A Voronoi Grid and a Particle Tracking Algorithm for DSMC

Frank Stollmeier and Martin Grabe

German Aerospace Center (DLR), Bunsenstr. 10, D-37073 Göttingen, Germany

Abstract. The DSMC method simulates a gas by three uncoupled steps: moving representative particles through a physical domain, performing probabilistic collisions and estimating the macroscopic state by ensemble averaging.

In order to ease computational treatment of these three steps it is convenient to discretize the space with a grid that fits into the boundaries of the physical domain. For efficient particle tracking it is useful that the cells of this grid are convex polyhedra which preferably have no indirect neighbors. To reduce discretization errors in the collision step and to take reasonable averages the cells should be nearly isotropic, whereas their volume is primarily determined by the local flow gradients. Especially the latter condition requires the density of the grid to be continuously adaptable in space and time. We show that grids derived from Voronoi diagrams fulfill these requirements very well.

Keywords: DSMC, particle tracking, mesh
PACS: 51.90.+r

INTRODUCTION

The Direct Simulation Monte Carlo (DSMC) [1] method allows computation of macroscopic flow field variables of a rarefied gas by evolving a number of representative particles in discrete time steps. Each time step consists of decoupled sub-steps, most importantly:

Convection Particle positions are deterministically updated according to their momentum. Boundary conditions are enforced, e.g. particles may interact with walls, they may be injected to or removed from the computed domain.

Collisions Particles interact with each other on a probabilistic basis.

Sampling Macroscopic flow parameters are determined from both space and time averaged particle properties.

For reasons of computational performance, each of these steps is typically carried out in a discretized physical space. Both computation time and quality of the solution impose requirements on this discretization and these requirements may even be different for each of the steps outlined above.

A diversity of meshing concepts are employed in DSMC implementations to date. Bird's DS2V/DS3V implementations [2] feature a cartesian grid with several levels of global refinement generated within a bounding box of the flow domain. Bounding surfaces are imposed on that grid. Groups of the smallest cells may be amalgamated to form regions not unlike the Voronoi regions described below. In NASA's DAC [3], surfaces are discretized independently from the flow domain by a triangular mesh and are immerged into a two-level embedded cartesian grid that fills a bounding box. The SMILE code by Ivanov et al. [4] similarly relies on a two-layer rectilinear adaptive grid. A number of DSMC codes dicretize the flow domain by unstructured tetrahedral meshes. Among these is the Monaco code [5] or the 3D-DSMC implementation by Wu et al. [6]. The very recent open-source implementation of the DSMC method in the OpenFOAM [7] distribution is able to handle unstructured, arbitrary polyhedral meshes [8].

Bird points out that spatial discretization is a main concern when applying the DSMC method to multi-dimensional problems [1] and states a number of criteria an ideal DSMC grid would have to fulfill. These may be summed up to three requirements: high computational efficiency, and adaptivity to arbitrary geometry as well as to local flow conditions.

The purpose of this paper is to demonstrate that cells in unstructured, polyhedral meshes derived from Voronoi diagrams intrinsically show many properties favourable in all major steps of a DSMC simulation. A fast particle tracking algorithm is proposed which operates very well on such grids.

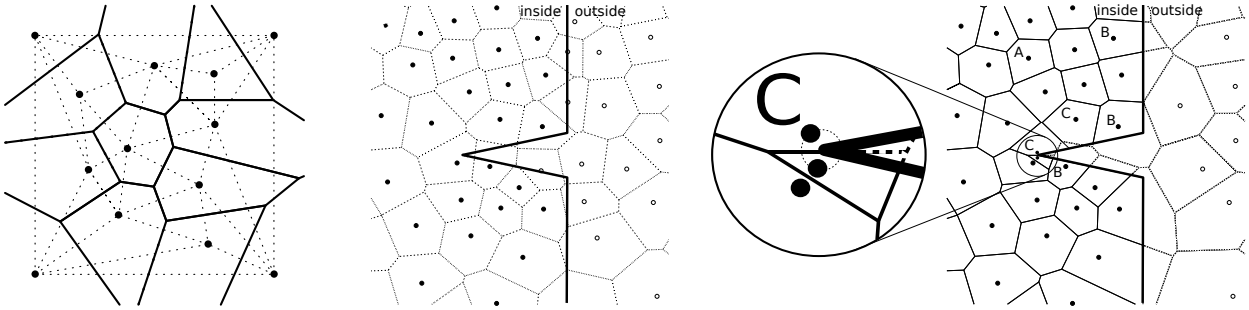

FIGURE 1. Voronoi (solid lines) and dual Delaunay diagram *(left)*, creating a Voronoi grid by imposing boundaries *(center and right)*.

METHODS

Voronoi grids

Voronoi diagrams are a conceptually simple partitioning of a space into regions. These regions are frequently described as "zones of influence". For a finite set of distinct points in a continuous space, all locations in that space are assigned to the closest point in the set. The concept is so universal, that tesselations resembling Voronoi regions can be observed in the most disparate environments and they are known for centuries in many different fields of science and economy, although not seldomly by different names. The book of Okabe et al. [9] is dedicated exclusively to Voronoi diagrams and supplements the treatment of theory and algorithms with an overview of historical and present applications.

For our purposes we conveniently restrict ourselves to the Euclidean space and let the centers of influence be points. Fig. 1 (*left*) shows an example of such a Voronoi diagram, as well as its dual, the Delaunay graph, in two dimensions. Note that the Voronoi *diagram* is not bounded, its outer regions are infinitely large. To make use of the polygonal/polyhedral cells for our purposes we need to introduce a boundary. We will call the bounded subset of the diagram a Voronoi *grid*. Creating a Voronoi grid from a Voronoi diagram can be done with two approaches. One is to mirror the sites that are closest to the boundary on a plane tangent to that boundary. This procedure is straight-forward in comvex boundaries, the geometry resolution is prescribed by the density of sites near the boundary.

Another approach is to generate a Voronoi diagram from a point set of desired density and distribution and impose a boundary on the diagram. The boundary will then generally cut existing Voronoi regions, Fig. 1 (*center*). As long as the part of the cut cell inside the domain is still convex, no extra measures need be taken. Wherever non-convex cells may arise, countermeasures are necessary. Firstly, no sites outside the boundary may claim space inside and secondly, every non-convex corner/edge is to be protected by additional sites, see Fig. 1 (*right*).

Adaptation of Voronoi grids involves simply adding or removing sites. Sites that are required for preserving the boundary information may of course not be removed. If additional sites are not randomly added but at the position of the vertices of existing Voronoi polyhedra, the cells remain isotropic as such a configuration approximates the densest packing of circles/spheres inscribed in a cell. In two dimensions such an approach preferably generates hexagonal cells, in three dimensions the average number of faces is close to fourteen.

A grid based on Voronoi diagrams offers some features that are easily implemented and may be of interest in some applications. A simple way to realize periodic boundary conditions in unstructured grids is to relink particles that reach a certain boundary and reinsert it at the corresponding opposite side, but here it has to be localized again. Fig. 2 *(left)* shows a Voronoi diagram where the positions of sites near imaginary boundaries (dashed lines) are similar. Consequently every bounding face (bold lines in Fig. 2, *left*) has a sibling of similar shape and size and they can easily be linked in the grid data structure.

The benefits of hierarchical meshes can also be made use of with Voronoi tesselations by assigning levels to the generating sites. Fig. 2 *(right)* attempts to illustrate the concept. Sites of coarser meshes are conveniently also sites in finer levels. One application is to use a coarser resolution for sampling and the finer level to compute collisions.

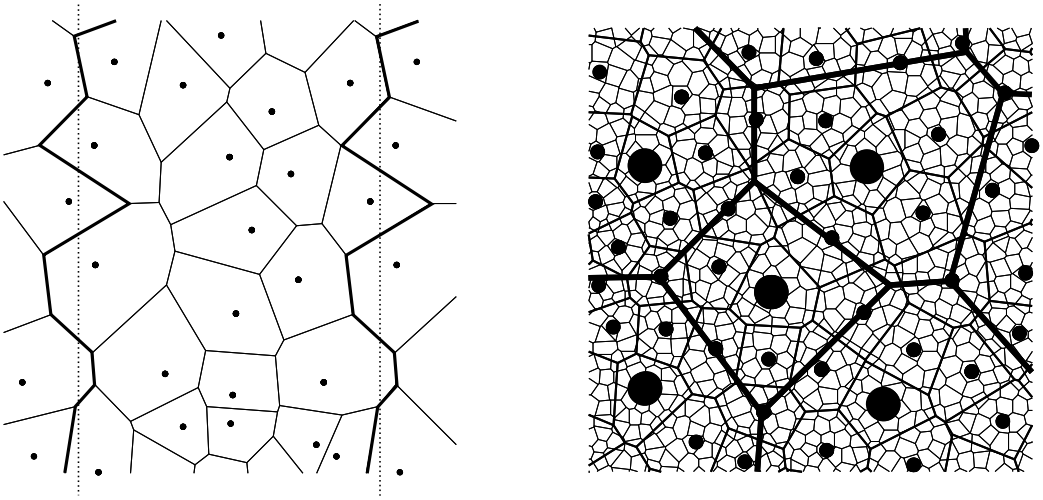

FIGURE 2. Examples of periodic *(left)* and hierarchical *(right)* Voronoi regions.

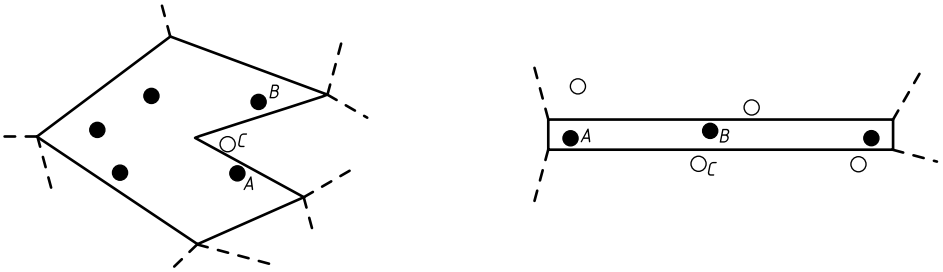

FIGURE 3. Unrealistic collision pair selection due to non-convex and anisotropic *(right)* cells.

Evaluation of cell and grid quality

Besides the global requirements for ideal DSMC *grids*, there are two desirable properties individual *cells* should possess: convexity and isotropy. In the convection step of the DSMC method, convex cells greatly simplify the algorithms for particle tracking, thus improving computational efficiency.

Convex and isotropic cells also allow a more sensible selection of potential collision partners, as Fig. 3 demonstrates by counterexample. While the restriction to forming candidate collision pairs from particles inhabiting the same cell is a price to pay for efficient computations, the process of selecting near neighbors becomes more distorted in non-convex or strongly anisotropic cells. By similar argument, convex and isotropic cells are also desirable in the sampling step.

We will restrict our discussion to unstructured grids with convex cells, as a Voronoi tesselation is guaranteed to yield convex regions. To investigate the effect of cell shape (i. e. number of faces), we will study particle motion and collision partner selection in *regular* polyhedra.

Effect of cell shape on mean transit distance

Tracking methods typically trace a particle's path by calculating which face of the current cell is intersected. Once that face is found, the particle is either linked to the neighboring cell, or boundary conditions are applied. In the perspective of a single cell, it is obviously advantageous to have as few faces as possible to test for intersection as the execution speed of the search algorithm will be proportional to the number of faces.

Combined to a grid, cells with few faces are more likely to have a high ratio of indirect to direct neighbors, thus not only nullifying the above mentioned advantage, but potentially introducing problems with proper particle location when a particle's path passes near a vertex. Fig. 4 attempts to illustrate this problem in 2D. While only one face needs

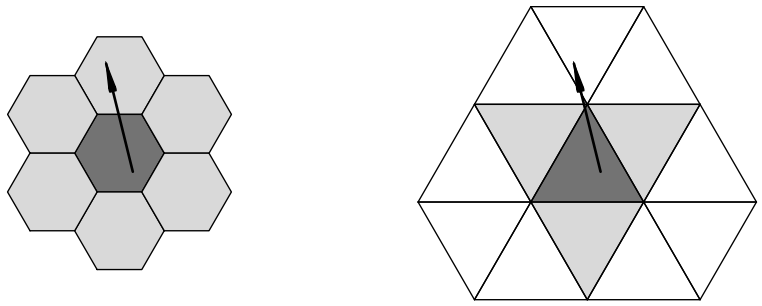

FIGURE 4. Direct (light gray) and indirect (white) neighbors of the reference cell (dark gray).

to be crossed to reach a direct neighbor, the number of faces to cross to reach an indirect neighbor is always greater, making particle tracking more costly.

The number of faces crossed when a particle travels a given distance, or correspondingly the average distance a particle can freely traverse a cell before hitting a face largely depends on the cell shape. We call this distance "mean transit distance" δ_t. A simple analytical expression can be derived for arbitrary convex polygons and polyhedra, for the sake of brevity only the line of argument is given here for the 2D case: the integral sum of the lengths of all trajectories that run parallel to each other with a certain orientation to the polygon is just the area A of the polygon. To obtain the *average* trajectory length for a certain orientation, divide A by the extent of the projection of the polygon onto the axis orthogonal to the trajectories. The mean transit distance is then simply the ratio of polygon area and the average length of all projections at all orientations. One can show that this average length of projections is just the diameter U of a circle with equal circumference. A similar argument can be constructed in three dimensions, this time examining a vonvex polyhedron with volume V and surface area A_s, to obtain:

$$\delta_t^{(2D)} = \frac{\pi A}{U}, \quad \delta_t^{(3D)} = \frac{4V}{A_s}. \tag{1}$$

We tested this surprisingly simple result by investigating particle motion in a single cell with specularly reflecting walls. In this way we simulate an infinitely large grid composed of congruent cells, regardless of wether such a grid can exist in reality.

Effect of cell shape on mean collision separation

The ratio of mean collision separation δ_c to mean free path λ was identified to be the single most important goodness parameter in DSMC simulations [2]. The magnitude of mean collision separation may depend on the shape of the container from which the collision partners are chosen and on the number of simulators N_S available in the container:

$$\delta_c = \sqrt[3]{V} \cdot f(\mathscr{F}, N_S), \tag{2}$$

where $\mathscr{F}$ is the number of faces of the regular polyhedra studied here. The original DSMC94 method involved random selection of collision partners, which made $\delta_c \propto f(\mathscr{F}, 2)$. Modern implementations employ near-neighbor selection schemes that generally lead to a reduction of δ_c with increasing N_S. Recently Bird et al. [10] studied the refinement $f(6,2)/f(6,N_S)$ in spatial resolution obtained through the more costly near-neighbor search in a cube. We extend this study by investigating other regular polyhedra.

We expect differences to be most pronounced in $f(\mathscr{F}, 2)$, as it is obvious that cell shape becomes unimportant when $N_S \to \infty$. To determine the numerical values for $f(\mathscr{F}, 2)$ a large number of points are randomly placed inside a polyhedron of unit volume. The points are uniformly distributed in the bounding box of the polyhedron. Two points that are inside the polyhedron are randomly selected and their Euclidean distance recorded. Repeating this process a large number of times yields an estimate of $f(\mathscr{F}, 2)$.

Require: $\vec{n}_i$, $\vec{s}_i$, $\vec{r}$, $\vec{v}$
  $t_{\min} \leftarrow 1$
  for all faces i **do**
    $A = (\vec{r} + \vec{v} - \vec{s}_i) \cdot \vec{n}_i$
    if $A \geq 0$ **then**
      $B = \vec{v} \cdot \vec{n}_i$
      $t = \frac{B-A}{B}$
      if not $0 \leq t \leq 1$ **then**
        $t_{\min} \leftarrow 0$
        $i_{\min} \leftarrow i$
        break for
      else if $t < t_{\min}$ **then**
        $t_{\min} \leftarrow t$
        $i_{\min} \leftarrow i$
      end if
    end if
  end for

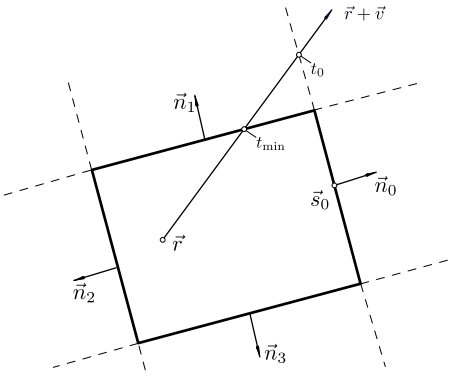

FIGURE 5. Proposed replacement algorithm for the `intersect()` routine in the algorithm of Haselbacher et al. [11]. Particle at $\vec{r}$ intersects with face 1 at a distance $t_{\min}$ *(right)*.

Particle Tracking for Voronoi grids

Particle relocation on unstructured grids typically requires tracking, as the target cell cannot be computed from an algebraic expression. Particle tracking routines exist for arbitrary polyhedral cells [8], but their tolerance for non-convex shapes imposes a performance penalty. Haselbacher et al. [11] recently published a conceptually simple algorithm, requiring cells to be convex always. The robust version of their algorithm detects and corrects faulty particle allocations that may occur due to numerical inaccuracies or adaptation. The robustness comes at the cost of performance however. The implementation of the Haselbacher-algorithm relies on a description of cell faces as planes with normal $\vec{n}_i$ and is straight-forward except for an `intersect()` method that calculates the first face $i_{\min}$ intersected by the trajectory and the distance to this face. The robust version conducts a "particle-in-cell" test before the move is carried out, which is an extra step and unnecessary in the majority of cases.

We modify the `intersect()` routine and propose to check wether the particle *target location* is outside the cell. If also the initial location is outside the cell bounds, the particle is not moved but instead reassigned to the cell associated with the face at which the inconsistency is first discovered, making the algorithm robust against faulty particle allocation. The algorithm is sketched in Fig. 5. Time t is in units of time step, $\vec{s}_i$ is a point on face i, $\vec{v}$ is the particle's displacement and $\vec{r}$ its initial position.

RESULTS

We exemplarily assess cell quality of the Platonic solids with unit volume. They are regular, convex polyhedra with 4, 6, 8, 12 and 20 faces. From (1) one can see that the average distance δ_t of a particle completely crossing a convex cell on a straight trajectory only depends on the ratio of volume to surface area and thus on the shape of the cell. For the polyhedra under investigation δ_t can be expressed analytically. A numerical simulation agreed very well with the predictions.

To compare the effect of cell shape on mean collision separation δ_c, we compare the respective worst case, that is the mean random separation of two points inside a polyhedron. The general variation of both δ_c and δ_t with the number of faces is shown on the left in Fig. 6.

In modern (DSMC07) collision partner selection routines employing a near-neighbor search the effect of cell shape is less pronounced and vanishes as the number of simulator particles gets very large. When comparing to tetrahedral cells with a population from 10 to 100 simulators, cell size may be increased by 10% already when using cubical cells and up to 15 to 20% for dodecahedrons or icosahedrons, maintaing δ_c.

The particle tracking algorithm presented here improves the one proposed by Haselbacher et al. [11] in the performance-critical aspect of detecting face intersection. The algorithm shown here is robust insofar as it is able

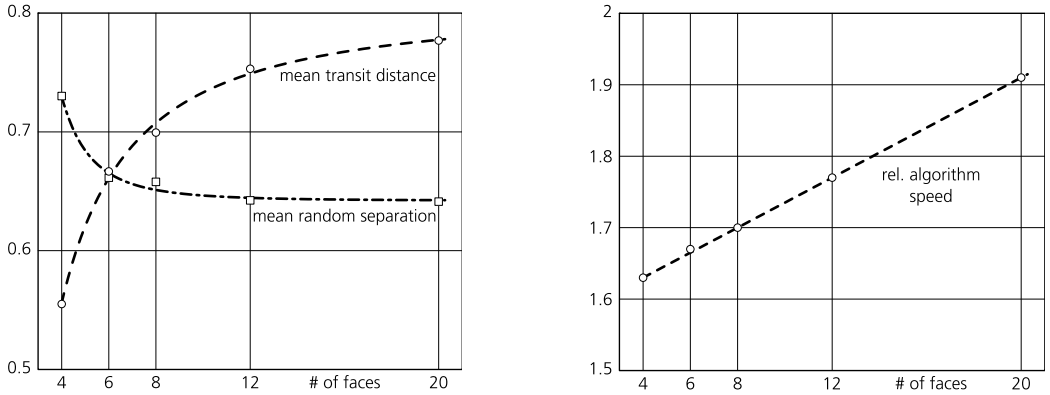

FIGURE 6. Mean transit distance δ_t and mean random separation $\delta_c(\mathscr{F},2)$ vs. number of faces *(left)*, performance of particle tracking with new algorithm relative to [11] for a traversal of 20,000 cells *(right)*.

to deal with misallocated particles. When compared to the robust version described in [11], a speedup of over 60% to more than 90% is to be expected, depending on the average number of faces per cell.

DISCUSSION

The results of cell shape analysis show that it is advantegous to employ more isotropic cells, i. e. cells with a higher number of faces to achieve a better mean collsision separation and avoid unnecessary treatment of face intersections in particle tracking. When Voronoi sites are distributed to approximate the densest packing of the cell's inspheres, the average number of faces per cell is about 14 (6 in two dimensions). Note that while absolute values for δ_t and $\delta_c(\mathscr{F},2)$ will change for distorted (anisotropic) cells (lower δ_t and higher $\delta_c(\mathscr{F},2)$), the trend will be the same.

The improved routine to determine which face a particle intersects always checks all faces in a cell and a linear dependence of performance on the number of faces is thus to be expected. The improvements made affect each pass in the loop over all faces, so that the relative performance is getting better the more faces are checked.

CONCLUSIONS

Returning to the three properties of an ideal DSMC grid stated earlier, we summarize that Voronoi grids can adapt to arbitrary boundaries as well as local flow conditions while preserving the favourable isotropy and convexity properties of the cells. Combined with an efficient particle tracking algorithm, we conclude that Voronoi grids comply very well with the requirements of accurate, versatile and efficient DSMC implementations.

REFERENCES

1. G. A. Bird, *Molecular Gas Dynamics and the Direct Simulation of Gas Flows*, Oxford University Press, 1994.
2. G. A. Bird, "The DS2V/3V Program Suite for DSMC Calculations," in *Rarefied Gas Dynamics: 24th International Symposium*, edited by M. Capitelli, Bari, 2004.
3. G. LeBeau, *Computer Methods in Applied Mechanics and Engineering* **174**, 319–337 (1999).
4. M. S. Ivanov *et al.*, "SMILE System for 2D/3D DSMC Computations," in *Rarefied Gas Dynamics: 25th International Symposium*, edited by A. K. Rebrov, and M. S. Ivanov, St. Petersburg, 2006.
5. S. Dietrich, and I. D. Boyd, *Journal of Computational Physics* **126**, 328–342 (1996).
6. J.-S. Wu, and Y.-Y. Lian, *Computers and Fluids* **32**, 1133–1160 (2003).
7. *OpenFOAM*, OpenCFD Ltd. (2009), URL `www.openfoam.org`.
8. G. B. Macpherson, N. Nordin, and H. G. Weller, *Communications in Numerical Methods in Engineering* **25**, 263–273 (2009).
9. A. Okabe, B. Boots, K. Sugihara, and S. N. Chiu, *Spatial Tesselations: Concepts and Applications of Voronoi Diagrams*, Wiley Series in Probability and Statistics, Wiley, 2000, 2nd edn., ISBN 0-471-98635-6.
10. G. A. Bird, M. A. Gallis, J. R. Torczynski, and D. J. Rader, *Physics of Fluids* **21**, 017103, 1–12 (2009).
11. A. Haselbacher, F. M. Najjar, and J. P. Ferry, *Journal of Computational Physics* **225**, 2198–2213 (2007).