal to one-half the perimeter of the triangle. The surface element area is then

$$S = \sum_{l=1}^{N} A_l \tag{5}$$

Obviously, if there are only three corners, Heron's formula can be used to compute the overall surface element area.

We may also determine the area of a cell face cut by a surface element. For a face lying in the x-y plane

$$S = \tfrac{1}{4}[S_x S_y + S_x D_y + S_y D_x - D_x D_y] \tag{6}$$

The terms in Eq. (6) are defined as follows:

$$S_x = S_1 + S_3 \quad S_y = S_2 + S_4 \quad D_x = |S_3 - S_1| \quad D_y = |S_4 - S_2|$$

The terms are defined as shown in Fig. 2. To apply this formula, we must zero out the length of any cell that is completely contained inside the body (such as side 3 of the face shown in Fig. 3).

Once the cell face areas have been determined using the procedures listed earlier, they may be used to compute the volume of the cell. If a plane is defined by three points, denoted (x_1, y_1, z_1), (x_2, y_2, z_2), and (x_3, y_3, z_3), we may define the following quantities:

$$A = \det\begin{bmatrix} y_1 & z_1 & 1 \\ y_2 & z_2 & 1 \\ y_3 & z_3 & 1 \end{bmatrix} \tag{7}$$

$$B = -\det\begin{bmatrix} x_1 & z_1 & 1 \\ x_2 & z_2 & 1 \\ x_3 & z_3 & 1 \end{bmatrix} \tag{8}$$

$$C = \det\begin{bmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{bmatrix} \tag{9}$$

$$D = -\det\begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \end{bmatrix} \tag{10}$$

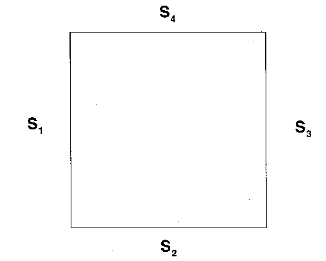

Fig. 2 Cartesian cell face side notation.

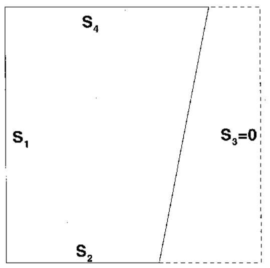

Fig. 3 Cartesian cell face with one side completely contained within the body.

The equation of the plane is then given by

$$Ax + By + Cz + D = 0 \tag{11}$$

Defining the quantity

$$F = \sqrt{(A^2 + B^2 + C^2)} \tag{12}$$

the volume of a cell with N planar faces is given by

$$V = -\frac{1}{3}\sum_{i=1}^{N}\left(\frac{D}{F}\right)_i S_i \tag{13}$$

where S_i is the surface area of each planar face.

For the surface reflection routines in the simulation algorithm, we need to know the unit normal vector for the surface element. If there are only three corner points, then the surface element is planar, and we can find the normal through

$$\boldsymbol{n} = (A/F)\boldsymbol{i} + (B/F)\boldsymbol{j} + (C/F)\boldsymbol{k} \tag{14}$$

where A, B, C, and F are given by Eqs. (7–10) and Eq. (12). However, if there are more than three points, we can compute the surface normal for each planar subface, then average the normals to get the normal for the face. It should be noted that the average normal will not necessarily have unit length; thus, the average normal is renormalized after the calculation.

Another important consideration with respect to the surface normal is that it points in the outward direction (i.e., into the flow). The correct direction is not guaranteed by the planar approximation. To ensure that the normal points in the right direction, we test the value of the function defining the surface a short distance away from the center of the surface element in the normal direction. If the function is negative, we know that the normal is pointing into the body, and we change the sign.

The Cartesian scheme is implemented on massively parallel platforms through the use of run time library support. This library, known as CHAOS,[7] was developed at the University of Maryland for use with irregular adaptive problems, of which DSMC is an example. CHAOS includes procedures for efficient communication as well as load balancing, and has been previously demonstrated to be well suited to Cartesian grid direct simulations.[8]

Structured, Body-Fitted Grid

A modified version of Bird's G2 code[1] is used for the axisymmetric results. This algorithm is well documented, and has been used for a variety of flow problems. Volume grids for this algorithm may be created using algebraic methods included as part of the preprocessing sequence, or by using commercial grid-generation software. The computational domain may be made up of contiguous regions, with each region configured for optimal resolution and simulation characteristics, a feature not possessed by the other two algorithms under study in this work.

Unstructured Grid

The unstructured grid simulation is based on Bird's G3 algorithm.[9] The original version of this code was used to compute three-dimensional flows on body-fitted, structured grids. Procedures have been added to the algorithm to permit the use of unstructured tetrahedral grids. Such grids are generated in this work utilizing the FELISA package,[10] which uses a variant of the advancing-front technique to build unstructured grids around the desired shapes. Construction of a suitable volume grid consists of 1) defining the curve segments and boundaries that form the computational domain; 2) specification of point, line, and triangle sources to control clustering; 3) generating a surface triangulation for the boundaries of the computational domain; and 4) generation of the volume grid using the surface triangulation as the initial front. Further details on the FELISA code system may be found in Ref. 10.

Results and Discussion

The two DSMC algorithms were used to compute N_2 flowfields around the 70-deg blunted cone with sting; the dimensions of the cone are shown in Fig. 4. The freestream conditions include a velocity of 1503 m/s, a number density of 3.72 $\times$ 10^{20} m^{-3}, and a temperature of 13 K; these conditions were used in experiments conducted in the SR3 low-density facility at the Centre National de la Recherche Scientifique (CNRS), Meudon, France,[11] and correspond to a freestream Knudsen number (based on overall diameter) of about 0.03. The first case is characterized by a zero angle of attack and omission of the sting from the computed solutions; the sting is included in the second case, which is conducted at $\boldsymbol{\alpha} = 10$ deg.

The grid created for the G2 solution to the first case is shown in Fig. 5, and Fig. 6 shows a representative surface grid generated for the Cartesian algorithm (hereinafter referred to simply as 3D). The former grid employs four separate regions, with a total of 120 cells along the body and 30 cells normal to the body. Although the surface grid shown in Fig. 6 includes the sting support, the solutions for case 1 omit the sting. In fact, the grid shown here is very similar to that used for case 2, except that both the top and bottom halves of the cone are included.

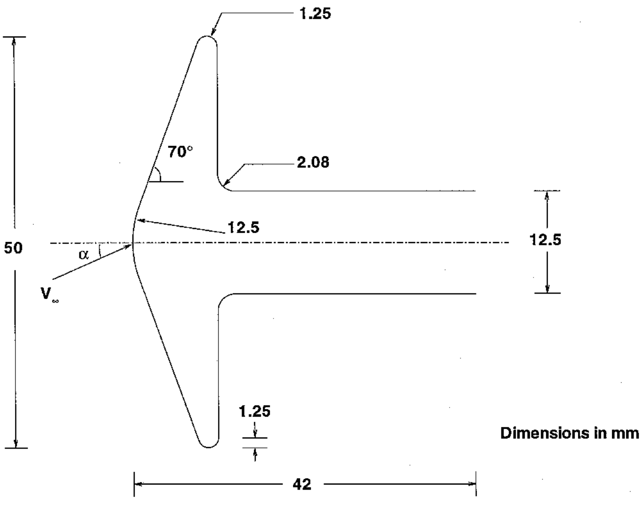

Fig. 4 CNRS blunted-cone geometry with sting.

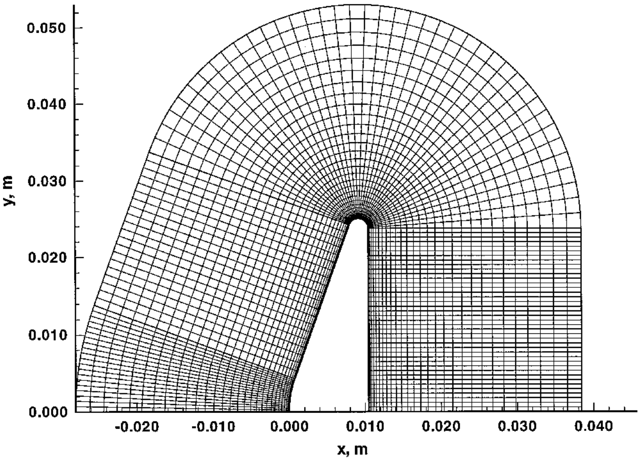

Fig. 5 G2 grid for case 1.

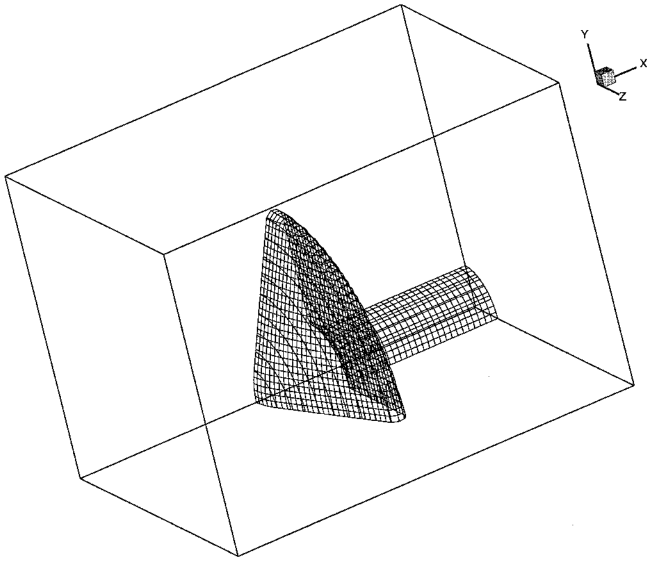

Fig. 6 Example of surface grid generated using Cartesian algorithm.

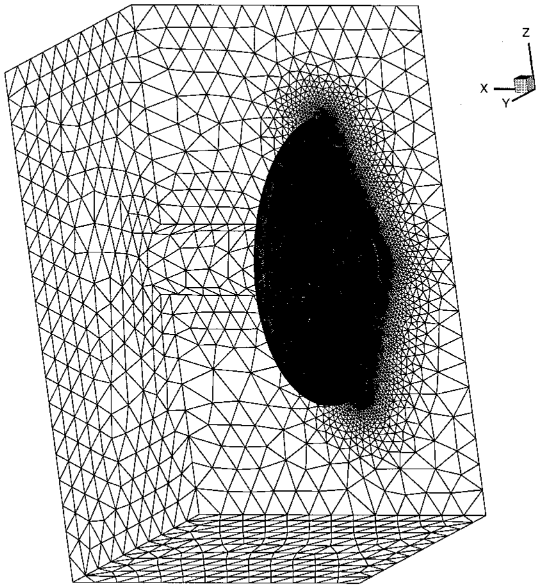

Fig. 7 G3 surface triangulation for case 2.

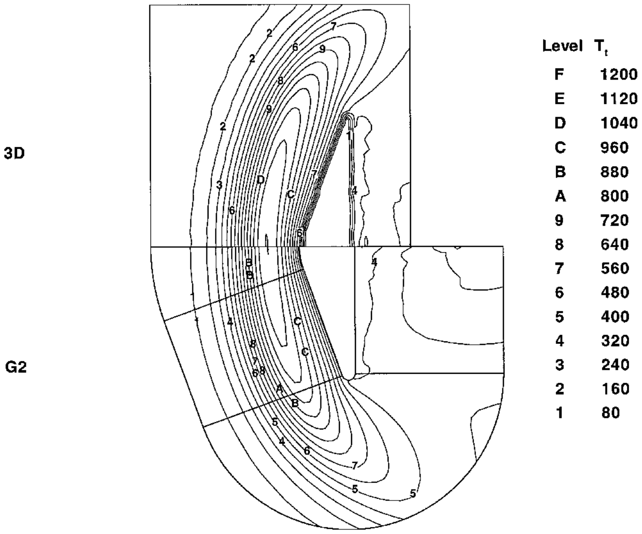

Fig. 8 Comparison of translational temperature contours for case 1.

Figure 7 shows the surface triangulation generated using the FELISA package for use with the unstructured grid algorithm (or G3) for the 10-deg angle-of-attack case. This plot illustrates that it is possible to perform clustering to resolve regions of large gradients, as well as regions of large surface curvature (such as the shoulder of the cone). Note that the computational domain for this case was constructed to be the same size and shape as that for the Cartesian solution.

Flowfield Comparisons

The translational temperature contours in the symmetry plane are compared for the first flow condition in Fig. 8. For this case, the forebody flowfield was of primary importance; since other researchers have shown[12] that the wake solution has little impact on the forebody solution, the differences in wake resolution may be considered unimportant. It may be observed that the two forebody results are virtually identical. The observed agreement between other flowfield properties was similarly good.

The experimental results presented in Ref. 11 include density flowfield visualizations obtained using an electron-beam fluorescence method for some of the cases run; Fig. 9 shows the measured density field for case 1, as well as the density results returned by the 3D algorithm for this case. It should be noted that the density visualizations were carried out with a surface temperature of about 290 K, which is different from the value of about 300 K specified for the surface heating measurements; all of the simulations were carried out with T_w = 300 K. However, it was surmised that this small difference would not preclude comparing the experimental and computational results. The plot shows that the agreement between the experimental data and the DSMC solution is fairly good. There are some discrepancies, for instance, the shock-layer thicknesses are somewhat different and the experimental shoulder expansion is somewhat more diffuse than that predicted by the simulation, but overall the results are quite similar. (Because of the very high degree of similarity between the G2 and 3D results, the G2 density field is not shown here.)

Translational temperature results for the second case are shown in Fig. 10. As with the zero angle-of-attack results, we see that the computations agree very well with one another throughout the flowfield. However, the G3 solution does show more noise than the 3D solution; this increased scatter is a consequence of the relatively low average number of particles per computational cell in the G3 simulation. We may also compare density results for the second case; the experimental visualization for a portion of the flowfield is shown in Fig. 11; whereas Figs. 12 and 13 show the contours predicted by the

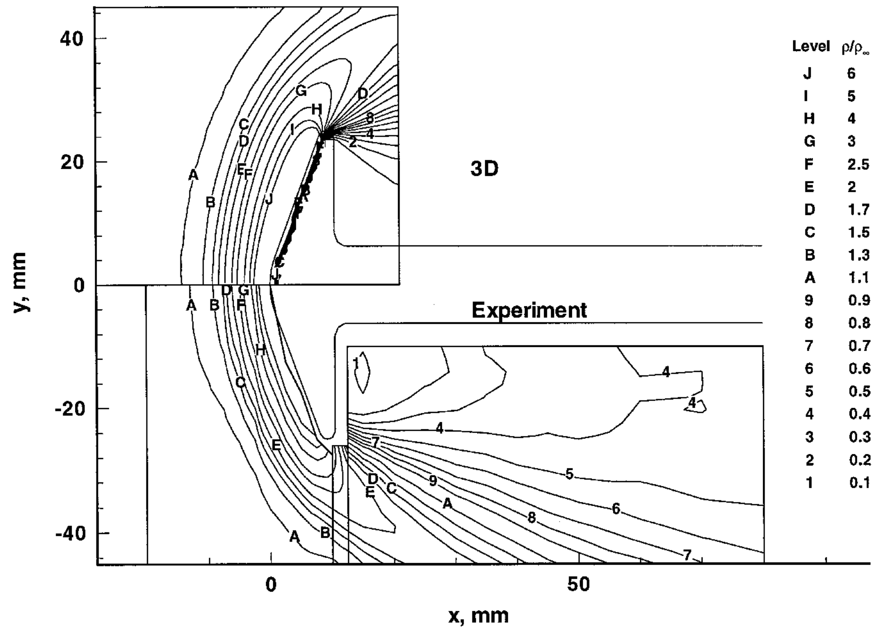

Fig. 9 Comparison of experimental and computed (3D) density fields for case 1.

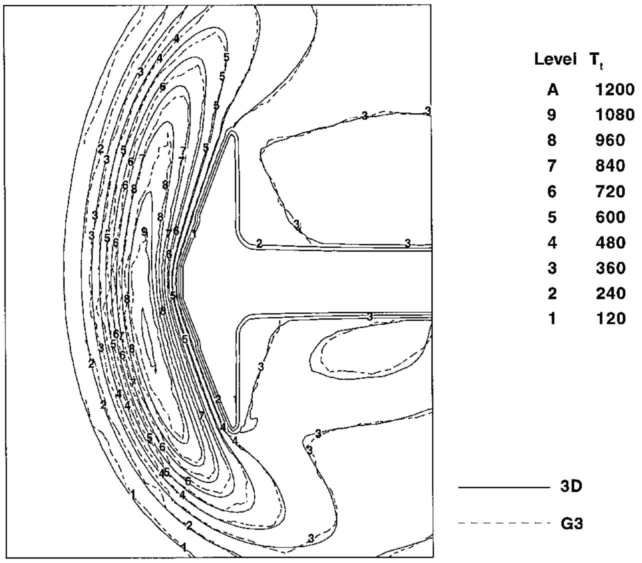

Fig. 10 Comparison of translational temperature contours for case 2.

3D and G3 algorithms, respectively. These plots show that the agreement between the two DSMC solutions is again quite good, although the noise in the G3 solution is again evident. Note the presence of the two circular structures in the near-wake region of the 3D solution. These artifacts are a consequence of extremely small cells in the base-sting juncture region, as well as the plotting package used. Both computed density fields show fair agreement with the experimental data (the same contour levels are used for all three plots), with the same types of discrepancies as observed for the zero angle-of-attack case.

Surface Comparisons

For this portion of the report, we shall restrict our attention to observations of heating rates in the symmetry plane, since no experimental data are available for other surface quantities. The heat transfer results for the first case are plotted vs the linear surface distance and compared to the measured data in Fig. 14. We can see that the two DSMC results lie virtually on top of each other, although the G2 heating distribution is somewhat smoother than that returned by the 3D algorithm. This difference is almost certainly because of differences in grid structure and sample size. The plot also shows that the predicted solutions agree very well with the experiment near the stagnation point; however, the agreement decreases rapidly as we move along the forebody, with a maximum error of about 28%. This difference is certainly significant, since the published uncertainty for the measurements is only 5%. However, other researchers have reported similar levels of error between calculations and the data from these experiments.[13] As indicated in Ref. 13, these discrepancies remain unresolved, but may be tied to wall conduction effects and upstream flow nonuniformities in the experiments.

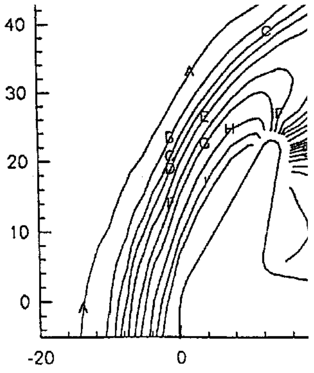

Fig. 11 Experimental density field in symmetry plane for case 2.

Figures 15 and 16 show the computed symmetry-plane heating rates along the windward surface for case 2, as well as experimental measurements from the CNRS facility. The first of these plots shows that the two computations agree reasonably well, with the largest discrepancies at the nose and shoulder. Figure 15 shows that the level of disagreement between the forebody solutions and measured heat fluxes is about the same as in the first case, with the difference lying in the 10–20% range. Note further that both DSMC solutions show comparable amounts of scatter, with both results being relatively smooth.

The heating results for the base plane and sting (Fig. 16) show somewhat better agreement with the experimental measurements in this region. Note that the measured heating rates along the base plane for both cases considered here were so small that the experimental values could only be characterized

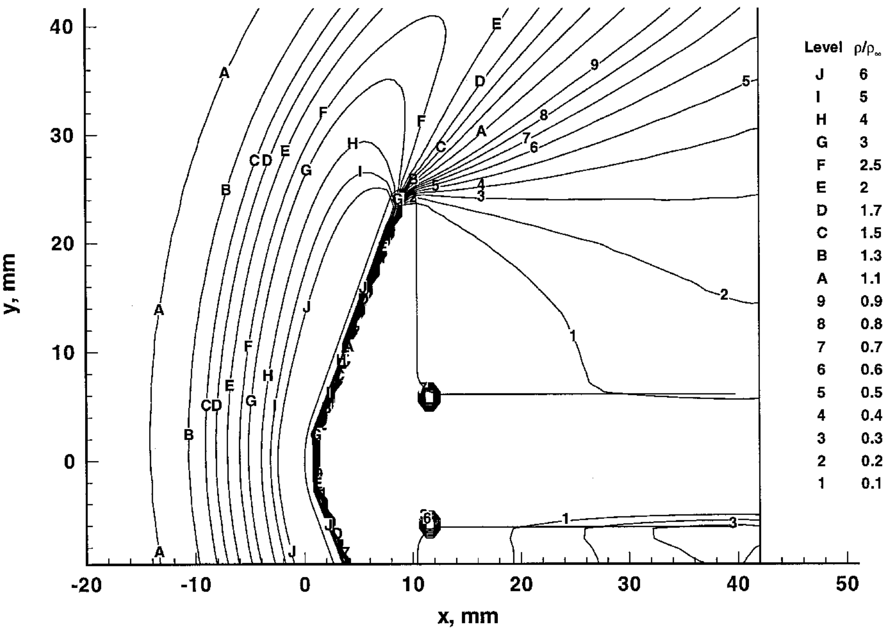

Fig. 12 3D density field for case 2.

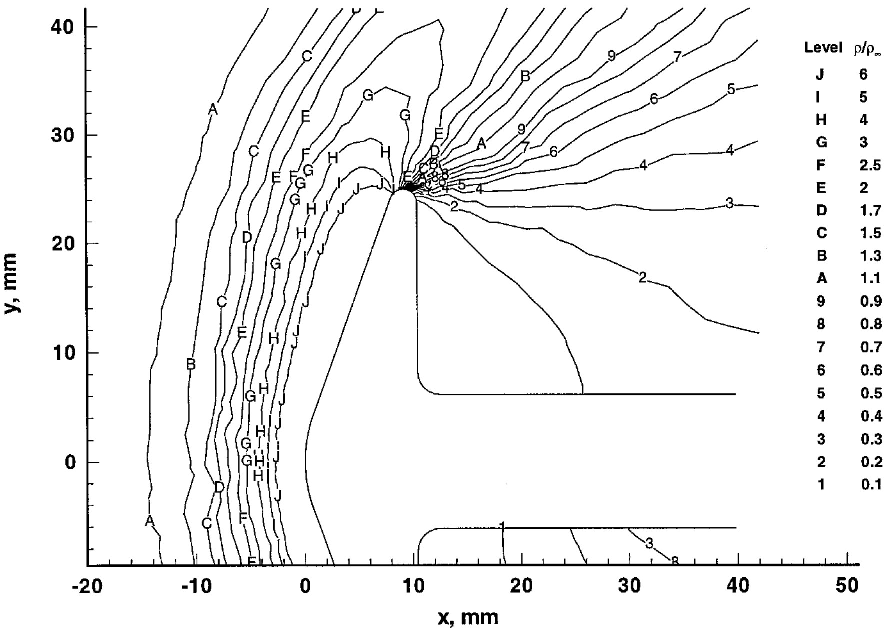

Fig. 13 G3 density field for case 2.

as being less than some upper bound. Hence, the apparent underprediction of the base-plane heating actually indicates that both computed results are well below the specified upper bounds (points marked with arrows). This plot also shows that the DSMC results along the base plane contain significantly more scatter than those for the forebody; again, this phenomenon may be attributed to small sample size and the fact that both grids overresolve the wake region. Such overresolution is unavoidable in the Cartesian solution; additionally, the wake cell size in the unstructured grid cannot be increased further without underresolving the sting surface.

Performance Characteristics

The 3D algorithm was run in parallel on the NASA Langley Research Center's Intel Paragon, with 40 nodes used for case 1 and 55 nodes utilized for case 2. However, the G2 and G3 codes were run on single processor workstations (Sun SPARCstation 2 and Sun SPARCstation 10, respectively). Timing comparisons between these simulations would be meaningless without a benchmark calculation. To this end, the original DSMC3 code was run on the workstations as well as on one processor of the Paragon. Computational efficiencies for these benchmark runs were used to rescale the SPARCstation 2 and Paragon timing results to be directly comparable with timings on the SPARCstation 10. This rescaling is accomplished by multiplying by the appropriate number of processors and the factor

$$(BCE)_{\mathrm{SPARC\,10}}/BCE$$

where *BCE* is the benchmark computational efficiency for the architecture being considered.

Before examining the timing results, let us look at some of the simulation parameters listed in Table 1. Note that the G2 solution for case 1 utilizes substantially fewer cells than the 3D solution, a consequence of grid clustering and the axisymmetric solution technique. However, the G3 solution actually

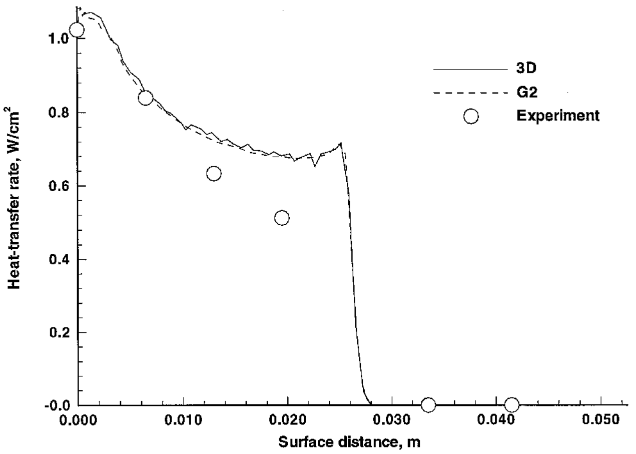

Fig. 14 Comparison of heating results for case 1.

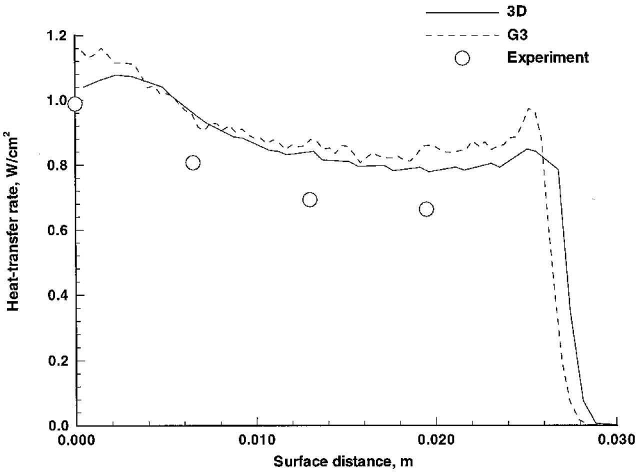

Fig. 15 Comparison of forebody heating results for case 2.

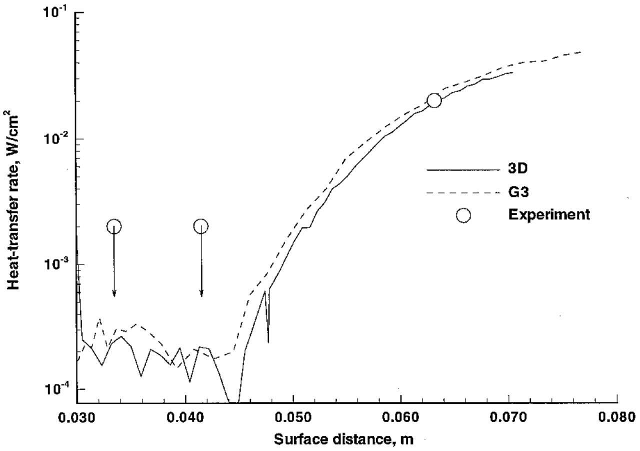

Fig. 16 Comparison of base plane and sting heating results for case 2.

Table 1 Comparison of simulation characteristics

Code	Case 1	Case 2
	Number of cells	
G2	3600	——
G3	——	611,143
3D	175,000	500,000
	Average number of simulated particles	
G2	144,000	——
G3	——	502,000
3D	2,540,000	6,500,000

Table 2 Comparison of performance characteristics

Code	Case 1	Case 2
G2 (SPARC 2)	100	——
G2 (scaled to SPARC 10)	44	——
G3 (SPARC 10)	——	106
3D (Paragon)	1.20 (40 nodes)	0.90 (55 nodes)
3D (scaled to SPARC 10)	26	27

Table 3 Overall run times for 100 steady-state prints

Code	Case 1	Case 2
G2 (SPARC 2)	40 h	——
G2 (scaled to SPARC 10)	18 h	——
G3 (SPARC 10)	——	147 h
3D (Paragon)	9.5 h	21 h
3D (scaled to SPARC 10)	208 h	633 h

uses more cells than the 3D simulation for case 2, and employs much fewer simulated particles because of memory restrictions. The low number of simulated particles is directly responsible for the noise in the G3 results mentioned earlier.

Table 2 presents per-particle computational efficiencies for each case run; several observations may be made about these results. First of all, the parallel algorithm is much faster on an absolute per-particle basis than the scalar codes, as might be expected. Once the parallel efficiencies are scaled, however, it becomes apparent that the Cartesian scheme is still faster per particle than the other two algorithms. Secondly, the G3 algorithm can be seen to be substantially more expensive per particle than the G2 scheme. This increase may be ascribed to two factors: 1) the fact that the G3 scheme must deal with an additional spatial dimension and 2) a consequence of the unstructured grid movement procedure. The latter pertains to the more frequent crossing of cell boundaries in an unstructured grid than in a structured one, which complicates and slows down the particle movement process.

The computational efficiencies, however, do not really tell us how much time is required to obtain a meaningful solution, nor which grid approach will return a solution faster. Table 3 lists run times for 100 steady-state print cycles (5000 steady-state samples) to help make these determinations. The scaled results indicate that, although the 3D code is substantially faster in a parallel environment, it is much slower than the body-fitted algorithms when confined to a single processor. These results imply that the Cartesian algorithm may benefit greatly from the development of a multiregion strategy similar to that employed by the G2 code. Alternately, the results also suggest that development of massively parallel body-fitted-grid DSMC algorithms is certainly a worthwhile pursuit.

Concluding Remarks

Comparisons of flowfield and surface results for the three DSMC grid approaches were shown to be quite favorable, with the G3 flowfield solutions showing slightly more scatter than the G2 and 3D results. The discrepancies between the calculated results and experimental data are fairly large, but commensurate with those reported elsewhere.

Performance results for the three algorithms indicate that the Cartesian approach is substantially faster on a per-particle basis. However, this advantage is offset by the need for using many more particles than in the body-fitted grid simulations.

As a result, the Cartesian scheme appears to require more time than the body-fitted-grid simulations (on an equivalent platform) to achieve the same number of steady-state samples.

Acknowledgments

This work is supported in part by NASA Cooperative Agreement NCC1-112, the Mars Mission Research Center funded by NASA Grant NAGW-1331, and a National Defense Science and Engineering Graduate Fellowship. Computer resources were provided by NASA Langley Research Center.

References

[1] Bird, G. A., "Monte Carlo Simulation in an Engineering Context," *Rarefied Gas Dynamics*, Vol. 74, Pt. 1, Progress in Astronautics and Aeronautics, AIAA, New York, 1981, pp. 239–255.

[2] Borgnakke, C., and Larsen, P. S., "Statistical Collision Model for Monte Carlo Simulation of Polyatomic Gas Mixtures," *Journal of Computational Physics*, Vol. 18, No. 3, 1975, pp. 405–420.

[3] Bergemann, F., and Boyd, I. D., "New Discrete Vibrational Energy Model for the Direct Simulation Monte Carlo Method," *Rarefied Gas Dynamics: Experimental Techniques and Physical Systems*, edited by B. D. Shizgal and D. P. Weaver, Vol. 159, Progress in Astronautics and Aeronautics, AIAA, Washington, DC, 1994, pp. 174–183.

[4] Bird, G. A., "Perception of Numerical Methods in Rarefied Gasdynamics," *Rarefied Gas Dynamics: Theory and Computational Techniques*, edited by D. P. Weaver and D. H. Campbell, Vol. 118, Progress in Astronautics and Aeronautics, AIAA, Washington, DC, 1989, pp. 211–226.

[5] Bird, G. A., *Molecular Gas Dynamics and the Direct Simulation of Gas Flows*, Clarendon, Oxford, England, UK, 1994.

[6] Gaffney, R. L., Jr., Salas, M. D., and Hassan, H. A., "Euler Calculations for Wings Using Cartesian Grids," AIAA Paper 87-0356, Jan. 1987.

[7] Hwang, Y. S., Moon, B., Sharma, S., Ponnusamy, R., Das, R., and Saltz, J., "Runtime and Language Support for Compiling Adaptive Irregular Problems on Distributed Memory Machines," *Software Practice and Experience*, Vol. 25, No. 6, 1995, pp. 597–621.

[8] Nance, R. P., Wilmoth, R. G., Moon, B., Hassan, H. A., and Saltz, J., "Parallel Monte Carlo Simulation of Three-Dimensional Flow over a Flat Plate," *Journal of Thermophysics and Heat Transfer*, Vol. 9, No. 3, 1995, pp. 471–477.

[9] Celenligil, M. C., and Moss, J. N., "Three-Dimensional Hypersonic Rarefied Flow Calculations Using Direct Simulation Monte Carlo Method," *Theoretical and Experimental Methods in Hypersonic Flows*, CP-514, AGARD, April 1993, pp. 27-1–27-11.

[10] Peraire, J., Morgan, K., and Peiro, J., "Unstructured Finite Element Mesh Generation and Adaptive Procedures for CFD," *Application of Mesh Generation to Complex 3-D Geometries*, CP-464, AGARD, 1990, pp. 18.1–18.12.

[11] Allegre, J., and Bisch, D., "Experimental Study of a Blunted Cone at Rarefied Hypersonic Conditions," RC 94-7, Centre National de la Recherche Scientifique, Meudon, France, Nov. 1994.

[12] Olynick, D. P., Moss, J. N., and Hassan, H. A., "Influence of Afterbodies on AOTV Flows," AIAA Paper 89-0331, Jan. 1989.

[13] Moss, J. N., Price, J. M., Dogra, V. K., and Hash, D. B., "Comparison of DSMC and Experimental Results for Hypersonic External Flows," AIAA Paper 95-2028, June 1995.