An Object-Oriented Serial DSMC Simulation Package

Hongli Liu and Chunpei Cai

Department of Mechanical and Aerospace Engineering, New Mexico State University, Las Cruces, New Mexico, 88003, USA

Abstract. A newly developed three-dimensional direct simulation Monte Carlo (DSMC) simulation package, named GRASP ("Generalized Rarefied gAs Simulation Package"), is reported in this paper. This package utilizes the concept of simulation engine, many C++ features and software design patterns. The package has an open architecture which can benefit further development and maintenance of the code. In order to reduce the engineering time for three-dimensional models, a hybrid grid scheme, combined with a flexible data structure compiled by C++ language, are implemented in this package. This scheme utilizes a local data structure based on the computational cell to achieve high performance on workstation processors. This data structure allows the DSMC algorithm to be very efficiently parallelized with domain decomposition and it provides much flexibility in terms of grid types. This package can utilize traditional structured, unstructured or hybrid grids within the framework of a single code to model arbitrarily complex geometries and to simulate rarefied gas flows. Benchmark test cases indicate that this package has satisfactory accuracy for complex rarefied gas flows.

Keywords: DSMC implementation, hybrid grid
PACS: 47.45.-n

1. INTRODUCTION

THE spaceport program in the New Mexico state demands for more research and education on space weather and hypersonic flows. As one part of the new aerospace engineering program in the New Mexico State University, we are developing a new set of compressible gas/plasma simulation packages, to serve as the education and research platforms for rarefied gas and plasma flows, such as those associated with hypersonic re-entry flows and space weather. This package is named GRASP ("Generalized Rarefied gAs Simulation Package").

For the DSMC method, there are several established implementations, including those educational programs by Bird [1]. Several other research groups developed different DSMC packages, such as SMILE [2], MONACO [3], and DAC [4]. In this paper, we will present the GRASP code architecture, data structure, a three dimensional simulation meshing method and other preliminary implementations of this object-oriented package.

Like most numerical methods, grid scheme plays an important role in efficient and accurate predictions by the DSMC method. Generally in GRASP, unstructured mesh systems are used for two-dimensional and axi-symmetric simulations to guarantee the program's stability and precision. For three-dimensional case, GRASP divides the computational domain into uniform cubic solid grids to track molecular trajectories efficiently, whereas the object surface is triangulated by any major CAD/CAE software and read in by the package. As such, GRASP is not only much efficient with unstructured meshes, but also capable of representing the objects' surface details with high precision. During the simulation process, most of simulated molecules are efficiently tracked within a cartesian coordination system, while the small region adjacent to the object surface is treated delicately like in an unstructured grid scheme.

2. THE DATA STRUCTURE

One important concept that GRASP adopts is "simulation engine". For DSMC simulation packages, usually there are two levels: cells and particles. However, only these two levels can not guarantee the package as good architecture for further development and can not fully utilize the advantages of C++ language. As a common major defect for different dimensions, different files and subroutines or conditional compilations are necessary in many DSMC packages. We introduce several classes of simulation engines which have internal inherence relations, as such two-dimensional, axi-symmetric, and three-dimensional simulation codes can coexist in the same package, and the usage of conditional compilation, "#ifdef" can be reduced to a minimum. Figure 1 shows the internal relations for several engine classes.

There are two popular particle storage data structures. One is described in the book by Bird [1], to utilize a large whole array storing information for all the particles; the other approach is to utilize linked lists inside each cell, i.e., to divide the single complete particle table into many small linked lists for each cell. In GRASP, we adopt the single large particle array [1], as such, less attention is needed since changing particle's location from one cell to another only requires a cell id change for the particle. The particle table in GRASP is resizable when more particles are needed, but currently the particle table is full. The particle table class can automatically apply for a double-sized memory chunk in the free store memory area, copy the current particle table to the larger memory chunk and release the memory storage for the smaller table. The memory for the mesh table is fixed. Figure 2 illustrates the two-dimensional engine class that contains two container classes for the particle and mesh.

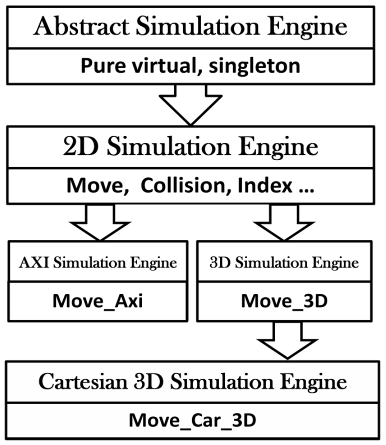

FIGURE 1. Relations of various simulation engine classes in GRASP-P1.0.

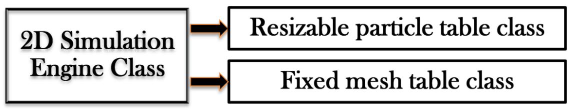

FIGURE 2. Resizable particle table and fixed mesh table inside a 2D simulation engine class.

GRASP-P1.0 is written in C++ language and it utilizes many special design patterns. One major consideration to introduce the concept of simulation engine is to resolve the architecture and interface conflict for future development. One class solely regulates the type of interfaces that it should consider, while its daughter classes, such as two-dimensional simulation engine class, concentrates on special function bodies definition. The other further derived classes from the daughter class concentrate on special function bodies to override. As such, GRASP has an open architecture, and we can add different simulation methods into this single package. Special design pattern, such as singleton [5] is implemented for the simulation engine class, which provides good code accessability.

The class concept in C++ is also used to package different elementary data variables and functions together into one data structure representing a DSMC cell. Its features, such as encapsulation and inheritance, can help to maintain the code in terms of scalability, portability, and commonality. For a 3D simulation situation, there are many space structures (such as cells) similar to cubic boxes. Many three-dimensional operations of GRASP-P1.0, such as particles' motions through the cells, collisions with the wall, are incorporated inside the boundary box class.

3. THE HYBRID MESH SYSTEM FOR 3D SIMULATIONS

For engineering applications, three-dimensional problems are of special interest because they can represent the flow fields more accurately than two-dimensional simplifications. However, to generate the meshes for three-dimensional problems with CAD software usually requires much engineering effort. In GRASP-P1.0, we implement a hybrid mesh method which requires simpler input of a triangulated object surface, while the real three dimensional cubic meshes are generated internally without user interactions. This is achieved with a special simulation engine class “Cart3D simulation engine”. As mentioned above, simulated molecules interact with walls many times. It is an important process in the DSMC simulation to handle the interactions between particles and the body surfaces accurately. In the GRASP-P1.0 three-dimensional scheme, to find the relationship of the wall and the space grids is one of the key pre-processing steps. There is a procedure which can directly and accurately identify and record body surfaces’ distribution within the three-dimensional cubic cells [6]:

1). Triangulate the object body surfaces freely and record their geometry information. Any major CAD software packages can accomplish this task; and output the results for GRASP to read in. This is the only external pre-processing step required by GRASP;

2). Automatically determine the large simulation domain. For example, the outer simulation domain of a spacecraft is automatically determined by offsetting the bounding box of the spacecraft;

3). Divide the whole computational domain into uniform cubic solid cells;

4). Identify the surface triangles and the spatial 3D cubic cells by two series of numbers; and if a body surface triangle is related to one cubic cell, record this relationship.

5). Record the numbers and the serial numbers of the surface triangles related to the current cell, and book the related cells with every surface triangle one by one. In GRASP, these records are stored in two fixed size arrays.

Steps 3,4,5 are accomplished by one C++ class. Obviously by automatically generating the three-dimensional mesh, this can reduce the engineering time for the whole simulation. At the same time, utilizing a cubic mesh can significantly increase the simulation speed, because the module of particle movement is greatly simplified.

4. TEST CASES

To demonstrate the capability of the GRASP package, we include two test cases in this paper. We compare the simulation results with those obtained by another established DSMC package or analytical results. The object surfaces in the test cases are assumed to be fully diffuse here.

4.1. Hypersonic Flow over a Wedge

The first test case is a hypersonic flow around a 40° wedge with a 10° angle of attack. As such, the top side of the wedge is equivalent to a 10° wedge and the bottom side is equivalent to a 30° one. The temperature of free stream argon gas flow is 200 K, the Mach number is 10, and the number density is 1.3×10^{20} m^{-3}. The wedge wall temperature is 300 K. The free stream Knudsen number is 0.05, the characteristic length of this case is the wedge base length.

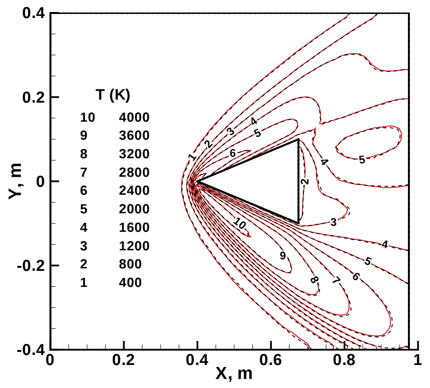

FIGURE 3. Temperature contours for a hypersonic flow over a wedge, Ma=10. Unit: K. Solid line: GRASP; dashed line: MONACO

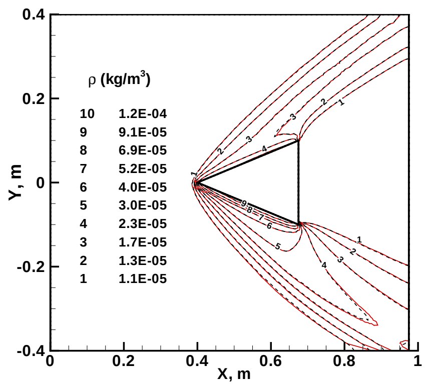

FIGURE 4. Density contours for a hypersonic flow over a wedge, Ma=10. Unit: kg/m^3. Solid line: GRASP; dashed line: MONACO

Figures 3 and 4 show that GRASP-P1.0 and MONACO predict virtually identical results for the temperature and density contours around the wedge, two different shock waves with different shapes at the front sides and the expansion waves at the rear side display clearly. The solid lines represent the GRASP-P1.0 simulation results, while the dashed lines are the MONACO results.

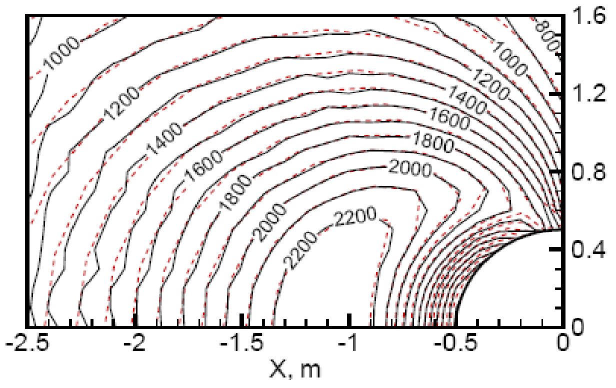

FIGURE 5. Temperature contours for a hypersonic flow over a sphere, Ma=5, Unit: K.

4.2. Hypersonic Free Molecular Flow over a Sphere

The next test case is a hypersonic free molecular flow over a sphere, simulated with GRASP-P1.0 three dimensional simulation engine with hybrid grids. The temperatures of the free stream argon gas flow and the sphere wall are 300 K. The free stream number density at inlet is 1.0×10^{20} m^{-3}. The radius of the sphere is 0.5 m. The purpose of this example is to validate the 3D particle movement module of the package, and compare some surface properties.

The cartesian grid system is utilized in all of the flow fields. The simulation volume is a cubic zone, and at the center is the object region representing a sphere, while the other six external surfaces are obtained by offsetting the bounding box covering the internal sphere. The offset values are regulated by input from a file, as such, adjusting the simulation domain and generating the internal mesh are highly automatic- it can be achieved by specifying or altering a parameter in the file, and the corresponding engineering time to solve this problem is thus significantly reduced.

Figure 5 shows the temperature distributions in the middle plane, to better illustrate the comparison, only the front side contours are compared. The solid lines represent the GRASP-P1.0 simulation results, while the dashed lines are the analytical results for collisionless flow over a sphere [7]. As we can see, even though the flow is collisionless, due

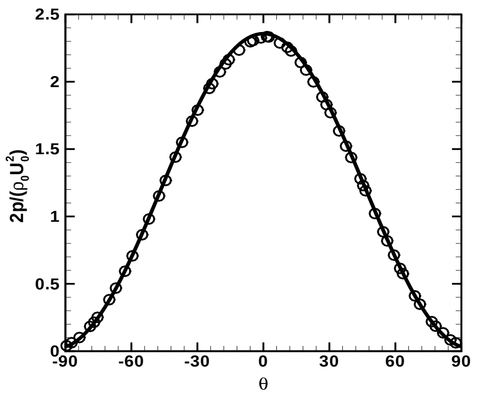

FIGURE 6. Normalized pressure distribution on sphere surface.

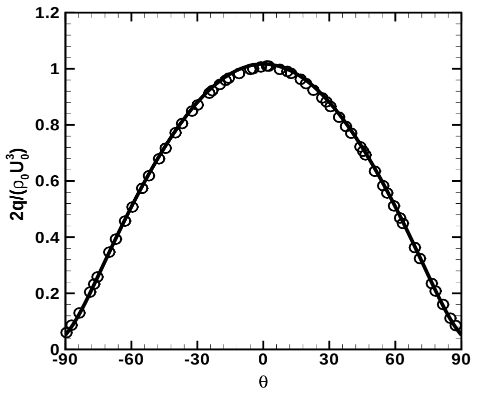

FIGURE 7. Normalized heat flux distribution on sphere surface.

to the high Mach number, in front of the sphere, the highest temperature can reach over 2200 K. It is also evident that for the 3D simulation, GRASP-P1.0 results have very good agreement with the analytical results. Figures 6 and 7 are comparisons of sphere surface pressure and heat flux results obtained by GRASP-P1.0 and the corresponding analytical results. It is evident that the three-dimensional simulation and analytical results are very close.

5. CONCLUSIONS

We have reported our development of a new object-oriented DSMC simulation package, GRASP-P1.0. This package is developed utilizing C++ and several design patterns, such as inheritance and singleton. We introduced the concept of simulation engine and included ten major classes in the package. The simulation engine concept successfully solves the conflicts between functionalities and interfaces. The architecture of GRASP-P1.0 is open for further development and is maintenance friendly, and we achieve two-dimensional, axisymetric, and three-dimensional simulation methods in a single package, and reduce the uses of conditional compilation to a minimum.

Another novel feature of GRASP-P1.0 is that we implemented the cartesian grids with C++ into the package as a special class. For reentry flow simulations, only triangulated object surface geometry is needed, and other major control information is provided through a card file. The 3D mesh is generated internally rather than externally, as such the engineering time for the simulation can be greatly reduced. For example, changing the outer domain size can be achieved conveniently by changing one input card, and no 3D mesh generation is required. By utilizing cubic cells, the particle movement module is greatly simplified as well.

ACKNOWLEDGMENTS

We acknowledge the partial financial supports from New Mexico State and ZONA Technology for the development work on GRASP-P1.0.

REFERENCES

1. G. A. Bird, *Molecular Gas Dynamics and the Direct Simulation of Gas Flows*, Oxford University Press, New York, 1994.
2. M.S. Ivanov, G.N. Markelov, and S.F. Gimelshein, “Statistical Simulation of Reactive Rarefied Flows: Numerical Approach and Applications”, *AIAA Paper*, 98-2669, 1998.
3. S. Dietrich, and I. D. Boyd, *Journal of Computational Physics*, Vol. 126, 1996, pp.328-342.
4. G.J. LeBeau, “A Parallel Implementation of the Direct Simulation Monte Carlo Method”, *Computer Method in Applied Mechanics and Engineering*, Vol. 174, 1999, pp.319-337.
5. E. Gamma, R. Helm, R. Johnson, and Vlissides, J., *(Addison-Wesley Profesional Computing Series) Design Paterns: Elements of Reusable Object-Oriented Software,* Addision-Wesley, Boston, 1995.
6. H. Liu, J. Fan, and C. Shen, “Validation of a Hybrid Scheme of DSMC in Simulating Three-Dimensional Rarefied Gas Flows”, CP663, *Rarefied Gas Dyanmics*: 23rd International Symposium, Edited by Ketsdever and Muntz.
7. C. Cai, K. R. Khasawneh, H. Liu, and M. Wei, *Journal of Spacecraft and Rockets*, Vol.46, No.6, November-December, 2009.