Available online at www.sciencedirect.com
SCIENCE DIRECT®
Computer Physics Communications 162 (2004) 166–187
Computer Physics Communications
www.elsevier.com/locate/cpc

Parallel three-dimensional DSMC method using mesh refinement and variable time-step scheme

Jong-Shinn Wu *, Kun-Chang Tseng, Fu-Yuan Wu

Department of Mechanical Engineering, National Chiao-Tung University, Hsinchu 30050, Taiwan

Received 21 May 2004; accepted 3 July 2004

Abstract

Application of variable time-step and unstructured adaptive mesh refinement in parallel three-dimensional Direct Simulation Monte Carlo (DSMC) method is presented. A variable time-step method using the particle fluxes conservation (mass, momentum and energy) across the cell interface is implemented to reduce the number of simulated particles and the number of iterations of transient period towards steady state, without sacrificing the solution accuracy. In addition, a three-dimensional *h-refined* unstructured adaptive mesh with simple but effective mesh-quality control, obtained from a preliminary parallel DSMC simulation, is used to increase the accuracy of the DSMC solution. Completed code is then applied to compute several external and internal flows, and compared with previous results wherever available.

Keywords: Monte Carlo method; DSMC; Variable time-step; Unstructured adaptive mesh refinement; Mesh-quality control

1. Introduction

1.1. Rarefied gas dynamics

In some flow regimes, the Navier–Stokes equations fail to approximate the gas dynamics behavior and the particle nature of the matter must be taken into account. One of these is the rarefied gas flow, which the mean free path becomes comparable with, or even larger than, the characteristic length of flow. Applications of high Knudsen number flows are now of practical scientific and engineering importance [1]. Boltzmann equation is generally considered as the governing equation for all flow regimes, but either the direct analytical or numerical solution of Boltzmann equation is very difficult to obtain for practical flows. An alternative method, known as Direct Simulation Monte Carlo (DSMC), was proposed by Bird [1,2] to compute the hypersonic flows under rarefied conditions. Later, Nanbu [3] was able to demonstrate mathematically that the DSMC method is equivalent to solving the Boltzmann equation as the simulated particle numbers become large. This method has become a widely

* Corresponding author.
E-mail address: chongsin@faculty.nctu.edu.tw (J.-S. Wu).

0010-4655/$ – see front matter
doi:10.1016/j.cpc.2004.07.004

used computational tool for the simulation of gas flows in which molecular effects become important. Further applications of the DSMC method included gas flows around spacecraft [4,5], pumping characteristics of high vacuum pump [6,7], conductance computation of an orifice [8], the slider air bearing of the computer hard disk [9] and, recently, the micro-scale gas flows [10–12], among others.

The basic idea of DSMC is to calculate practical gas flows through the use of no more than the collision mechanics in a probabilistic way. The simulated molecules move in the computational domain so that the physical time is a parameter in the simulation and all flows are computed as unsteady flows. An important feature of DSMC is that the molecular motion and the intermolecular collisions are uncoupled over the time intervals that are much smaller than the mean collision time. Both the collision between molecules and the interaction between molecules and solid boundaries are computed on a probabilistic basis. Hence, this method makes extensive use of random numbers. In most practical applications, the number of simulated molecules is extremely small as compared with the number of real molecules. The macroscopic quantities such as mean velocities and temperatures are sampled and averaged from cells. The details of the procedures and the consequences of the computational approximations can be found in Bird [1,2] and are not repeated here for brevity.

1.2. Variable time-step method

In addition to the parallel implementation of the DSMC method [13–17], the other strategy to increase the computational speed is to reduce the number of simulated particles by varying particle or cell weighting in the computational domain, while maintaining relatively uniform particle distribution per cell, e.g., radial weighting for axisymmetric flow [1,2], particle weighting [18] and variation of time-steps [18,19]. It can be shown that careless use of cell or particle weighting often introduces some detrimental effects to the statistical accuracy, which is caused by repeatedly cloning the particles in the flow field [18]. Ivanov et al. [19] proposed a variable time-step method that divides the computational domain into four subdomains having single time-step in each sub-domain. This results the reduction of number of simulation particles. Thus, the variable time-step method may represent one of the simplest and most efficient ways of particle weighting that avoids the problem of particle cloning, if careful grid manipulation is done [18]. Not only does it satisfy the fluxes conservation (mass, momentum and energy) exactly when a simulated particle moves across cell interface during simulation, but also it reduces tremendously the number of iterations required for transient period towards steady state. It is thus appropriate to use the variable time-step in a highly varying size of mesh system, resulting from the solution-based adaptive process, which is the objective in the current study.

1.3. Unstructured adaptive mesh

The DSMC method requires the introduction of computational cells (meshes) similar to those in CFD, while the cells are mainly used for selecting collision partners, sampling and averaging the macroscopic flow properties. Many physical problems involve very complicated geometry of objects; thus, unstructured mesh has been recommended to take advantage of the flexibility of handling this situation [2,10,12]. In addition, using unstructured mesh has the flexibility of applying graph-partitioning technique for parallel implementation of the DSMC method [16,21,22]. In principle, the size of cell used in DSMC method have to adapt according to the local density variation (i.e., less than the local mean free path) or have to be refined near the body surface to obtain accurate prediction of pressure, friction and heat transfer. However, these are not known *a priori* in general. For flow field having highly non-uniform density variation, generation of an appropriate mesh often becomes a very demanding and time-consuming task. The appropriate mesh used for final computation is often obtained through trial-and-error. Previously, the unstructured adaptive mesh has been suggested as a better solution to the above difficulties [18–25]. Thus, an efficient way to adapt the unstructured mesh to the DSMC solution is required to obtain an accurate DSMC solution under such circumstances.

Among the very few studies along this line, Boyd's group [4,13,18] have developed a parallel DSMC software named *MONACO*, which incorporates the unstructured mesh refinement to make sure the cell size less than the local mean free path. For two-dimensional simulation, if the cell needs to be re-

fined, a node is simply added at the centroid to divide it into several subcells. After cell refinement, an edge swapping method (Delaunay triangulation) is applied to improve the cell quality. Three-dimensional computational meshes are generated by a commercial code, named *FELISA* [20], which also has a capability for generating and adapting unstructured tetrahedral mesh.

In addition, Ivanov's group [19] have proposed a parallel DSMC code named *SMILE* with adaptive grid refinement, which employs a Cartesian grid consisting of uniform background cells. Its main advantage is a simple and effective indexing of particle in cells to reduce the computational time. The linear dimensions of embedded cells depend on the magnitude of local density. The adaptation of linear dimensions is conducted in three directions depending on the density gradient that provides a high spatial resolution in areas of strong flow gradients.

Harvey's group [21,23] have applied a solution-based, two-dimensional re-meshing adaptive grid technique (mesh regeneration using the advancing front method) in unstructured mesh to study the hypersonic flow field with highly non-uniform density variation, involving shocks and expansion waves. However, some unexpected results such as lower accuracy for a refined mesh, as compared with a coarse mesh, arose due to smaller particle-per-cell caused by too many cells in the flow field.

LeBeau et al. [24] also proposed a parallel three-dimensional DSMC Analysis Code (*DAC*) that can be applied to a broad range of low-density flow problems. A 2-level embedded Cartesian grid system, that is completely uncoupled from the surface representation, is automatically generated by the software to relieve the user of this traditionally time-consuming task and ensures the accuracy of the simulation.

Recently, Wu et al. [25] also proposed a two-dimensional DSMC method combining unstructured adaptive mesh by using *h-refinement* technique. Some adaptation parameters and criteria have been adopted to reduce the number of cells, while maintaining the accuracy of the solution. In their study, anisotropic cell refinement is used to remove hanging nodes caused by isotropic cell refinement. It is very efficient but the mesh, having high aspect ratio, often appears since no mesh quality control is implemented, which is not acceptable for an accurate DSMC method. In addition, Wu and Tseng [16] have developed an efficient parallel DSMC code using graph-partitioning technique to dynamically re-decompose the unstructured mesh.

1.4. Objectives of the paper

Based on previous reviews, the objectives of the current research are, firstly, to complete a parallel three-dimensional DSMC code, incorporating variable time-step method and unstructured adaptive mesh; secondly, to analyze the combination in detail using a supersonic flow past a sphere; thirdly, to apply to compute a near-continuum twin-jet interaction in a near-vacuum environment and the pumping performance of a spirally grooved drag pump. Simulated results are then compared with previous simulation and experimental data wherever available.

The paper begins with descriptions of the parallel implementation of the DSMC method, variable time-step method, unstructured adaptive mesh and finally the overall combination of the above features. Results are then considered treating test problems and conclusions follow in turn.

2. Numerical method

2.1. Parallel DSMC method

In the current study, we have extended previously implemented two-dimensional parallel DSMC code [14,16,17], which utilized multi-level graph-partitioning technique [16] to dynamically re-decompose the computational domain, into a three-dimensional parallel DSMC code [22]. One of the main reasons to adaptively re-decompose the computational domain is that load distribution is generally not known *a priori* for the DSMC simulation, especially during the transient period of simulation. The advantage by using graph-based partitioning technique is the "dimensionless" nature of the graph theory; thus, the extension from two- [16] to three-dimensional [22] code is conceptually straightforward. Details of the implementation can be found in Wu and Tseng [16, 22] and thus are only briefly described as follows.

The current DSMC method is implemented on an unstructured mesh using particle ray-tracing technique

by defining a cell neighbor-identifying array [12,14–17,22,25], which takes the advantage of the cell connectivity information, and could be potentially hybridized with the continuum method, such as the N–S solver, using unstructured mesh. Number of particles in each cell is taken as the vertex (cell center in the DSMC mesh) weight and unitary weight is used for edge cut under the framework of graph theory. Stop at Rise (SAR) [26] scheme is used to determine when to repartition the computational domain by defining a degradation function, which represents the average idle time per time-step for each processor including the cost of repartition. Communication of particle data between processors only occurs when particle hits the inter-processor boundary, while communication of cell data only occurs when repartitioning the domain takes effect. Data for communication is sent and received as a whole to reduce the communication time between processors. From previous studies [16,22], the current parallel DSMC method using dynamic domain decomposition generally runs 30–100% faster than that using static domain decomposition up to 64 processors on IBM-SP2 parallel machine, considering a high-speed lid-driven cavity flow as the test problem. For some complicated problems [16,22], the speedup can be even 200% faster since it is very hard to exactly partition the domain in a load-balanced manner before simulation if static domain decomposition is used.

The current parallel code, in SPMD (Single Program Multiple Data) paradigm, is implemented on the IBM-P690 parallel machines (distributed memory system) using message passing interface (MPI) to communicate information among processors. In addition, it is essentially no code modification required to adapt to other parallel memory-distributed machines (e.g., PC-Cluster system) employing the same MPI libraries for data communication.

2.2. Variable time-step method

In DSMC, particle distribution per cell has been shown to be linearly proportional to the inverse of number density for two-dimensional flow if the constant weight for each particle is used and cell size scales exactly with local mean free path; that is, the number of simulated particles is lower in higher-density regions, while low-density regions are over resolved [18]. More computational time is indeed wasted in calculating the low-density regions than is needed, while the samples in high-density regions are not enough. This situation is even more obvious in three-dimensional flow, which the number of simulated particles per cell is squarely proportional to the inverse of gas density [18]. To obtain a more uniform distribution of simulated particles per cell throughout the computational domain without detrimental effects caused by particle cloning, a variable time-step method for DSMC is then proposed on the unstructured mesh system as follows. Advantages of implementing the variable time-step scheme are to reduce both the simulated particle numbers and the number of iterations for transient period towards steady state, when the sampling normally begins in DSMC.

Fig. 1 shows the schematic diagram of the variable time-step concept for a simulated particle moves across the cell interface. Fluxes (mass, momentum and kinetic energy) conservation should be enforced exactly when a simulated particle crosses the cell interface. Thus,

$$\frac{W_1\Phi_1}{A\Delta t_1} = \frac{W_2 N_2 \Phi_2}{A\Delta t_2} \tag{1}$$

where Ws, Φs ($= m, mv, mv^2/2$ or other internal energy) and Δts are the particle weight, conserved flux quantity and time-step, respectively, and the numbers at subscript represents cell numbers. Note that A represents the area of cell interface between cell 1 and 2. N_2 is number of the simulated particle in cell 2, which originated from cell 1. There are several choices of the corresponding parameters to satisfy Eq. (1), with which we can play. The best choice is to set $N_2 = 1$ (without particle cloning) and $\Phi_1 = \Phi_2$ (without changing the velocity) across the cell interface, such that $W_1/\Delta t_1 = W_2/\Delta t_2$ always holds. In

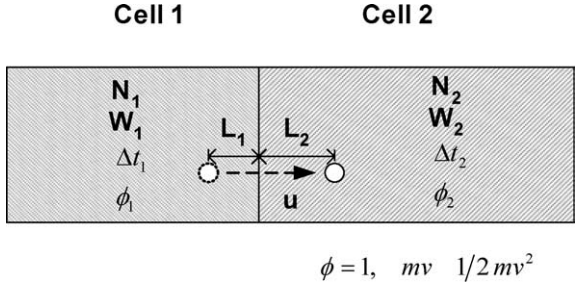

Fig. 1. Schematic diagram of variable time-step concept for a simulated particle crossing a cell interface.

other words, $W_i/\Delta t_i$ will be the same for all the cells throughout the computational domain. Using this approach, resulting number of simulated particles per cell for the three-dimensional flow scales with Δx ($\sim\sqrt[3]{V_c}$, V_c is the cell volume) [18] if cell size is proportional to the local mean free path, which otherwise scales with $(\Delta x)^2$. In doing so, the simulated particle will only have to adapt its weight that is proportional to the size of time-step, which is approximately commensurable to the local mean free path if solution-based adaptive mesh is used. Of course, the remaining time for a simulated particle, when crossing cell interface, should be rescaled according to the ratio of time-steps in original and destination cells. In the DSMC code, a reference time-step is first computed by selecting a reference cell volume (often the minimum cell volume). Then, time-steps in other cells are scaled using the fact $W_i/\Delta t_i =$ constant. The current implementation is comparatively flexible as compared with that of Markelov and Ivanov [19], where the computational domain was divided into few numbers of zones, within which time-step is constant in each zone.

2.3. Three-dimensional unstructured adaptive mesh

Similar to the *h-refinement* scheme for the two-dimensional unstructured triangular mesh presented previously [25], it has been extended to treat three-dimensional unstructured tetrahedral mesh in the current study. It is not as simple as it originally looks as compared with the two-dimensional case. In addition, a simple but effective mesh-quality control mechanism is added to improve the mesh quality for the three-dimensional case. Details about the general features, adaptation parameters and criteria and adaptation procedures are then described next in turn.

2.3.1. General features

General features of the current three-dimensional mesh adaptation are proposed as follows:

(1) unstructured tetrahedral mesh;
(2) *h-refinement* with mesh embedding;
(3) local cell Knudsen number (inversely proportional to density) and free-stream parameter (ratio of local density to free-stream value) as the mesh adaptation parameters for external flow.

2.3.2. Adaptation parameter and criteria

All mesh adaptation methods need some means to detect the requirement of local mesh refinement to better resolve the flow features in the flow fields and hence to achieve more accurate numerical solutions. This also applies to the DSMC method. It is important for the adaptation parameter to detect a variety of flow features, but does not cost too much computationally. Often gradient of properties such as temperature, density or velocity is used as the adaptation parameter to detect rapid changes of the flow-field solution in CFD. However, by considering the statistical nature of the DSMC method, density may be adopted instead as the adaptation parameter [25]. Using density as the adaptation parameter in DSMC is natural since it is generally required that the cell size be much smaller than the local mean free path to satisfy the underlying assumption of DSMC, in which particle movement and collision can be uncoupled.

There are two adaptation criteria to determine if the cell should be refined or not. One is the density local Knudsen number, Kn_c, and the other is free-stream parameter ϕ_i. They are described in detail as follows.

Firstly, to use the density as an adaptation parameter, a local cell Knudsen number is defined as

$$Kn_c = \frac{\lambda_c}{\sqrt[3]{V_c}} \tag{2}$$

where λ_c is the local cell mean free path based on VHS model [18] and V_c is the volume of the tetrahedral cell. When the mesh adaptation module is initiated, local Knudsen number at each cell is computed and compared with a preset value, Kn_{cc}. If $Kn_c < Kn_{cc}$ in some cells, then mesh refinement is required for those cells. This adaptation parameter is expected to be most stringent on mesh refinement (more cells are added); hence, the impact to DSMC computational cost might be high, but is required to obtain an accurate solution.

Secondly, considering the practical applications of mesh adaptation in external flows, we have added another constraint, $\phi_i \geqslant \phi_0$, where ϕ_i, free-stream parameter of each cell, is defined as

$$\phi_i = \frac{\rho_i}{\rho_\infty} \tag{3}$$

and ϕ_0 is a preset value, which is normally taken as 1.05. Not only does the above constraint help to reduce the total refined number of cells to an acceptable

level by reducing greatly the number of cells in the free-stream region, but also it reduces the total computational time tremendously by using very large time-step (hence, very large particle weight) in this region.

2.3.3. Adaptation procedures

As mentioned earlier, the basic idea of refining the three-dimensional unstructured mesh by *h-refinement* is similar to that for two-dimensional unstructured mesh [25], but the corresponding *h-refinement* process is relatively complicated for three-dimensional unstructured tetrahedral mesh, as shown in Fig. 2 for removing hanging nodes. General procedures for the current three-dimensional *h-refinement* scheme include:

(1) *Isotropic mesh refinement*: Isotropic mesh refinement is employed for the cells, which flag for mesh refinement based on the adaptation criteria;
(2) *Removing hanging node*(*s*): An-isotropic or isotropic mesh refinement is utilized in the (interfacial) cells next to those cells that have been isotropically refined, as shown in Fig. 2, which will be detailed shortly;
(3) *Mesh quality control*: Isotropic mesh refinement is again used to remove the cells having high as-

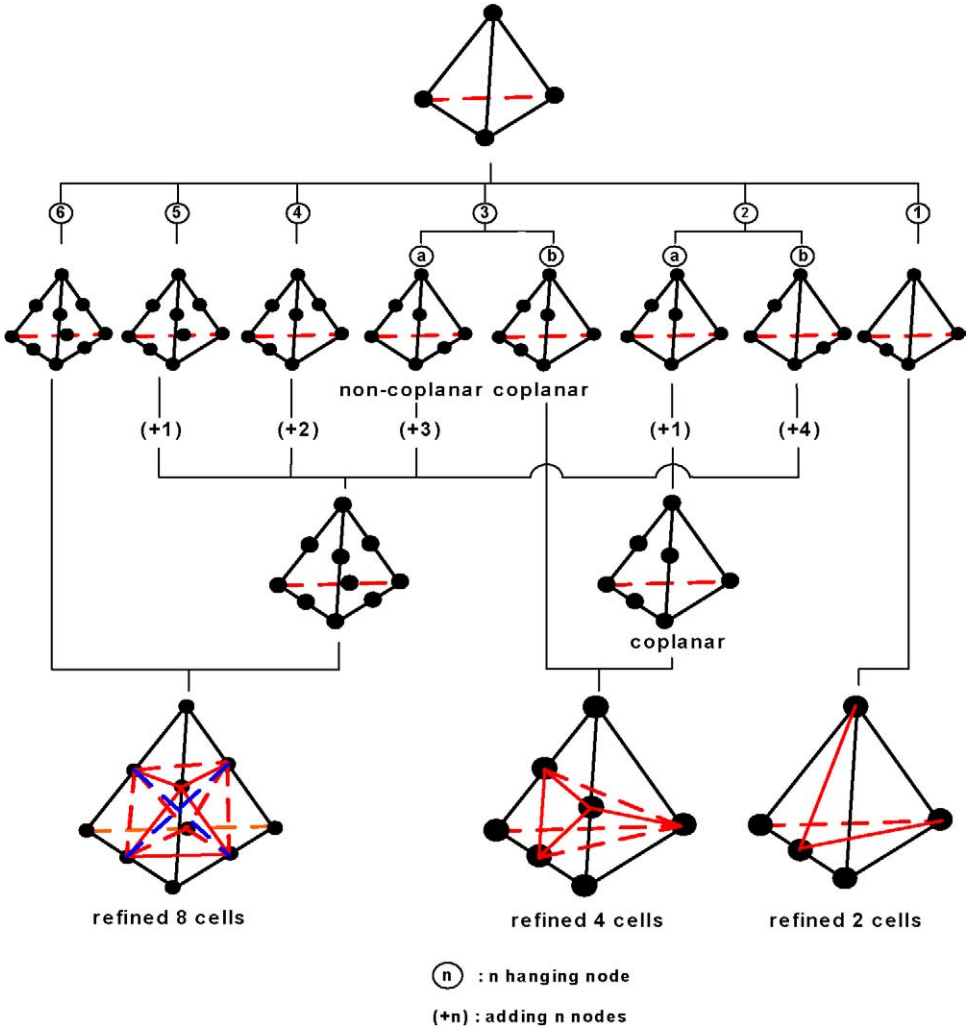

Fig. 2. Removal of hanging nodes in mesh adaptation procedures for a three-dimensional tetrahedral mesh.

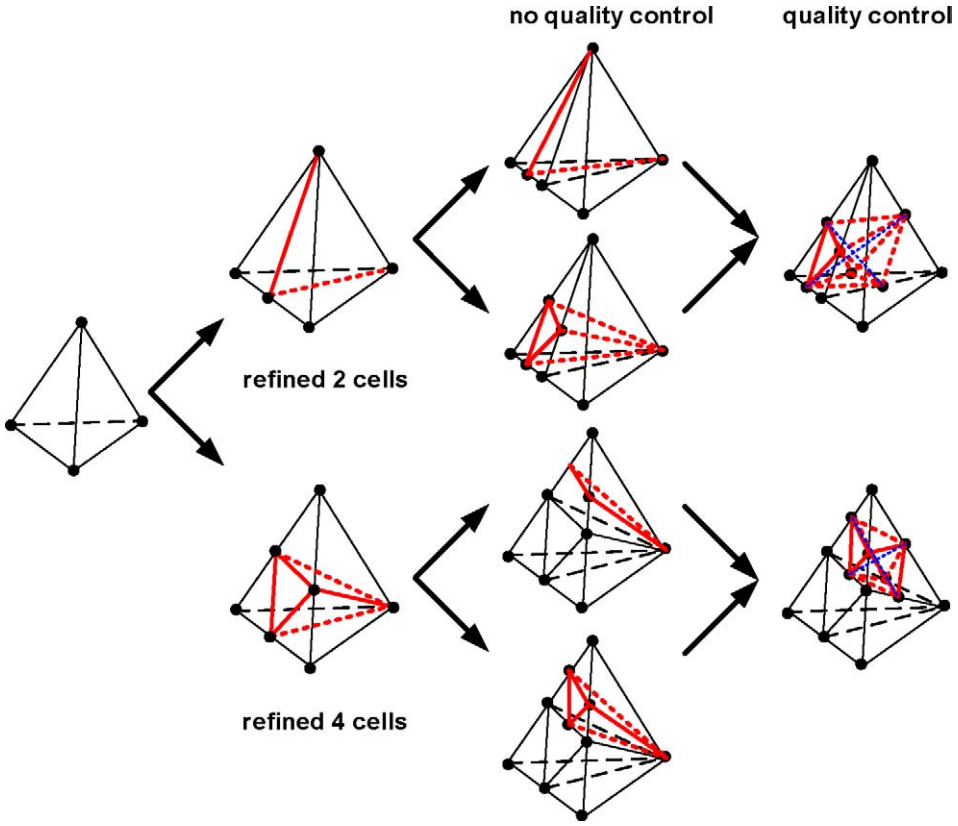

Fig. 3. Schematic diagram of mesh quality control for removing cells having high aspect ratio.

pect ratio, as shown in Fig. 3, which is simple but effective in improving the mesh quality.

Besides, one other important step during the mesh adaptation procedures is to modify the old cell neighbor-identifying arrays and add the new cell neighbor-identifying arrays accordingly.

Next, we will explain how to remove hanging nodes (step 2 in the above) generated by isotropic mesh refinement (step 1 in the above) as sketched in Fig. 2. In brief summary, there are at most six types of hanging-node numbers for the tetrahedral mesh, in which there may exist different arrangements of the hanging-node location for each type, e.g., types 2a and 2b for two hanging nodes and types 3a and 3b for three hanging nodes, as shown in Fig. 2. General rule for removing 2–6 non-coplanar hanging nodes (types 2b, 3a, 4, 5 and 6) is to add enough node(s) on the edge(s) such that eight refined tetrahedral cells are formed by isotropic mesh refinement. For two or three coplanar hanging nodes (types 2a and 3b), node is either added on the edge of the same co-plane (type 2a) into type 3b such that four refined tetrahedral cells can be formed by an-isotropic mesh refinement. If only one hanging node is found, then two refined tetrahedral cells are formed by an-isotropic mesh refinement directly.

2.4. Overall DSMC computational procedure

In the current study, the overall procedure of the parallel DSMC method combining the variable time-step method and unstructured adaptive mesh is illustrated in Fig. 4. Preliminary parallel DSMC simulation first carries on using the initial partitioned coarse mesh to gather enough samples for later mesh adaptation. Approximately 50,000 sampling particles per cell are considered acceptable for statistical uncertainties since the density is a sampling quantity having zeroth-order velocity moment. Then, mesh refinement

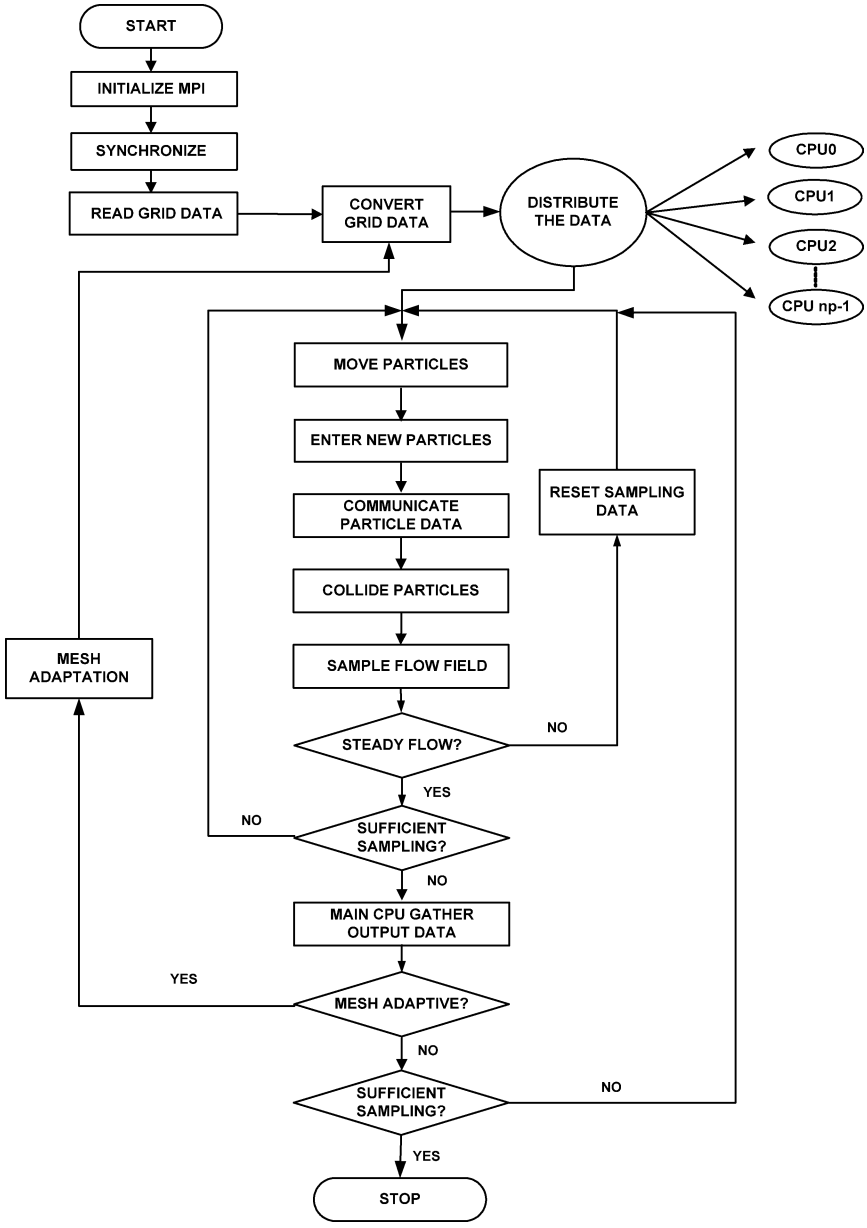

Fig. 4. Overall parallel DSMC computational procedure using variable time-step method and unstructured adaptive mesh.

is conducted for those cells flagging for refinement and the sampled data on the initial (coarse) *parental cells* are proportionally redistributed (based on the magnitude of cell area) to the fine *child cells* accordingly. Note that mesh refinement is conducted in serial mode (on the server itself) rather than in parallel mode for the time being. This process repeats until the adaptation criteria are all satisfied in each cell or maximum number of adaptation levels is reached. In the process, time-step in each cell is adjusted according to the concept of variable time-step discussed earlier. In practice, the mesh adaptation runs very fast since only integer operations are involved except adding positions for new nodes. Most of the time is spent on

updating/modifying/adding cell neighbor-identifying arrays. We have used the concept of loops over nodes, instead of cells, by searching through the cells sharing the same node, which turns out to be very efficient. For example, for processing three million 3-D unstructured tetrahedral cells, it takes less than 20 minutes on a 1.6-GHz (Intel) personal computer. Preliminary results show that the preprocessing time increases approximately linearly with the number of cells. In spite of this fact, parallel implementation of the mesh adaptation is currently in progress to make the most of the parallel computing resources.

The final-level unstructured adaptive mesh, with updated cell neighbor-identifying arrays and node coordinates, is then repartitioned using the final density distribution in each cell as the vertex weight. Finally, the parallel DSMC simulation carries on by increasing proportionally the number of simulated particles to the expected amount in the computational domain.

3. Results and discussions

The current implementation is verified and discussed in detail by computing a three-dimensional supersonic flow past a sphere. Then, it is applied to simulate a three-dimensional near-continuum twin-jet interaction in a near-vacuum environment and the pumping performance of a spirally grooved drag pump. Related issues about the advantages of using variable time-step method and unstructured adaptive mesh will be discussed along with the presentation of simulation results, especially the case of supersonic flow past a sphere. In addition, physics of the flow field related to these problems will be described as brief as possible, since we are interested in demonstrate its capability in the current study.

All the DSMC simulations presented hereafter are carried on using 32 processors of IBM-P690 parallel machine using dynamic domain decomposition, unless otherwise specified. Only evolution of domain decomposition will be shown later to demonstrate the effectiveness of dynamic domain decomposition. Details of the parallel performance are obviously out of the scope of the current paper and will be skipped in the discussion.

3.1. Supersonic flow past a sphere

3.1.1. Flow and simulated conditions

Considering a supersonic flow past a sphere, related flow conditions are listed as follows: VHS nitrogen gas, free-stream Mach number $Ma_\infty = 4.2$, free-stream number density $n_\infty = 9.77\text{E}20$ particles/m^3, free-stream temperature $T_\infty = 66.25$ K, stagnation temperature $T_0 = 300$ K, fully thermal accommodated and diffusive sphere wall with the temperature T_w (equal to T_0). The corresponding free-stream Knudsen number Kn_∞ is 0.1035, based on the free-stream mean free path and diameter of the sphere. Details of the flow and simulated conditions can be found in Table 1. These represent the flow conditions of the experiments by Russell [27]. The flow structure of this flow is simulated as a three-dimensional case by considering only 1/16 of the full domain, which takes advantage of the flow-field symmetry. In addition, 20,000 time-steps are used to sample the flow properties.

3.1.2. Evolution of adaptive mesh and effects of variable time-step

To satisfy the constraint of the cell size and the save the number of cells in the free-stream region of an external flow, the local Knudsen number criterion (Kn_{cc}) and free-stream parameter (ϕ_0) for mesh adaptation are chosen as 2 and 1.05, respectively. Evolution of adaptive surface mesh at each level (level-0: 5353 cells; level-1: 22,510 cells; level-2: 164,276 cells) with mesh quality control is presented in Fig. 5. In Fig. 5(c) (level-2 adaptive surface mesh), it illustrates that the high-density region due to the bow

Table 1
Complete listing of physical and VHS parameters of a supersonic flow past a sphere

N_2 gas	$Kn = \lambda_\infty / D = 1.035\text{E}{-1}$	$D = 1.28\text{E}{-2}$ (m)	$Re_L = 30$
$p_\infty = 0.893$ Pa	$n_\infty = 9.77\text{E}20$ (#/m^3)	$Ma_\infty = 4.2$	$U_\infty = 697.022$ (m/s)
$m_{\text{ref}} = 4.65\text{E}{-26}$ (kg)	$d_{\text{ref}} = 4.17\text{E}{-10}$ (m)	$T_{\text{ref}} = 273$ (K)	$T_w = 300$ (K)
$T_0 = 300$ (K)	$T_\infty = 66.25$ (K)	Zr[a] $= 5$	$\omega = 0.74$

[a] Rotational relaxation number.

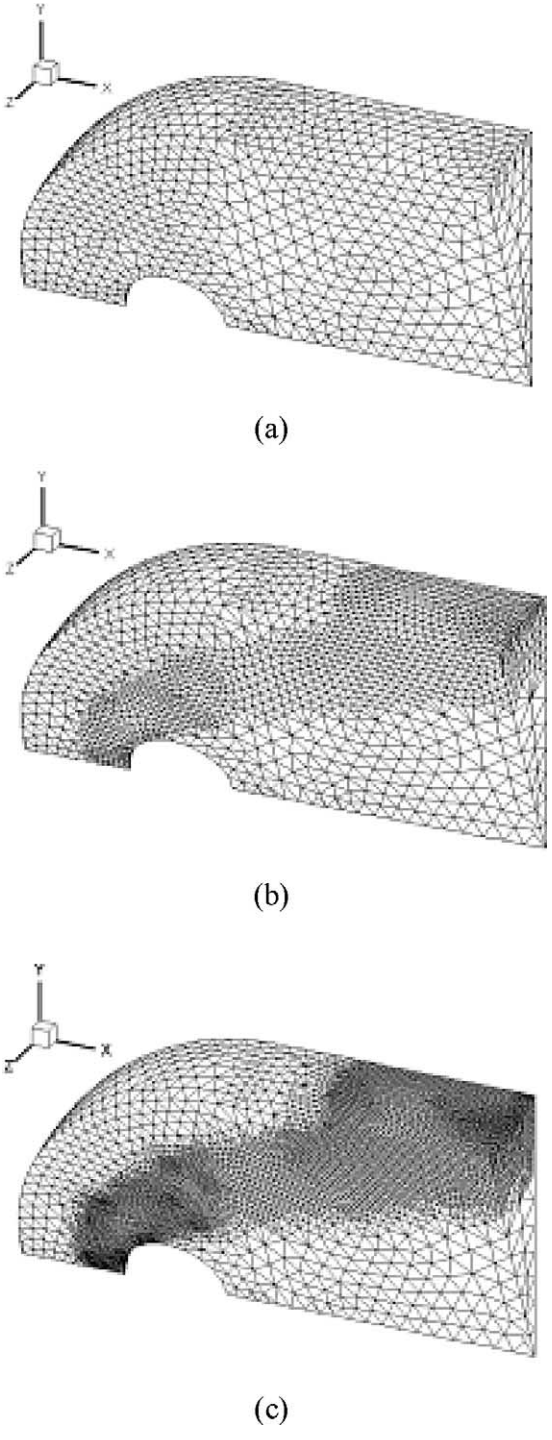

Fig. 5. Evolution of unstructured adaptive surface mesh at different levels for a supersonic flow past a sphere. (a) Level-0 (5353 cells); (b) level-1 (22,510 cells); (c) level-2 (164,276 cells).

shock around the sphere is clearly captured with much finer mesh distribution. In addition, the mesh distribution in the free-stream region is comparably coarse as compared with the local mean free path due to application of the free-stream parameter mentioned earlier. Application of the free-stream parameter effectively reduces the number of cells in the free-stream region, where the cell-size requirement ($\Delta x < \lambda$) could be relaxed due to nearly uniform density distribution (vanishing density gradient) in this region.

Variation of particle distribution is reduced greatly as illustrated in Fig. 6, which shows the comparison of distribution of averaged particles per cell on level-2 adaptive mesh using constant time-step (CTS) and variable time-step (VTS) methods, with the same reference (minimum) time-step (1.E−08). In this figure, using VTS method, the reduction of averaged number of particles per cell can be as large as 10-fold and 30-fold in the regions of oblique bow shock and wake region behind the sphere, respectively. This is achieved by keeping both the particle weight and averaged particle per cell the same in the reference cell (minimum cell, near the stagnation point in this case) for both VTS and CTS methods. Resulting simulated particles at steady state are reduced from 1.7 millions, using constant time-step (CTS), to 0.34 millions, using variable time-step (VTS) in this case (Fig. 7). In the other words, the statistical uncertainty in the minimum cell is kept approximately the same when comparing both methods. In addition, another benefit of applying VTS method to the current unstructured adaptive mesh is that it can reduce dramatically the number of iterations of transient period towards steady state, as illustrated in Fig. 7. In Fig. 7, the required number of iterations for transient period if VTS is applied is about only 25% of that if CTS is applied. This is because appropriate time-step in each cell is used to approach steady state if VTS is used, rather than very small time-step is used in large cell if CTS is used. In the current case, if we expect roughly the same statistical uncertainty in the minimum cell to obtain macroscopic properties for both CTS and VTS methods, the combination of VTS and unstructured adaptive mesh could save the computational time up to one order of magnitude. Not only does the above-mentioned combination decrease the computational cost greatly, but also it can adaptively satisfy the cell-size requirement ($Kn_c > 1$) in the flow field, as shown in Fig. 8, where distribution of local

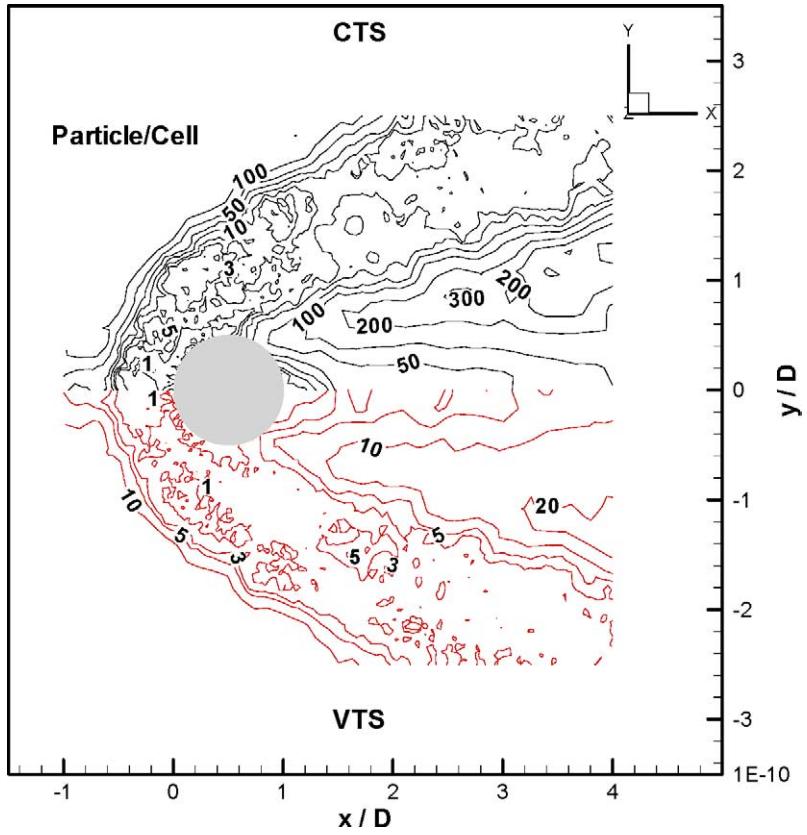

Fig. 6. Comparison of distribution of averaged particles per cell on level-2 adaptive mesh using constant time-step (CTS) and variable time-step (VTS) for a supersonic flow past a sphere.

cell Knudsen number is illustrated on both initial mesh and level-2 adaptive mesh. For the distribution of local cell Knudsen numbers, it shows all values are greater than unity ($Kn_c > 1$) except in the free-stream region, where a free-stream adaptation criterion has been applied to reduce the number of cells.

Corresponding number of cells at each level of mesh adaptation (Fig. 5) is summarized in Table 2 along with those without mesh quality control. The number of cells increases from 5353 (initial) to 164,276 (level-2 with mesh quality control) and 151,732 (level-2 without mesh quality control). Increase of number of cells due to the simple mesh quality control is relatively limited (∼ 8%), but it greatly improves the mesh quality, as illustrated in Fig. 9 (surface mesh), where the interfacial cells with larger aspect ratio (in the regions before and after the bow shock) without mesh quality control (upper part of Fig. 9) have been effectively removed as shown at the bottom of the same figure. Also the increase of computational time due to mesh quality control is computationally negligible since most operations of mesh quality control are only integer related as mentioned earlier.

Table 2
The number of cells at each level of mesh adaptation with or without mesh quality control

Level	0	1		2	
mesh quality control	–	No	Yes	No	Yes
cell no.	5353	22,510	22,510	151,732	164,276

3.1.3. Evolution of domain decomposition

Fig. 10 illustrates the initial and final domain decomposition for a supersonic flow past a sphere using

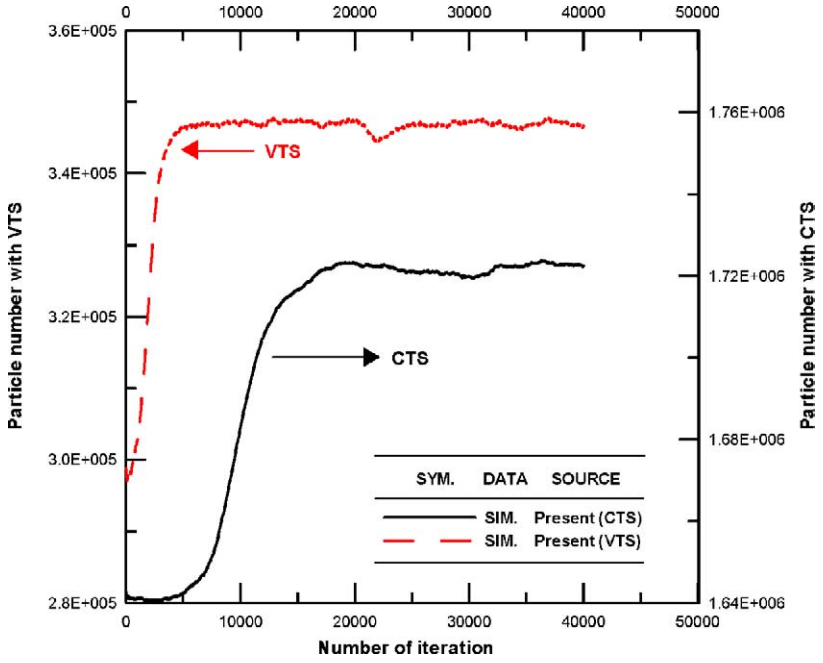

Fig. 7. Total number of simulated particles as a function of number of iteration using constant time-step (CTS) and variable time-step (VTS) for a supersonic flow past a sphere.

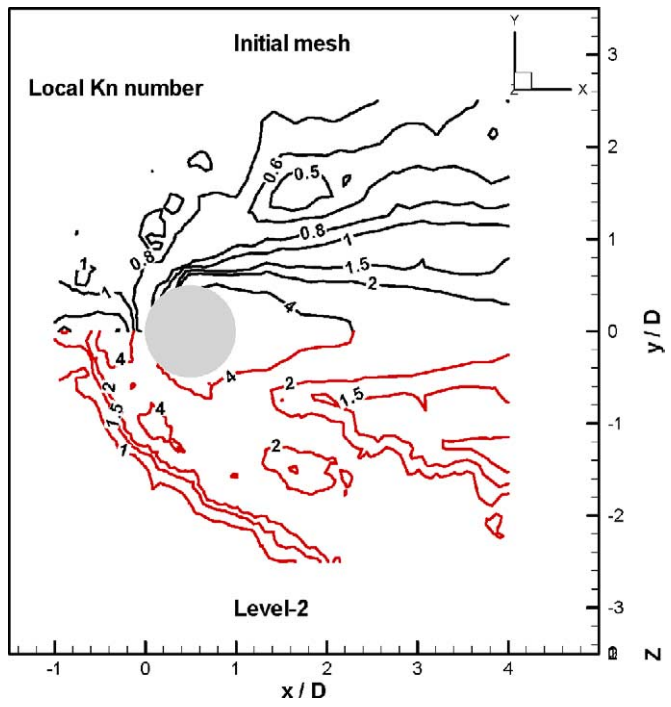

Fig. 8. Surface local cell Knudsen number distribution on initial and level-2 adaptive mesh for a supersonic flow past a sphere.

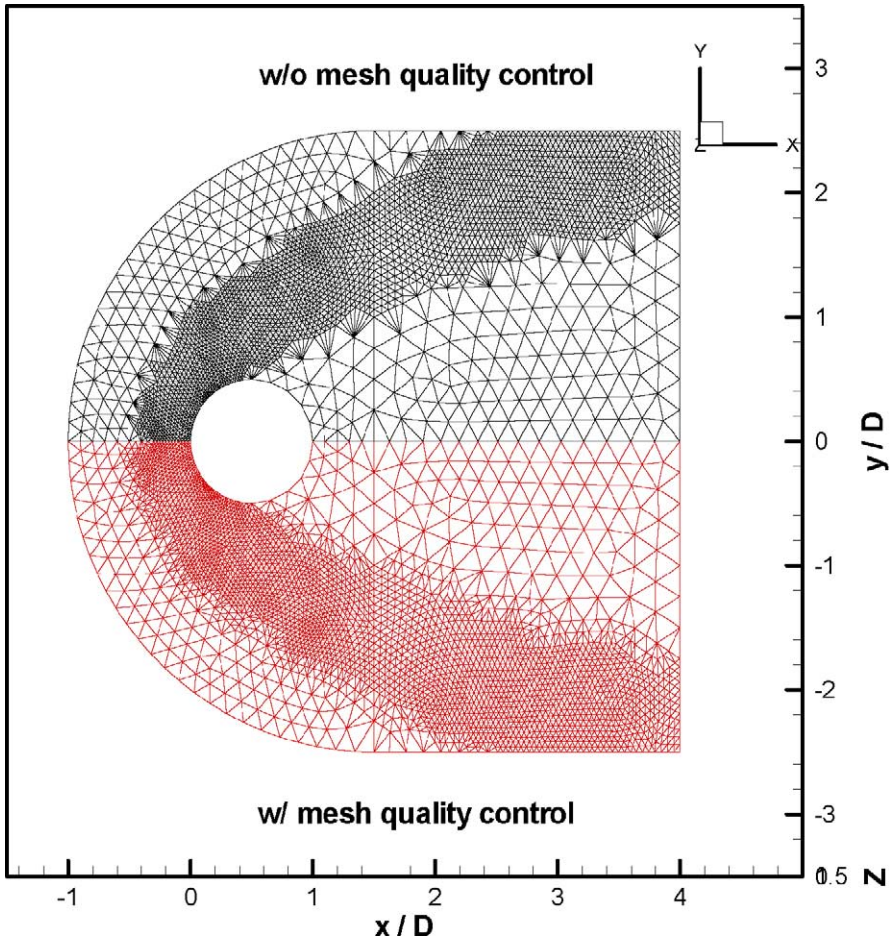

Fig. 9. Level-2 adaptive surface mesh distribution with (lower) and without (upper) mesh quality control for a supersonic flow past a sphere.

32 processors. Note that initial domain decomposition for initial un-adaptive mesh is obtained using the number of particles as the weight for each cell assuming uniform density distribution, while the final domain decomposition is obtained via repeated graph repartitioning, which reflects the steady-state density distribution. It is obvious that the initial domain decomposition has adapted to a totally different domain decomposition following the flow dynamics of the problem. This clearly shows the effectiveness of the dynamic domain decomposition, which has been discussed in detail in Wu and Tseng [16,22].

3.1.4. Density distributions

Fig. 11 presents the normalized density contour using both initial and level-2 adaptive mesh for a supersonic flow past a sphere on axisymmetric plane. Results show that maximum density occurs near the stagnation point in front of the sphere and minimum density occurs in the wake region as expected. The difference between the computational results using initial and level-2 adaptive mesh is obvious, which shows the use of adaptive mesh in the current study may not be trivial. Similar situation can also be found for other properties, such as translational/rotational temperatures and Mach number. In addition, Fig. 12 shows the computational normalized density distribution along the centerline with experimental data by Russell [27]. Results show that the better agreement with experimental data is found using level-2 adaptive mesh near the stagnation region in front of the sphere due to better resolution of flow properties in the region.

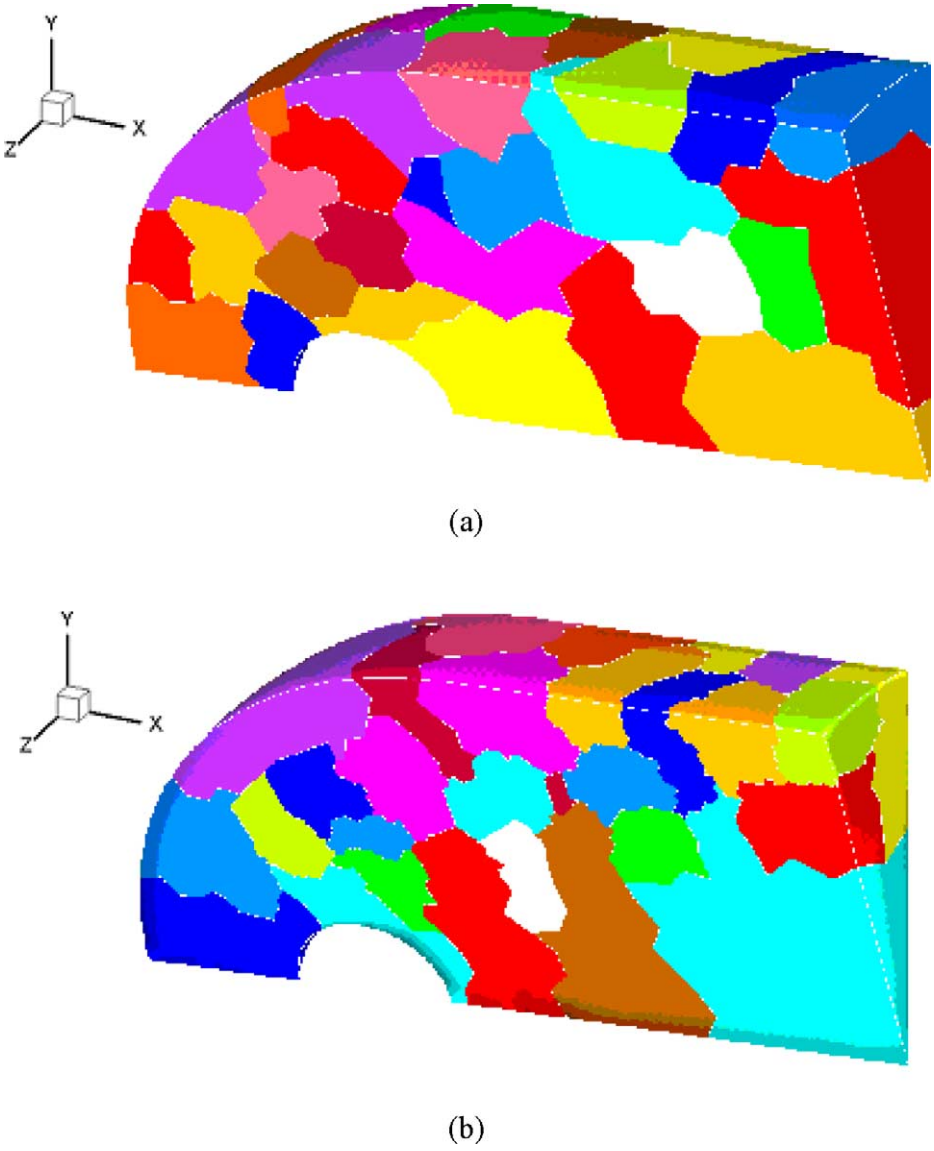

Fig. 10. Evolution of domain decomposition for a supersonic flow past a sphere. (a) Initial; (b) final.

3.2. Twin-jet interaction in a near-vacuum environment

3.2.1. Flow and simulation conditions

Considering two parallel, near-continuum, under-expanded round jets issuing from sonic orifices into a near-vacuum environment, the flow field is simulated and compared with experimental data. Related flow conditions are listed as follows: the test gas is nitrogen gas; stagnation pressure $P_0 = 870$ Pa; stagnation temperature $T_0 = 285$ K; background pressure $P_b = 3.7$ Pa; resulting pressure ratio $P_0/P_b = 235$. Background pressure effect is neglected in the simulation, similar to previous simulation by Dagum and Zhu [28]. Thus, vacuum boundary conditions at the outer boundaries are employed for simplicity. The corresponding Knudsen number Kn_{th} $(= \lambda_{\text{throat}}/D)$ is 0.00385. Corresponding flow conditions represent a

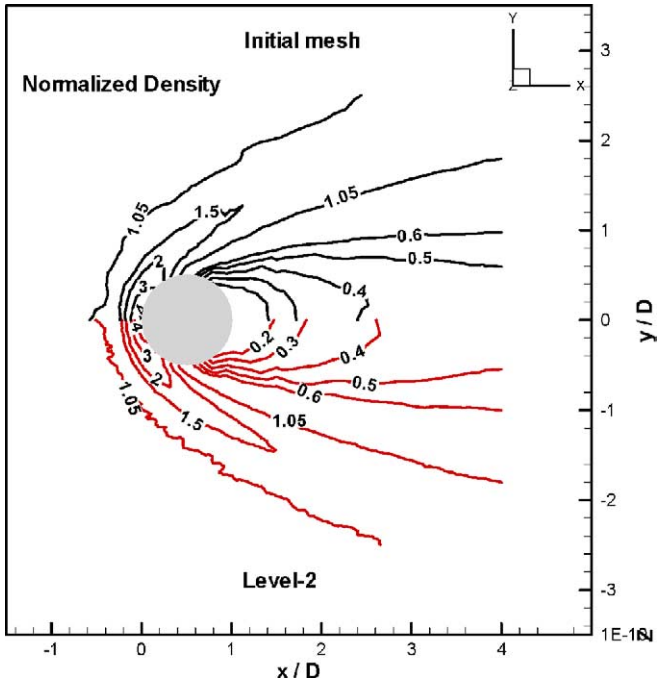

Fig. 11. Normalized density contour using initial and level-2 adaptive mesh for a supersonic flow past a sphere.

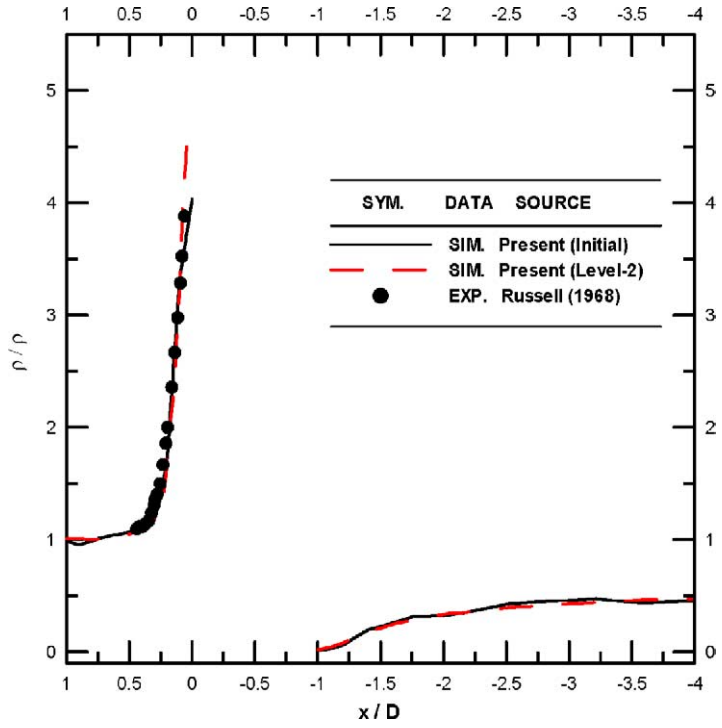

Fig. 12. Computed normalized centerline density along with experimental data for a supersonic flow past a sphere.

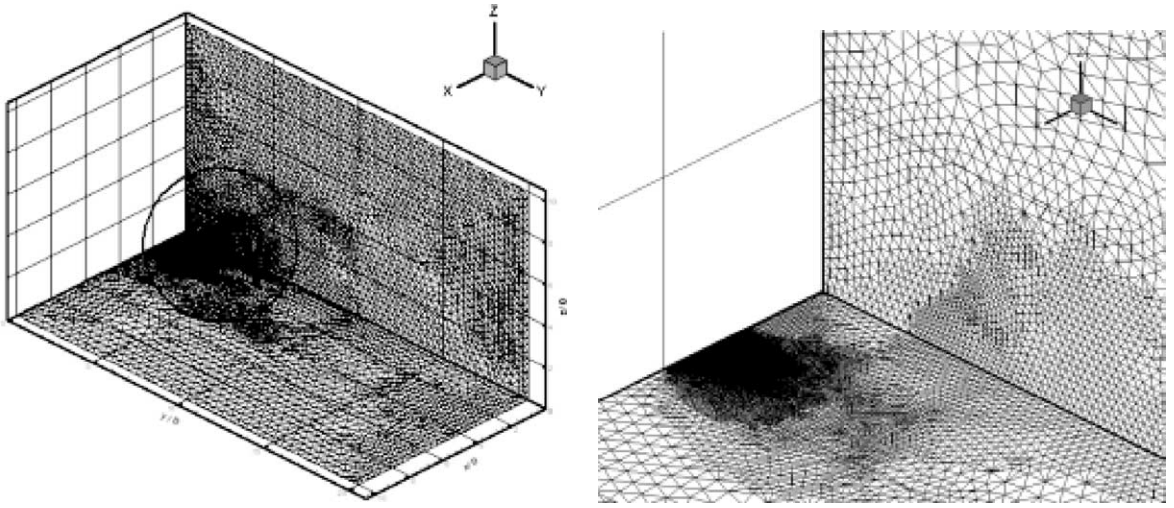

Fig. 13. Surface mesh distribution on $x-y$ and $y-z$ planes for the twin-jet interaction in a near-vacuum environment.

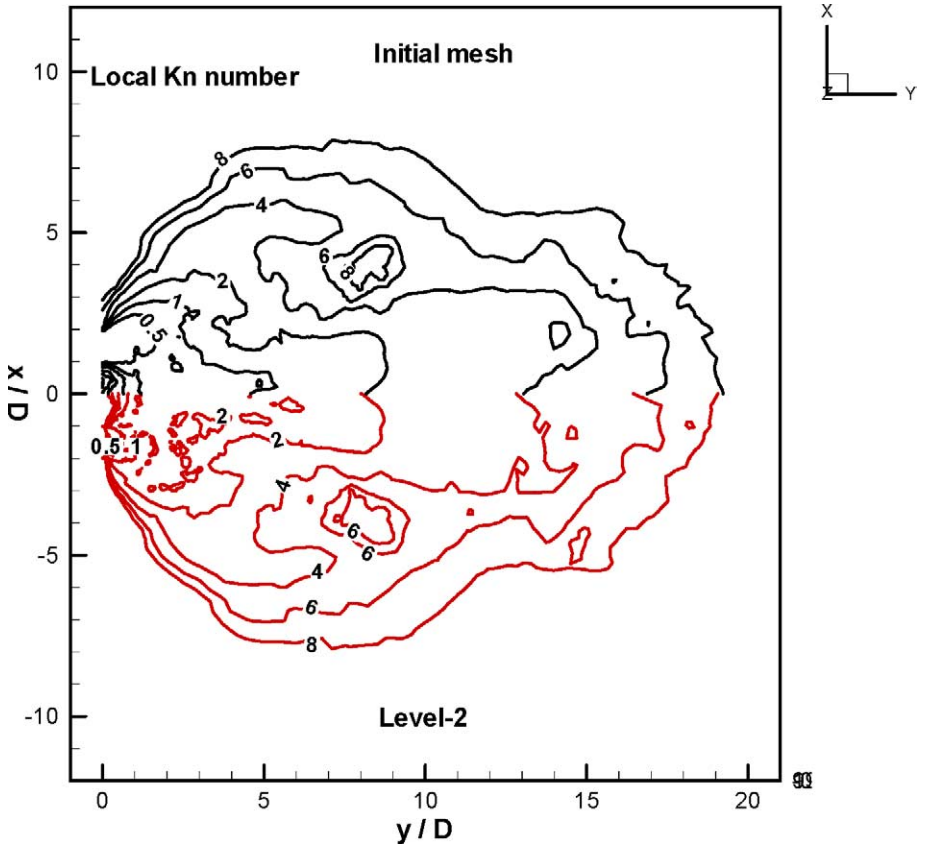

Fig. 14. Surface local cell Knudsen number distribution on initial and level-2 adaptive mesh for the twin-jet interaction in a near-vacuum environment.

challenging computational problem since it involves flow regimes from a near-continuum flow at the inlet to a near free-molecular flow at the outer boundaries, where the DSMC method may be the only available tool for analyzing this problem correctly.

Computational domain is reduced to 1/4 of the original physical domain by taking advantage of the flow symmetry. Height (H), width (W) and length (L) of the simulation domain are taken long enough as $10D$, $10D$ and $20D$, respectively, which D is the diameter of the orifice. Note that the distance between the two primary jet centerline is three times of the orifice diameter to match the experimental conditions. In order to compare to the previous simulation data, internal energy is relaxed through the Borgnakke–Larsen model [29], using a fixed rotational collision number of 5. The resulting simulation particles are about 8 millions at steady state and the number of cells is approximately 0.66 millions after 2 levels of mesh refinement using $Kn_{cc} = 2$ (Fig. 13). Fig. 13(a) shows the surface mesh along x–y and y–z planes, while Fig. 13(b) illustrates the exploded view of the surface mesh in the same planes near the sonic orifice, where the mesh is refined to meet the adaptation criteria. Corresponding local cell Knudsen number distribution is shown in Fig. 14, where most of the flow field satisfies the general cell-size requirement except the locations very near the sonic orifice ($0.5 < Kn_{cc} < 1$). Besides, 20,000 time-steps are used to sample the particles for obtaining macroscopic properties. The reference (minimum) time-step is 7.61E−09.

3.2.2. Density contour

Fig. 15 presents the normalized density contours (with respect to the density at the sonic orifice) at plane $z = 0$, where the value of the contour is plotted as the height. Density along the centerline of primary jets decreases dramatically due to expansion, while the density between the two jets increases rapidly initially and then decreases later in the main flow direction due to

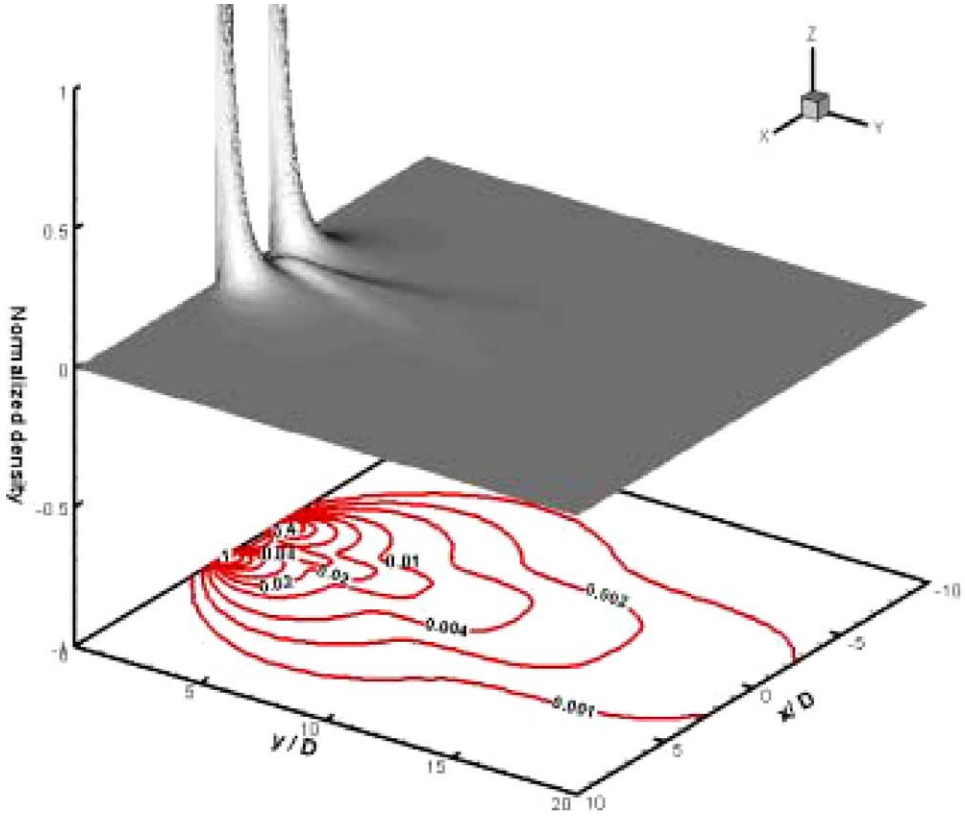

Fig. 15. Normalized density distribution on x–y plane ($z = 0$) for the twin-jet interaction in a near-vacuum environment.

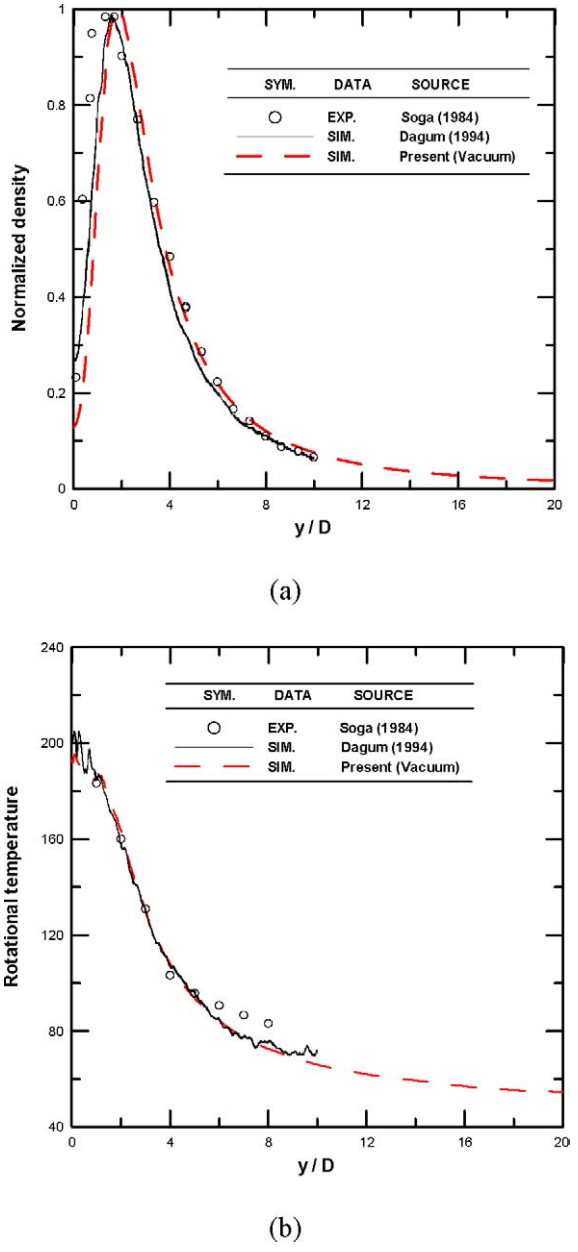

Fig. 16. Property distribution along y-axis (symmetric line). (a) Normalized density; (b) rotational temperature.

the compression of particles expanding from the two jets. Thus, a secondary jet is clearly formed along y-axis between the two primary jets. Similar evidence for the formation of a secondary jet between the two

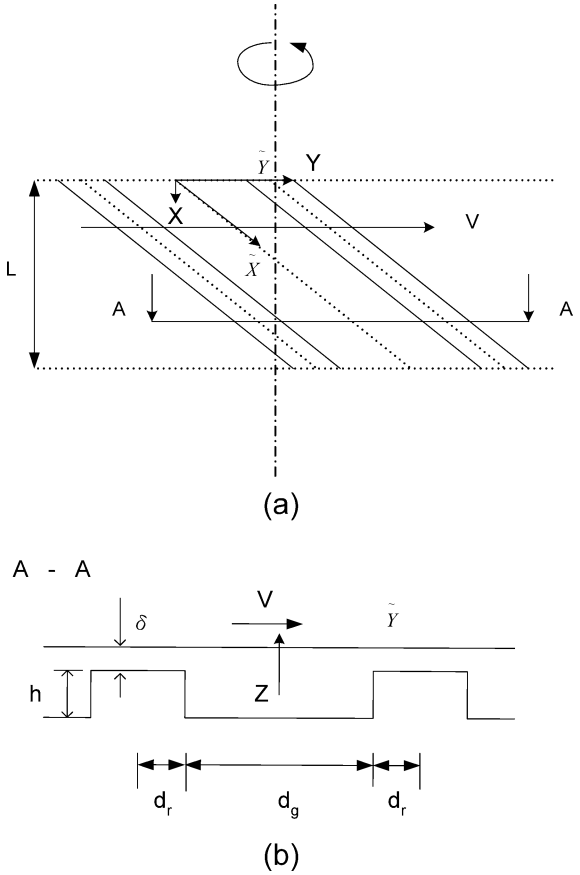

Fig. 17. Sketch of the spirally grooved drag pump.

jets exists also for temperature (translational and rotational) contours, which is not shown for brevity.

3.2.3. Centerline properties distribution

Figs. 16(a) and 16(b) show the density and rotational temperature distribution along the symmetric centerline (e.g., y-axis in Fig. 13, in the direction of major flow direction), respectively. Experimental data measured using laser induced fluorescence by Soga et al. [30] and DSMC simulation data by Dagum and Zhu [28] are also included for comparison. It is clear that current simulation data agree reasonably well with both experimental data [29] and previous DSMC simulation data [27], where the computational domain is smaller as compared with the current simulation. Similar trend is also found for rotational temperature distribution as shown in Fig. 16(b). In addition, good agreement between the current simulation and experiments

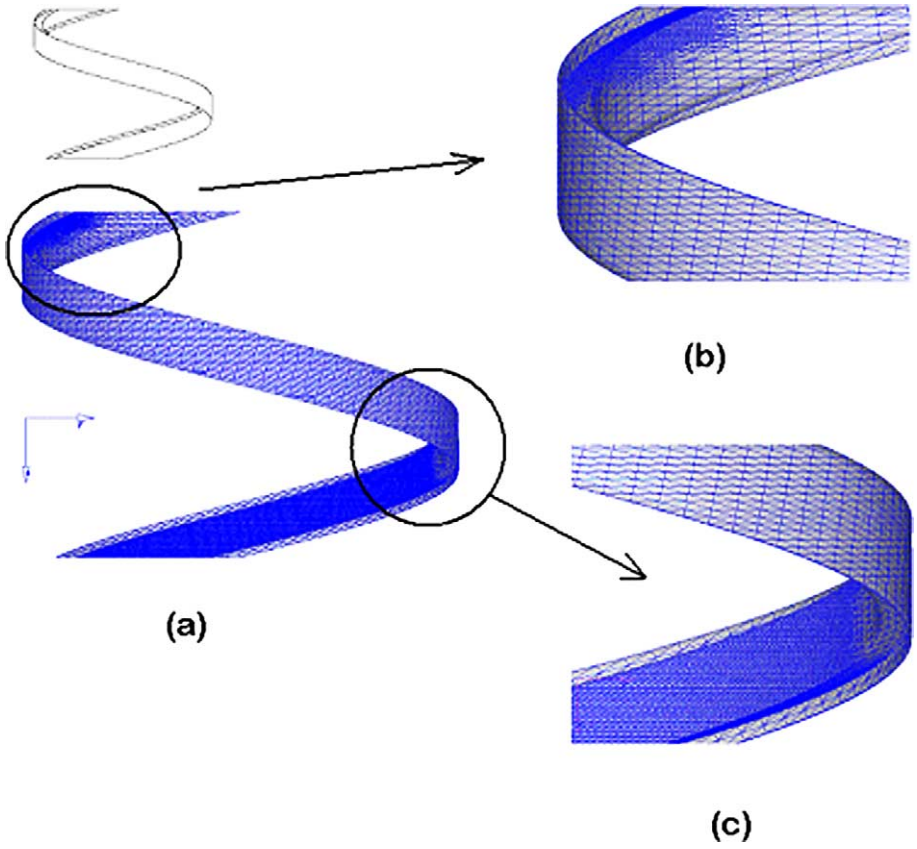

Fig. 18. Adaptive surface mesh distribution and exploded views of the refined mesh for the spirally grooved drag pump ($P_e/P_i = 100$).

can be also found for density profiles at different y/D positions, which is not shown for brevity.

3.3. Flow field of a spirally grooved drag pump

3.3.1. Flow and simulated conditions

The use of turbomolecular pumps makes it possible to realize a clean (oil vapor free) vacuum, which is often required by the semiconductor materials processing. Hence, the understanding of the related flow field in these molecular drag pump is very important to improve its the pumping performance. Nanbu et al. [31] also computed and experimented pumping performance of the spiral drag pump with the different output pressures and geometrical variations of gap. Thus, as a final test case, we are interested in demonstrating the use of the current DSMC implementation to analyze the pumping performance of a spirally grooved drag pump.

Considering a spirally grooved drag pump (Fig. 17), the rotor rotates counterclockwise, viewing from the top, flow field is simulated using the current DSMC implementation. Related flow conditions are listed as follows: the test gas is nitrogen gas; rotational speed of rotor $\omega = 36{,}000$ rpm; inlet pressure $P_i = 1$ Pa; outlet pressure $P_e = 100$ Pa; gas temperature $T_0 = 296$ K; axial length of rotor $L = 115$ mm, screw angle $\alpha = 15°$; width of ridge $d_r = 2.705$ mm; width of groove $d_g = 13.06$ mm; depth of groove $h = 4$ mm; gap between ridge and casing $\delta = 0.48$ mm. Resulting compression ratio P_e/P_i is 100. Knudsen number at the inlet $Kn_i = 1.3249$, while the Knudsen number at the exit is $Kn_e = 0.013249$, both based on the width of groove d_g. Coriolis force due to rotation of the rotor is considered in this case, but it is found that the effect is relatively small, less than 0.5% in this case. Treatment of pressure boundary conditions developed previously [32] is used to handle the inlet and outlet

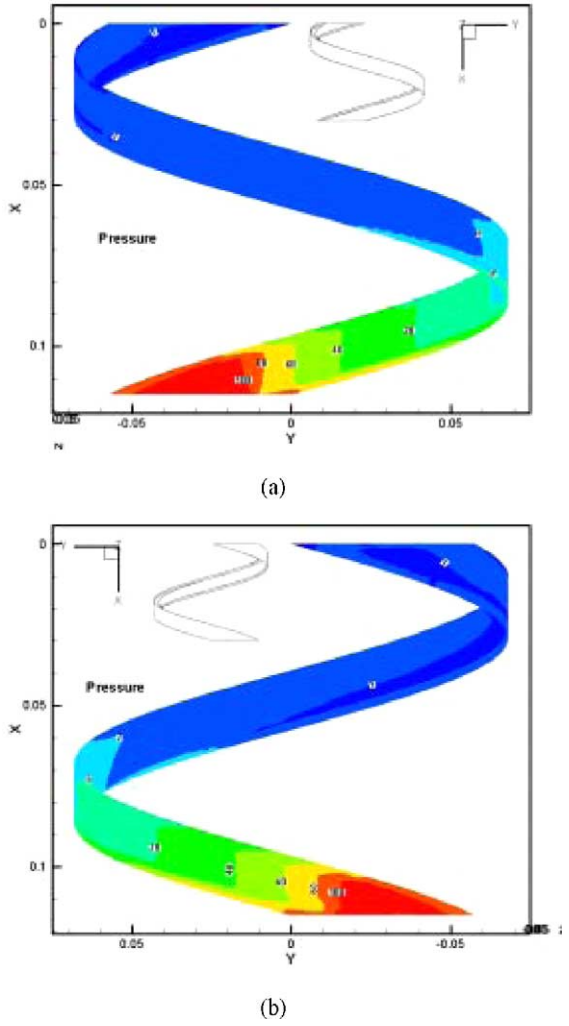

Fig. 19. Pressure contour on the surface of the spiral groove ($P_e/P_i = 100$), viewing at two angles by rotating 180° along x-axis.

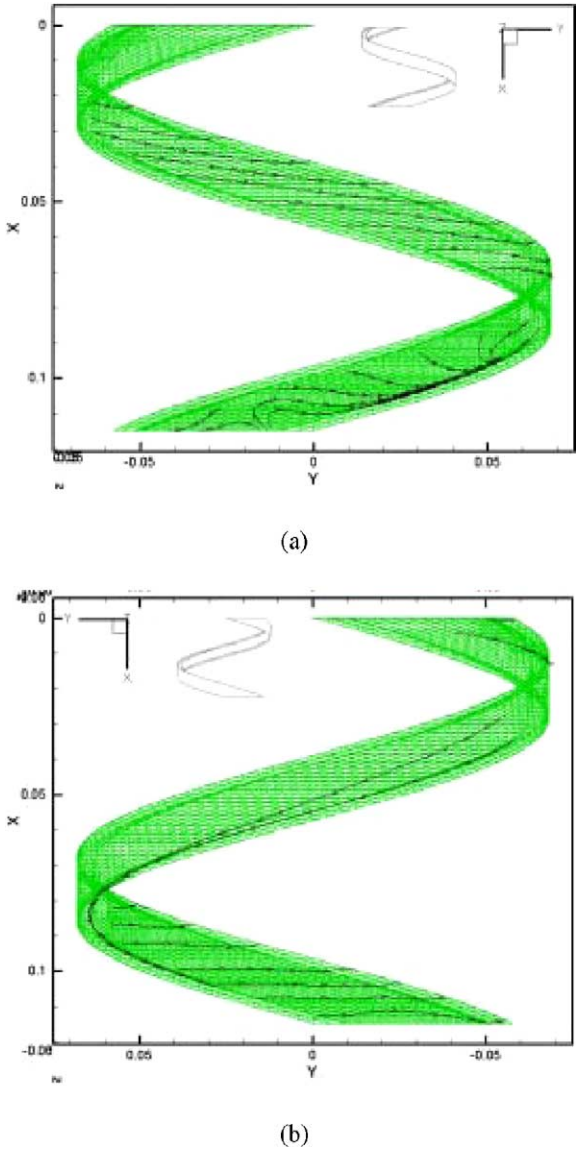

Fig. 20. Velocity vector plot on the surface of the spiral groove ($P_e/P_i = 100$), viewing at two angles by rotating 180° along x-axis.

pressure boundaries iteratively by assuming the equilibrium states at both inlet and exit. Besides, 300,000 time-steps are used to sample the particles for obtaining macroscopic properties. The reference (minimum) time-step is 2.18E−07.

3.3.2. *Evolution of adaptive mesh*

The flow is modeled by simulating a single groove only by taking advantage of the periodicity of the grooves at the circumference of the rotor. In addition, rotational coordinate system is taken on the rotor such that the rotor remains still while the casing rotates in the opposite direction. The resulting simulation particles are about 1.6 millions at steady state and the number of cells is approximately 0.42 millions (initially 27,000 cells) after 2 levels of mesh refinement using $Kn_{cc} = 1$. Note that the mesh at the very small gap, where the periodic boundary conditions apply, refrains from mesh adaptation to make the

handling much easier when simulation particles move out of these boundaries. Fig. 18(a) shows the surface mesh, while Figs. 18(b) and 18(c) illustrate the exploded view of the surface mesh near the inlet and exit, respectively, where the mesh is refined to meet the adaptation criteria. It is obvious that the mesh near the wall at inlet is refined twice due to the impact of inflow and the mesh near the exit is refined twice due to the higher pressure at the exit. Fig. 19(a) illustrates the surface pressure contours along the groove, while Fig. 19(b) shows the same quantity by rotating 180° along x-axis direction. It shows that pressure remains approximately constant and low in the most part of the groove channel ($\sim 80\%$) except at the walls where the inflow impinges, while it increases dramatically at the end of the groove channel due to the appearance of a vortex (reversed flow), which can be seen in the plot of stream line (Fig. 20). Fig. 20(a) clearly illustrates that a strongly reversed flow occurs at the end of the groove channel in the groove close to the bottom due to very small velocities and high adverse pressure gradient. One the contrary, the flow is not reversed at the end of the groove channel near the case since the velocities are much larger due to the dragging of the high-speed rotating case (Fig. 20(b)). In addition, the computational throughput of this case is 32.162 lPa/s.

4. Conclusions

In the current study, a parallel three-dimensional DSMC method combining variable time-step and unstructured adaptive mesh is presented. In the parallel DSMC method, a multi-level graph-partitioning scheme is used to dynamically re-decompose the computational domain based on a decision policy, Stop at Rise (SAR) scheme, which determines the optimal timing for repartitioning. In addition, a variable time-step method using the concept of fluxes (mass, momentum and energy) conservation across the cell interface is implemented to further reduce the number of simulation particles and required transient iterations towards steady state, without sacrificing solution accuracy. In addition, an *h-refined* unstructured adaptive mesh with mesh quality control, obtained from a preliminary parallel simulation, is used to increase the accuracy of the DSMC solution. A supersonic flow past a sphere ($Kn_\infty = 0.01035$) is used as the verification case. Detailed analysis shows that combination of variable time-step scheme with unstructured adaptive mesh can increase the computational speedup to on order of magnitude without compromising the accuracy of the solution, in addition to the parallel speedup. A near-continuum twin-jet interaction ($Kn_{\text{throat}} = 0.00385$) in a near-vacuum environment and a spirally grooved drag pump with compression ratio $K = 100$ are used to demonstrate its applications. All results are found to agree favorably with previous experimental and simulation data wherever available.

As mentioned earlier, it is relatively fast for the serial mesh adaptation module, development in parallelizing this module and incorporating into the parallel DSMC code is currently in progress and will be reported in the very near future.

Acknowledgements

This investigation was supported by the National Science Council, Grant No. NSC 92-2623-7-009-003 and National Space Program Office of Taiwan, Grant No. 92-NSPO(A)-PC-FA06-01. The authors also would like to express their sincere thanks to the computing resources provided by the National Center for High-speed Computing of National Science Council of Taiwan. Also, sincere thanks go to Dr. Walshaw for generously providing the partitioning library, PJOSTLE.

References

[1] G.A. Bird, Molecular Gas Dynamics and the Direct Simulation of Gas Flows, Oxford Univ. Press, New York, 1994.
[2] G.A. Bird, Molecular Gas Dynamics, Clarendon Press, Oxford, UK, 1976.
[3] K. Nanbu, Theoretical basis on the direct Monte Carlo method, in: V. Boffi, C. Cercignani (Eds.), Rarefied Gas Dynamics, vol. 1, Teubner, Stuttgart, 1986.
[4] I.D. Boyd, et al., J. Spacecraft and Rockets 31 (1994) 271.
[5] K.C. Kannenberg, Computational method for the Direct Simulation Monte Carlo technique with application to plume impingement, Ph.D. Thesis, Cornell University, Ithaca, NY, 1998.
[6] Y.K. Lee, J.W. Lee, Vacuum 47 (1996) 807.
[7] Y.K. Lee, J.W. Lee, Vacuum 47 (1996) 297.
[8] A. Beskok, G.E. Karniadakis, Microscale Thermophys. Engrg. 3 (1) (1999) 43.
[9] F.J. Alexander, et al., Phys. Fluids A 6 (1994) 3854.

[10] E.S. Piekos, K.S. Breuer, Trans. ASME J. Fluids Engrg. 118 (1996) 464.
[11] R.P. Nance, et al., J. Thermophys. Heat Transfer 12 (1998) 447.
[12] J.S. Wu, K.C. Tseng, Computers & Fluids 30 (2001) 711.
[13] S. Dietrich, I.D. Boyd, J. Comput. Phys. 126 (1996) 328.
[14] J.S. Wu, et al., Internat. J. Comput. Fluid Dynamics 17 (5) (2003) 405.
[15] J.S. Wu, Y.Y. Lian, Computers & Fluids 32 (8) (2003) 1133.
[16] J.S. Wu, K.C. Tseng, AIP Conference Proceedings 663 (2003) 406.
[17] J.S. Wu, K.C. Tseng, J. Fluids Engrg. ASME Trans. 125 (2003) 181.
[18] K. Kannenberg, I.D. Boyd, J. Comput. Phys. 157 (2000) 727.
[19] M.S. Ivanov, et al., AIAA Paper No. 1998-2669.
[20] J. Peirö, et al., Felisa System Reference Manual, 1995.
[21] C.D. Robinson, Particle simulations on parallel computers with dynamic load balancing, Ph.D. Thesis, Imperial College of Science, Technology and Medicine, UK, 1998.
[22] J.S. Wu, K.C. Tseng, Internat. J. Numer. Meth. Engrg., 2004, in press.
[23] L. Wang, J.K. Harvey, The application of adaptive unstructured grid technique to the computation of rarefied hypersonic flows using the DSMC method, in: J. Harvey, G. Lord (Eds.), 19th International Symposium on Rarefied Gas Dynamics, vol. 843, 1994.
[24] R.G. Wilmoth, et al., AIAA Paper No. 1996-1812.
[25] J.S. Wu, et al., Internat. J. Numer. Meth. Fluids 38 (4) (2002) 351.
[26] D.M. Nicol, J.H. Saltz, IEEE Trans. Comput. 39 (9) (1988) 1073.
[27] D.A. Russell, Phys. Fluids 11 (8) (1968) 1679.
[28] L. Dagum, S.K. Zhu, J. Spacecraft and Rockets 31 (6) (1994) 960.
[29] C. Borgnakke, P.S. Larsen, J. Comput. Phys. 18 (1975) 405.
[30] T. Soga, et al., Experimental study of interaction of underexpanded free jets, in: Proceedings of 14th International Symposium on RGD 1, 1984, p. 485.
[31] K. Nanbu, et al., Trans. Japan. Soc. Mech. Engrg. B 57 (1991) 172.
[32] J.S. Wu, et al., JSME Internat. Ser. B 44 (3) (2001) 439.