Chapter 6

Computational Meshing for CFD Simulations

Andreas Lintermann[1]

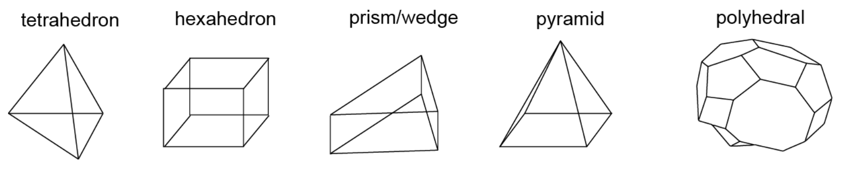
The original article is available under https://10.1007/978-981-15-6716-2_6

Abstract In computational fluid dynamics modelling, small cells or elements are created to fill the volume to simulate the flow in. They constitute a mesh where each cell represents a discrete space that represents the flow locally. Mathematical equations that represent the flow physics are then applied to each cell of the mesh. Generating a high quality mesh is extremely important to obtain reliable solutions and to guarantee numerical stability. This chapter begins with a basic introduction to a typical workflow and guidelines for generating high quality meshes, and concludes with some more advanced topics, i.e., how to generate meshes in parallel, a discussion on mesh quality, and examples on the application of lattice-Boltzmann methods to simulate flow without any turbulence modelling on highly-resolved meshes.

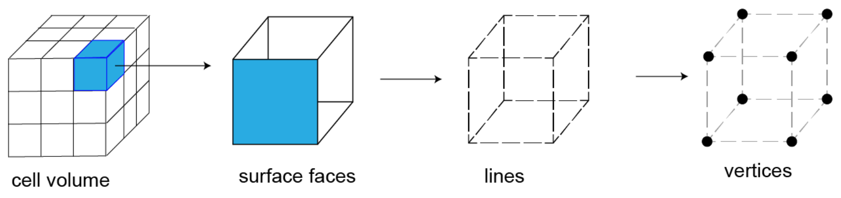
6.1 Introduction

The segmented airway region from CT/MRI scans provides surface boundary information (see Chapter 5). The internal space, where inhaled air or particles pass through, is an enclosed volume formed by the surface boundaries. In computational fluid dynamics CFD modelling, small cells or elements are created to fill this volume. They constitute a mesh and thereby discretise the space. The mathematical equations that represent the flow physics (Chapter 6, and 7) are then applied to each cell of the mesh. The end result is the ability to store time-dependent flow properties such as pressure, velocity, or temperature in each individual cell.

[1]A. Lintermann
Forschungszentrum Jülich GmbH
email:A.Lintermann@fz-juelich.de



(a) Different types of mesh elements.



(b) Mesh hierarchy.

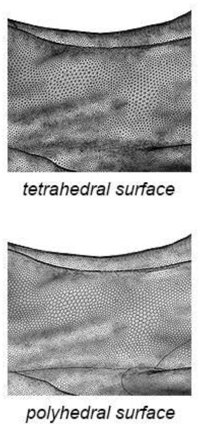
Figure 6.1: (a) Types of mesh elements including tetrahedron, hexahedron, prism/wedge, pyramid, and polyhedral. (b) Mesh topology demonstrating the hierarchy for each level of meshing.

Generating a high quality mesh is extremely important to obtain reliable solutions and to guarantee numerical stability. In the CFD community, mesh generation is also referred to as grid generation, which has become a separate discipline in itself and remains a very active area of research and development. This is evident with many existing commercial codes on the market having their own powerful, built-in mesh generators, as well as with a number of independent grid generation packages available.

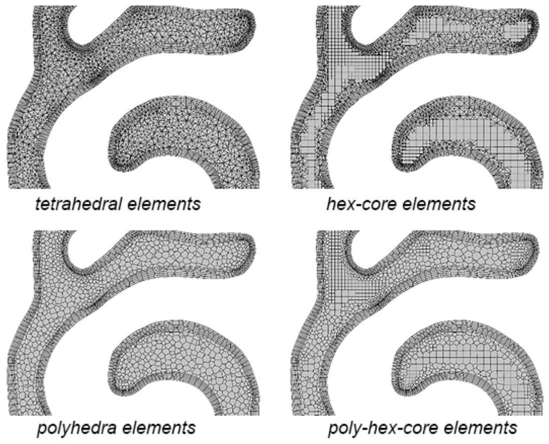
Generating a mesh for the first time is a daunting process, where decisions have to be made on the arrangement of discrete points (nodes) throughout the computational domain, and the type of connections for each point. The quality of the mesh leads to either success or failure of the numerical simulation.

6.2 Mesh Types

The nasal airway volume is treated as a 3D geometry and therefore the mesh elements/cells are generally hexahedral, tetrahedral, square pyramids (pyramids, extruded triangles, wedges or triangular prisms) or polyhedral, see Fig. 6.1a. It is useful to envision the mesh topology as a hierarchical system, where higher topology assumes the existence of topologies beneath it. Fig 6.1b shows a 3D cell inherit the lower topologies, i.e., a volume is made up from faces, a face is made from lines, and a line is made from vertices. A common meshing approach is to use a bottom-up approach, where firstly vertices, lines, and faces (from the segmentation step) are created to form a surface mesh and to subsequently fill the enclosed volume with 3D cells, i.e., the mesh elements. This means that for boundary conforming meshes care should be taken in generating good quality surfaces from the segmented airway models.



(a) Surface elements.



(b) Different kinds of internal meshes.

Figure 6.2: Unstructured meshes are typically used for complex geometries where (a) irregular shaped 2D surface elements can wrap around the surface. Thereafter, (b) the internal mesh may contain 3D elements such as tetrahedral, polyhedral, hex-core, and a combination of polyhedral-hex-core elements.

6.2.1 Unstructured Meshes

The nasal airway geometry is highly irregular and does not fit exactly to orthogonal lines or Cartesian coordinates. As a result, an irregular unstructured surface consisting of 2D elements can be created to wrap around the contoured geometry. Figure 6.2a exemplary shows surfaces consisting of tetrahedral and polyhedral elements. The surface can internally be filled with the 3D unstructured elements of the same kind or can be a hybrid mix of different elements including structured and prism layers, see Fig. 6.2b. An unstructured grid is typically identified by irregular connectivity. To store the data for the computation, new data structures, e.g., edge based, face based, or cell based structures are needed to hold the connectivity scheme. A node of a cell can be connected to any number of nodes from its neighbours, which may lead to complex connectivity dependencies. The descriptions of some of these connectivity algorithms include Delauney, advancing front, and quadtree/octree (see Ch. 6.3).

Unstructured CFD meshing technologies have matured and their use has become more prevalent with commercial CFD vendors. They provide the user with simpler automated methods in achieving a meshed model and a user-friendly interface [2]. The typical workflow of generating an unstructured mesh involves:

– detection of the edges of a surface and establishment of nodes to form the edges of each surface

– creation of new nodes on the surface and connecting them to fill the surface with

[2]A large number of grid generation software and open source codes exist with a listing of some available software given in the Appendix of this book.

2D elements such as triangles or polygonal 2D shapes (advancing front method - an example is shown in Fig. 6.3)

– for 3D geometries the interior volume (and in 2D the interior area) is filled and all the elements are reconnected (Delauney method).

Recently, there has been considerable development towards the creation of meshes containing polyhedral cells in the interior domain. Such meshes have the flexibility of an unstructured mesh and do not suffer from the overhead associated with large tetrahedral meshes. The application of such cells is still in its infancy. Nevertheless, polyhedral meshing has shown to have tremendous advantages over tetrahedral meshing with regard to the attained accuracy and efficiency of corresponding numerical computations.

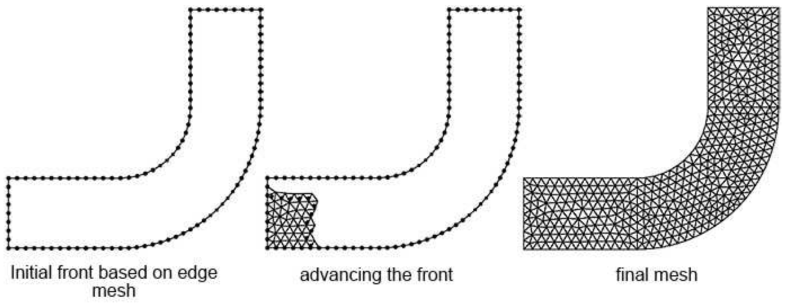


Figure 6.3: Mesh construction for a 2D 90° bend pipe based on the advancing front method. The initial front is established on the edge mesh setting. It advances by recursively adding new points and to subsequently creating a triangular element for each added point.

6.2.2 Structured Meshes

By definition, a structured mesh is one that has eight nodal corner points in three dimensions and is characterised by regular connectivity. It uses a Cartesian system with an (i, j, k) indexing to locate neighbouring cells. On a regular orthogonal geometry, creating a structured mesh is the simplest technique to undertake, since a Cartesian mesh can be conveniently generated. Figure 6.4 shows an example of a structured mesh with the corresponding indexing. When the geometry exhibits curvatures, as for example shown in Fig. 6.5, an orthogonal structured mesh applied to the geometry requires adjustments resulting in staircase-like steps in the region of the curved sections. These kinds of meshes require additional treatment to accurately prescribe boundary conditions in a numerical simulation and to accurately compute wall-shear stresses, heat fluxes, and boundary-layer effects. An approach to overcome these issues, is the generation of a body-fitted mesh. Therefore, an orthogonal mesh is applied to the geometry and a mapping of the distorted region in physical space onto a rectangular region in the curvilinear coordinate space is performed. This is in general done through coordinate transformation functions. For example, creating a body-fitted mesh for a

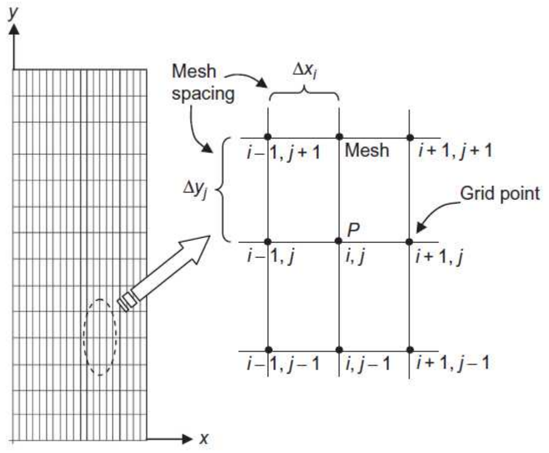


Figure 6.4: Ordered indices for a structured mesh of a 2D orthogonal domain.

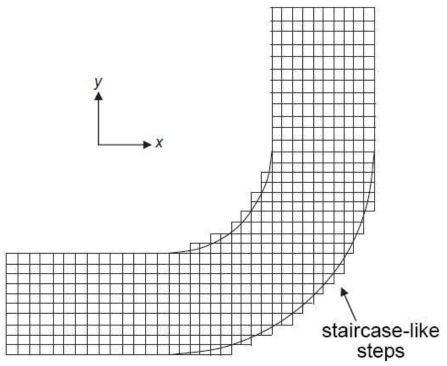


Figure 6.5: An example of a mesh using staircase-like steps for a 90° bend geometry.

90° bend geometry, the walls need to coincide with lines of constant η, see Fig. 6.6, while in the streamwise direction of the flow lines correspond to specific values of ξ in the computational domain. The simulation is performed on the curvilinear coordinates rather than on the Cartesian coordinates. Therefore, a transformation function is used to switch domains from $f(x, y)$ to $f(\eta, \xi)$ to yield new independent variables. Another technique to adapt structured meshes to non-orthogonal geometries is to use block-structured or multiblock meshes, where a number of structured blocks is assembled and connected. This method is effective for complicated shapes, where it is difficult to apply a single block. Take for example the circle shown on the left side in Fig. 6.7, where a body-fitted mesh leads to highly skewed and deformed cells at the perimeter of the geometry. These highly skewed cells generally lead to numerical instabilities and erroneous results in a simulation. Using a block-structured mesh, as shown on the right side of Fig. 6.7, the outer perimeter takes the shape of an annular grid. Hence the term O-grid has been established for these kind of meshes. Additionally, a square block fills the centre of this mesh. Other block-structure topologies include O-, C-, or L-grids.

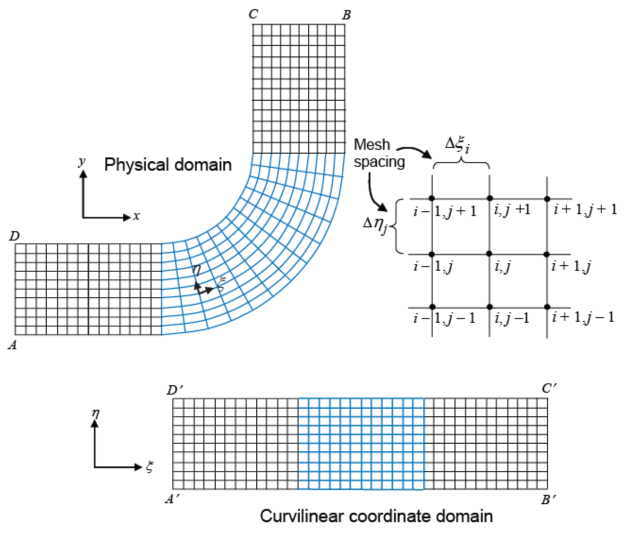


Figure 6.6: Body-fitted mesh applied to a curved geometry. A transformation of coordinates is requires.

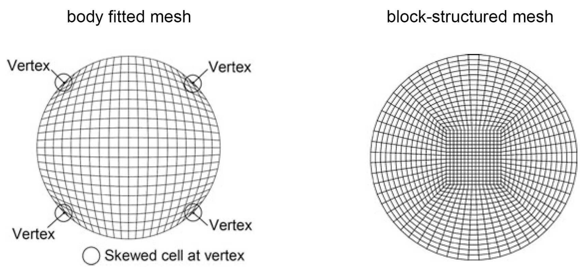


Figure 6.7: A body-fitted mesh which creates skewed cells, and a block-structured mesh (using O-grid)

An O-grid has lines of points where the last point wraps around and meets the first point, hence creating a circular 'O' shape. Similarly, a C-grid has lines that bend in a semicircle shaped like the letter 'C' (essentially half an O-grid), while an L-grid forms an L-shape derived from a quarter of an O-grid. Examples are depicted in Fig. 6.8. Further examples to overcome the issue of staircase-like approximations of curved walls are applying a cut-cell method, i.e., reshaping the staircase elements to fit the geometry, or to include a treatment in the computation, cf. [5, 31, 49]. This approach is also followed by the meshing method presented in Ch. 6.3, where a technique to generate hierarchical Cartesian meshes for nasal cavity geometries in parallel is described.

6.2.3 Structured Versus Unstructured Meshes

A structured mesh has the advantage that its layout is simple due to the direct and straightforward connections between adjacent cells. For example, a three-dimensional cell is connected by six neighbouring cells with points easily accessible by triple indices (i, j, k), incrementing or decrementing with each adjacent cell. This indexing makes CFD solvers simple to program and leads to a small memory footprint. However, the

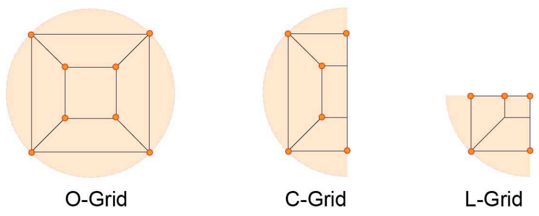


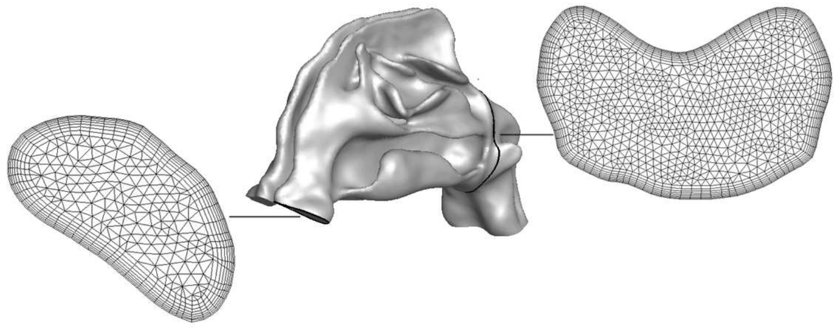
Figure 6.8: Block structures used to map onto curved sections of a circle (O-grid), semi-circle (C-grid), and quarter-circle (L-grid).

creation of structured meshes for complex geometries such as the nasal airway is highly time consuming as it involves a lot of manual work. Therefore, in most literature, unstructured tetrahedral, hexahedral, or polyhedral meshes are used (see, e.g., the meshes depicted in Fig. 6.9). An unstructured mesh has the advantage that it is much more suited to fit within complex shapes. Furthermore, the volume filling can, e.g., be performed automatically using a Delauney triangulation or a hierarchical approach, see Ch. 6.3. In regions of high curvatures, however, tetrahedrals tend to become over-stretched leading to high aspect ratios that affect the skewness of the cells - this is of course not the case for hexahedral cells. Additionally, tetrahedral cells can be difficult to align with the flow direction. These two problems can impede convergence and may lead to artificial errors in the solution, known as numerical diffusion. This common source of error is also called false diffusion as it is a product of numerical error and does not represent a physically occurring phenomenon. The other major drawback of unstructured meshes is the treatment of the nodal points of any cell. Unlike in structured meshes, they cannot be accessed by triple indices (i, j, k) in three dimensions. Instead additional data calculations are needed to connect an arbitrary number of neighbouring nodes. This requires additional computational memory and further complicates the solution algorithms that are used to solve the flow field variables. This typically results in increased computational times compared to structured-mesh-based codes to obtain a solution.

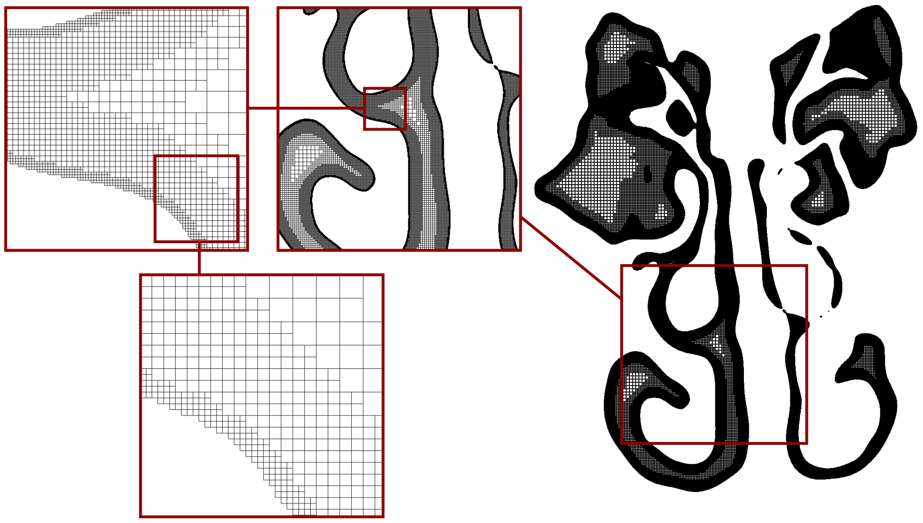
In the near-wall region, both types of meshing algorithms need to achieve a resolution that has reasonable accuracy in resolving boundary layer flows. This is achieved by either adding prism layers along walls or by refining hexahedral cells in hierarchical manner. An example for mixed hybrid meshes consisting of tetrahedra and prisms in the vicinity of the walls is shown in Fig. 6.9a. Another example, where a hierarchical Cartesian mesh is boundary-refined, is shown in Fig. 6.9b.

6.2.4 Polyhedral Meshing

An alternative to tetrahedral meshes is the use of polyhedral elements, which have become increasingly popular [40]. The major advantage of using polyhedra compared to tetrahedra is their larger number of adjacent neighbouring cells (typically ten, compared to four for tetrahedra). This allows greater resolution in gradient approximations. Polyhedra can be viewed as cells created from combining tetrahedral cells. Thereby, the overall cell count is decreased [10] and highly skewed cells are reduced. This makes polyhedral cells less sensitive to stretching due to their irregular shape. Due to the increased number of nodes and faces per cell, and increased number of neighbours,



(a) Tetrahedral mesh for the inner core region and prism layers lining the wall region.



(b) Boundary-refined computational mesh consisting in total of $1.8 \cdot 10^9$ elements [33]. Only every second grid line is shown.

Figure 6.9: Two types of boundary-refined meshes.

polyhedral meshes require more computing operations per cell. This is, however, compensated by the rapid time to reach convergence.

6.3 Parallel Mesh Generation of Hierarchical Cartesian Meshes

The intense increase of computational power over the last decades has enabled simulation software to simulate physical phenomena at continuously growing resolution and accuracy. The pace of algorithm development lacks behind the pace of hardware development and hence the full potential of todays computer architectures is in many cases left unexploited, especially when it comes to CFD simulations.

At small scale, the generation of computational meshes in serial for simple geometric shapes is not a big challenge and efficient algorithms already exist for quite some time [52, 53]. The considered geometries become, however, continuously complex and in general the complexity of the topology of computational meshes follows this behaviour. This more and more necessitates to write efficient parallel algorithms that can create such computational meshes, e.g., for intricate anatomical shapes such as the nasal cavity, consisting of hundreds of millions up to billions of elements [6, 22, 33, 36]. Since memory limitations prevent the generation of such meshes in serial, algorithm developers need to think about multi-level communication-reducing parallelisation strategies, decompositioning methods, load-balancing, and parallel I/O. This renders corresponding software development an issue of computer science, and to be more precise, of high-performance computing (HPC).

Hierarchical Cartesian meshes are an exceptional tool to discretise the space for CFD computations. They allow for a straightforward mesh generation, their rectangular grid lines enable to easily apply high-order schemes, and refinement and coarsening can be performed on the fly [6, 33]. Latter feature is in combination with a dynamic load-balancing scheme a powerful tool to run large-scale balanced simulations on HPC systems [21, 48]. Their drawbacks result from their infeasibility for simulations of flow around bodies at high REYNOLDS numbers with small boundary layer thicknesses. That is, the near-wall region is populated with a massive amount of mesh elements in streamwise direction although the gradient, e.g., of the fluid velocity, in this direction is rather small. Boundary-fitted meshes or meshes combining near-wall prism layers and tetrahedral elements are in such cases more advantageous [38, 39], require, however, in many cases either intense manual work or their generation is quite complex and not easy to parallelise.

Since the REYNOLDS number is rather low in respiratory flow, i.e., it is usually on the order of $Re \in \mathcal{O}(10^4)$ [17, 18, 30], hierarchical Cartesian meshes are good candidates to simulate the corresponding flow. In the following, a parallel mesh generation technique is described that enabled the generation of such meshes and scales up to $\mathcal{O}(10^5)$ cores to generate $\mathcal{O}(10^{11})$ mesh elements. For more details, the interested reader is referred to the original publication [33], where the algorithm is first described. Application to biomedical flows can be found in [27, 30, 34, 35] and Ch. 6.6 briefly presents some examples. More non-biomedical applications can be found in [7, 41, 42, 49, 50], i.e., for the simulation of jets [7], direct particle simulations [49, 50], or rotating fan setups [41, 42].

6.3.1 Description of the Parallel Meshing Algorithm

The algorithm to generate meshes consists of a serial and a parallel part. The serial part executes the same algorithm on each message passing interface (MPI) process $n \in \{1, \ldots, N_m\}$, where N_m is the total number of spawned MPI processes on some cluster system. In contrast, the parallel part performs individual work on separate sub-domains of the output of the serial part. In the following, first the serial part is explained before the algorithmic details of the parallel part are discussed.

In the initial stage, each process reads the geometry from disk, i.e., the nasal cavity surface, which is stored in standard tessellation language (STL) format and usually stems from computer tomography (CT) images [35], cf. Ch. 5. From the triangles an alternating digital tree (ADT) [4] is generated to efficiently retrieve triangle `ids` for triangle / mesh element intersections and inside / outside detections. Around the geometry an initial cube element on a hierarchy level l_0 with an edge length $\delta x(l_0)$ of the maximum extent of the geometry is generated. A subdivision algorithm continuously splits this cube element into its eight children and creates parent-child relationships and element-neighbourhood information. This constitutes a mesh hierarchy representing an octree in three dimensions, see Fig. 6.10.

This refinement is continuously performed until a mesh with levels l_0 up to a user-defined level l_a exists. In each iteration, elements that live outside the geometry, i.e., cells that do not need to be considered in the computation, are removed from the octree. This is done by identifying the elements that intersect the geometry and use those as a boundary for a recursive flagging algorithm that separates inside from outside cells. The outside elements are then deleted from memory. Subsequently levels $l < l_a$ are removed from the octree, a space filling Hilbert curve [46] is placed into the remaining elements, and the Hilbert `ids` are used to subdivide the mesh for further parallel processing. That is, each MPI process removes those elements it is not responsible for, but keeps those elements that are neighbours to the remaining elements. These elements, so called halo elements, are in a later stage responsible to create a consistent neighbourhood information across all MPI processes before the mesh is written to disk.

In the subsequent parallel phase, each MPI process continues to uniformly refine its local mesh hierarchy to a level $l_b > l_a$ under removal of the corresponding outside elements. Since it is desirable to have a high resolution in the vicinity of the nasal cavity tissue surface or in regions with high velocity gradients, e.g., for an accurate computation of wall-shear stresses, the mesh is furthermore region-refined. Two distinct methods can be used separately or in combination. Patch refinement allows the user to define

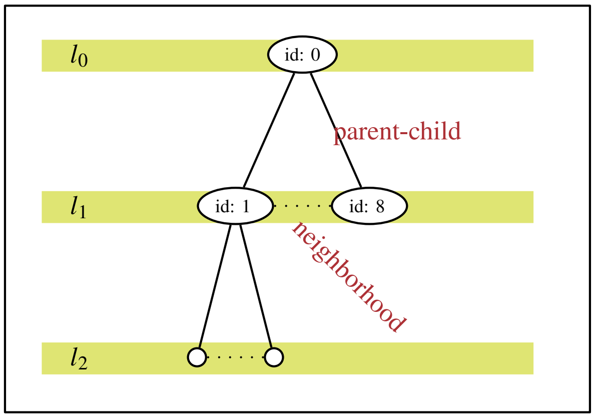


Figure 6.10: Hierarchical octree representation of the Cartesian mesh with parent-child relationships and neighbourhood information. Levels l_0, the initial cubic element around the geometry, and levels l_1 and l_2 are shown.

volumetric regions in space that are to be further refined. Elements residing in these regions are flagged for further subdivision until a level l_c^α is reached. In contrast, boundary-refinement starts by measuring the distances in units of elements of level l_b from the wall and identifies those elements with a distance $d \leq d_u$, where d_u is a user-defined constant. These cells are then further refined. Based on a distance rule, the measurement and refinement is repeated on the subsequent levels until a fine resolution on level l_c^β is reached. It should be noted that since the distance measurement is a recursive process it needs to take distance information from neighbouring MPI processes into account. While for the boundary refinement the restriction to have only level distances of maximum 1 between neighbouring elements is implicitly solved by applying the distance rule and providing a decent value for d_u, in patch refinement the user is responsible to guarantee a mesh output that is usable for computations. The algorithm features a load-balancing strategy to avoid imbalances due to excessive local refinement and out-of memory failures.

In the final stage, the global neighbourhood and a unique identification for all elements is generated across MPI boundaries via the halo elements on all levels. The number of elements in the subtrees of the elements on all levels are counted and used as for a weighting measure. The elements on refinement level l_a are used to generate a list $\mathcal{L} = \{e_1, \ldots, e_k\}$. The simplest approach is placing all elements on level l_a into $\mathcal{L}$, leading to a Hilbert-sorted list. In a subsequent simulation, the entries of the list $\mathcal{L}$ and their corresponding subtrees are then evenly distributed among the MPI processes. However, undesirable weightings on level l_a that exceed a certain threshold, e.g., an element $e^+ \in \mathcal{L}$ that has compared to other elements on level l_a many more descendants in the octree, might cause an imbalance at simulation start. Such cases are treated by replacing e^+ by its child elements $c(e^+)$. That is, $\mathcal{L} = \{\ldots, e^+, \ldots\}$ is replaced by $\mathcal{L}^\sim = \{\ldots, \{c(e^+)\}, \ldots\}$. The mesh and the list $\mathcal{L}^\sim$ are finally written to disk using parallel I/O with parallel NetCDF [26] or HDF5 [11]. The option to write a parallel geometry file to also distribute the triangles of the STL among the MPI processes in the simulation exists at this stage as well [27].

6.3.2 Meshing Example and Parallel Performance

The algorithm can now be employed to generate large-scale meshes with billions of elements on HPC systems. The availability of list $\mathcal{L}$ at the end of the meshing process allows to use a quasi arbitrary number of MPI processes N_s for the simulation with $o \leq N_s \leq |\mathcal{L}|$. The number N_s is independent from the number of MPI ranks used for the mesh generation N_m and has a lower bound of o, which is determined by the amount of available memory per process, i.e., at process number o it is assumed that the memory is completely exhausted. Figure 6.9b shows an example of a mesh of a nasal cavity with $1.8 \cdot 10^9$ elements with boundary-refined regions. The base level is at $l_a = 8$ and the uniform refinement level at $l_b = 10$. On the finest level $l_c^\beta = 13$ the mesh resolution is at $12.651 \cdot 10^{-3} mm$. The parallel performance can be analyzed by considering the strong scalability of the meshing process and the absolute run time for several cases. Such analyses were performed on the HERMIT[3] and JUQUEEN[4] HPC

[3]HERMIT is the predecessor of the currently installed HAZEL HEN system at HLRS Stuttgart
[4]JUQUEEN is the predecessor of the currently installed JUWELS system at JSC

systems at the High-Performance Computing Center Stuttgart (HLRS) and the Jülich Supercomputing Centre (JSC) for cubic domains. The meshes to be generated consisted of $C_1 = 9.82 \cdot 10^9$, $C_2 = 78.54 \cdot 10^9$, and $C_3 = 0.64 \cdot 10^{12}$. On HERMIT, cases C_1 and C_3 were tested, where C_1 was scaled from $4,096$ up to the full machine with $112,768$ cores and C_3 was generated on the full machine. A scaling for the latter case was not possible due to memory limitations. On JUQUEEN, cases C_1 and C_2 were tested, i.e., case C_1 was scaled from $4,096$ up to $131,072$ cores ($1/4$ of the system) and C_2 from $32,768$ up to $262,144$ cores ($1/2$ of the system). Figure 6.11 shows the results of the scaling experiment. Obviously, case C_1 scales well across both systems up to $32,768$ cores and experiences a slight drop in efficiency up to the maximum tested core counts. The JUQUEEN system performs slightly better than the HERMIT from a scalability point of view, the wall clock time is, however, better on HERMIT, i.e., it is $20.95s$ on $112,768$ HERMIT cores compared to $27.63s$ on $131,072$ JUQUEEN cores. Considering case C_2, a perfect scalability is visible up to $262,144$ cores and even a slight superlinear behavior is visible for both core counts $131,072$ and $262,144$. The fastest execution is measured for $262,144$ JUQUEEN cores with $47.21s$. The large mesh C_3 is generated in $267.99s$ on the full HERMIT.

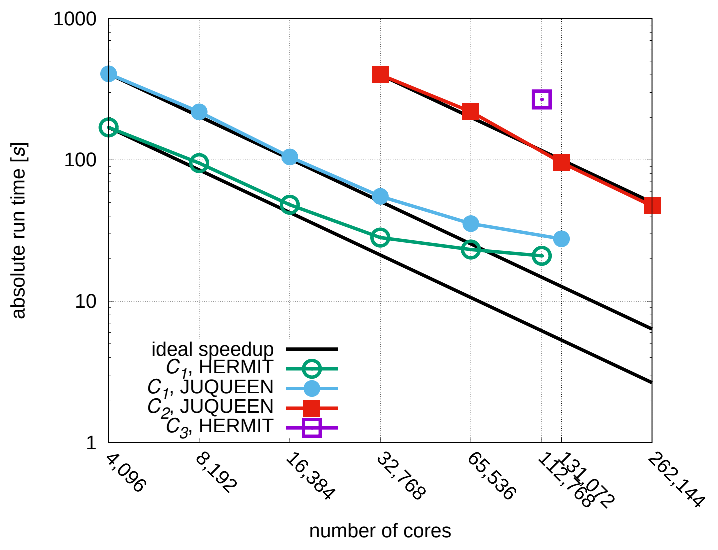


Figure 6.11: Results of strong scaling experiments on the HERMIT and JUQUEEN HPC systems at HLRS and JSC. Three different cases with $C_1 = 9.82 \cdot 10^9$, $C_2 = 78.54 \cdot 10^9$, and $C_3 = 0.64 \cdot 10^{12}$ elements were considered to mesh cubic domains [33].

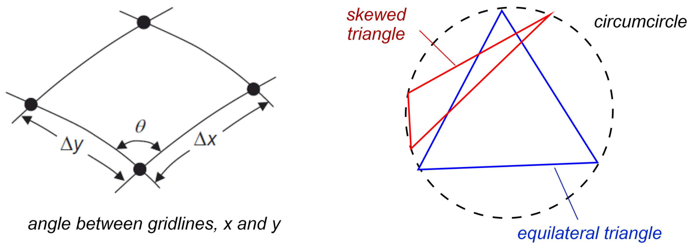
6.4 Quality of Meshes

Generating a quality mesh is a challenging step, which can be considered as an art on its own. Frequently, users' experiences in mesh design dictate the final mesh quality.

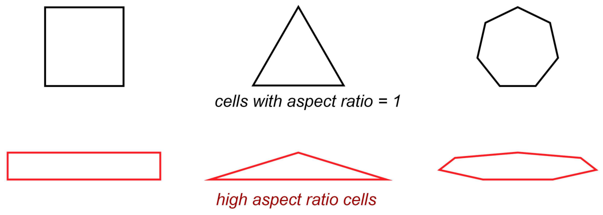
There are, however, some general guidelines that can help to form a good basis to work from.

6.4.1 Skewness and Aspect Ratio

One primary source determining mesh quality is the cell shape, which can be characterised by its skewness and aspect ratio. Skewness or grid distortion, relates the angle θ between grid lines, where angles $\theta \approx 90°$ (orthogonal) are desired, see Fig. 6.12a. This is obviously the case for Cartesian meshes. For other types of elements, the mesh is considered skewed if it is $\theta \leq 45°$ or $\theta \geq 135°$. This classification can be applied to all types of cells and face shapes, and can always be applied to prism layers. For triangles and tetrahedra, the skewness is determined by relating the actual shape to their equivalent equilateral triangle and tetrahedron.



(a) Mesh skewness determined by the angle made between gridlines or based on the equivalent area (volume in 3D) of the actual cell relative to its equilateral cell.



(b) Differently shaped cells with aspect ratios of 1.0 and their respective high aspect ratio cells.

Figure 6.12: Skewness and aspect ratio of mesh elements.

Large aspect ratios should always be avoided in important flow regions, where rapid changes, e.g., due to jets, flow separation, attachment and recirculation, occur. Distorted cells in these regions can degrade the solution accuracy and may result in poor convergence. An aspect ratio in the range of 0.2 to 5 within the interior region is desirable, while for boundary layers in the vicinity of a wall, the condition may be relaxed. In Cartesian meshes with uniform grid distance δx in all directions, the aspect ratio is obviously always 1.0. Figure 6.12b shows an examples of different aspect ratios for different kinds of cell types.

To summarise, a quality mesh should ensure that the following conditions hold:

– minimal cell skewness

– minimal changes in cell volumes between adjacent cells

– suitable aspect ratios (depending on the cell location)

– sufficient resolution in regions of rapid flow changes, e.g., in regions with sharp gradients or boundary layers

6.4.2 Mesh Independence

Mesh independence refers to a mesh being optimised to have the minimum number of cells to obtain a solution that will not change if any further mesh refinement is made, i.e., the solution becomes independent of the resolution for decreasing element sizes and increasing numbers of cells. This typically involves monitoring a fluid flow parameter of interest, e.g., the velocity or the pressure, under successive grid refinement. Ideally, at least three significant, different grid resolutions should be evaluated, where each subsequent mesh is refined or coarsened. Upon refinement, the grid cells become smaller and the number of cells in the flow domain increases. If a global mesh refinement is not feasible, selective local refinement of the mesh in critical flow regions should be applied, cf. Ch. 6.2.3.

In the CFD community, mesh independence is evaluated by estimates of the discretisation error using Richardson extrapolation [44]. More recently, the grid convergence index (GCI) was introduced by Roache [45] as a method to uniformly evaluate the quality of CFD results. Inthavong et al. [20] propose an alternative method that uses a 2D cross-sectional plane subtraction method, where scalar values, e.g., the velocity magnitude, from a cross-sectional plane are interpolated onto a regularly spaced grid. The interpolated grid values from any two meshed models can then be compared by subtracting the values from each cell. In [20], six planes across the nasal cavity are used to represent the entire domain. The method is also extended to 3D by introducing a volume subtraction.

Performing mesh independence studies on a small number of patient models is feasible. However, where large numbers of patients, e.g., over 20, need to be treated, the task of performing these studies becomes extremely time consuming. In Ch. 1, Fig. 1.2b, the number of mesh elements used for the CFD simulations increased from order of tens of thousands of elements to millions of elements. The large variations in mesh numbers among different studies has led to investigations of what constitutes a sufficiently resolved mesh. Vinchurkar and Longest [54], Frank-Ito et al. [12], and Bass and Worth Longest [1] recommended prism cells meshes to resolve the near wall boundary layers, with Frank-Ito et al. [12] suggesting the use of 4 million tetrahedral cells with three prism layers for near-wall regions.

It should, however, be noted that a general estimation of the number of cells required to accurately run a simulation cannot be given. Flow phenomena such as jets, flow separation, and even turbulence strongly depend on the considered Reynolds number, which might locally vary and hence require an additional increase of the resolution. Furthermore, the method to solve the governing equations of fluid mechanics also plays an important role. While RANS (Reynolds-Averaged Navier-Stokes) computations might already reach grid independence at a relatively low numbers of cells, large eddy simulations (LES) and direct numerical simulations (DNS) require a massive amount of cells. This is examplarily shown for a highly-resolved nasal cavity simulation in Ch. 6.6, where a sub-millimeter resolution and a number of cells $n > 100 \cdot 10^6$ is required [30, 35].

6.5 Advanced Mesh Applications: Lattice-Boltzmann Methods to Simulate Complex Flows in Complex Geometries

Many different numerical methods exist to simulate complex compressible or incompressible flow in intricate geometries. Fortunately, the flow in the nasal cavity is incompressible and it is not necessary to consider MACH numbers M that exceed the incompressibility regime limit of roughly $M \approx 0.3$ [17, 18, 30]. A straightforward approach is to numerically solve the governing equations of fluid mechanics, i.e., the Navier-Stokes equations, directly by means of, e.g., finite volume, finite difference, finite element, or discontinuous Galerkin methods. In these methods, in the incompressible limit, it is required to equate the pressure by solving the pressure Poisson equation.

In the last few decades, the lattice-Boltzmann (LB) method gained a lot of popularity. Instead of describing the fluid mechanics on a macroscopic level as done by the above mentioned computational methods, the LB method considers the statistics of particle interactions and distributions in a finite volume and derives the macroscopic flow variables from the statistics. The standard LB methods are quasi-incompressible, do, however, not require to solve a Poisson equation for the pressure as the pressure is an implicit solution of the numerically solved LB equation. Both DNS, which resolves all scales, and with some extensions, LES, which models non-resolved scales, can be performed with LB methods. Furthermore, the compute kernels can easily be parallelised leading to high parallel efficiency. In the LB method, complex boundaries can be treated in a straightforward manner with a high order of accuracy. This excels the LB method for the simulation of intricate flows in complex geometries such as the human respiratory tract.

In the following, a brief introduction is given to the physics of fluids from a statistical point of view and to different LB methods that are suited for the simulation of respiratory flows including LES modelling approaches. For more details on the fundamentals of fluid mechanics, the interested reader is referred to Ch. 7.

6.5.1 The Boltzmann Equation

Ludwig Boltzmann (1844-1906) developed a general equation to describe the evolution of the particle probability distribution functions (PPDF) $f(\mathbf{x}, \xi, t)$ following the laws of gas kinetics, i.e., the conservation of the number of particles and the statistical description of particle movement and collisions. The so called Boltzmann equation is given by

$$\overbrace{\underbrace{\frac{\partial f}{\partial t}}_{\text{temporal change}} + \underbrace{\xi \cdot \nabla_{\mathbf{x}} f}_{\text{convective term}} + \underbrace{\frac{\mathbf{F}}{m} \cdot \nabla_{\xi} f}_{\text{external forces}}}^{\text{transport}} = \overbrace{\int_{\xi_\beta} \int_{\Omega} \xi_r C(\xi_r, \Omega)(f'_\alpha f'_\beta - f_\alpha f_\beta)\, d\Omega\, d\xi_\beta}^{\text{collision}} . \tag{6.1}$$

The left-hand side (LHS) describes the collision step with the temporal change of the PPDFs, the convective term, and the influence of external forces. The right-hand side (RHS) describes the collision of particles in a statistical sense in an infinitesimal fluid volume. In Eq. 6.1 the quantity t represents the time, $\xi = (\xi_1, \xi_2, \xi_3)^T$ is the vector of the molecular velocity, $\xi_r = \xi_\alpha - \xi_\beta$ is the relative velocity vector between two particles with velocity vectors ξ_α and ξ_β, and $\mathbf{F}$ is the external forcing vector. Furthermore, $C(\xi_r, \Omega)$ is the differential cross section of the collision, in which the relative momenta of the colliding particles turns through an angle θ into the element of the solid angle $d\Omega$, and PPDFs with $<'>$ are PPDFs after a collision. The nabla operators ∇ with subscripts $\mathbf{x}$ and ξ represent differentiation operators with respect to the spatial directions $\mathbf{x} = (x_1, x_2, x_3)$ and the velocity directions of ξ. Via a Chapman-Enskog development the Navier-Stokes equations can be recovered from the Boltzmann equation, see e.g. [2].

Equation 6.1 is an integro-differential equation and hence hard to solve. Bhatnagar, Gross, and Krook (BGK) therefore developed the BGK equation [3], which replaces the RHS integral formulation of Eq.6.1 by a relaxation towards a thermodynamical equilibrium, i.e., the lattice-BGK equation is given by

$$\frac{\partial f}{\partial t} + \xi \cdot \nabla_{\mathbf{x}} f + \frac{\mathbf{F}}{m} \cdot \nabla_{\xi} f = \omega_c (F - f), \tag{6.2}$$

where F is the PPDF at equilibrium and ω_c is the collision frequency.

6.5.2 Computational Lattice-Boltzmann Methods

Different numerical LB methods exist. The single relaxation (SRT) and multiple relaxation time (MRT) methods, are however, the most frequently used methods. Both are, depending on the considered Reynolds number, well suited for the simulation of respiratory flows and are discussed in the following before some details on boundary conditions and extensions for LES and refined meshes are presented. Finally, results of some performance analyses are shown.

Single Relaxation Time Method

To numerically solve Eq.6.2, the equation is discretised in i-many space directions and in time, leading to the lattice-BGK equation [43]

$$f\,(\mathbf{x} + \xi_i \delta t, t + \delta t) = f_i\,(\mathbf{x}, t) + \omega_x \delta t \cdot \left(f_i^{eq}\,(\mathbf{x}, t) - f_i\,(\mathbf{x}, t)\right) \tag{6.3}$$

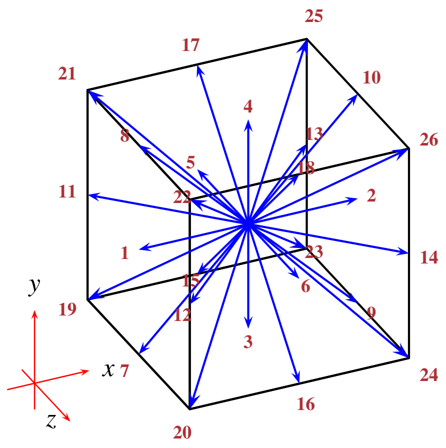


Figure 6.13: $D3Qi$ discretisation scheme in three dimensions with directions $i \in \{1, \dots, 27\}$. The remaining particle-distribution has subscript $i \in \{15, 19, 27\}$ for the models $D3Q15$, $D3Q19$, and $D3Q27$.

with discretised equilibria PPDFs f_i^{eq}, and time increments δt. The equilibrium distribution function is given by

$$f_i^{eq}(\mathbf{x}, t) = \rho t_p \underbrace{\left[1 + \frac{v_a \xi_{i,a}}{c_s^2} + \frac{v_a v_b}{2c_s^2} \cdot \left(\frac{\xi_{i,a}\xi_{i,b}}{c_s^2} - \delta_{ab}\right)\right]}_{\chi}, \qquad a, b \in \{1, 2, 3\} \quad (6.4)$$

where ρ is the non-dimensional density, t_p is a discretisation-method-direction-dependent weighting factor, $\xi_{i,a}$ and $\xi_{i,b}$ are components of ξ, v_a and v_b are components of the non-dimensional velocity vector $\mathbf{v} = (v_1, v_2, v_3)^T$, $c_s = 1/\sqrt{3}$ is the non-dimensional speed of sound, and δ_{ab} is the Kronecker delta. The discrete collision frequency is given by

$$\omega_x = \frac{c_s^2}{\nu + \delta t c_s^2/2} \quad (6.5)$$

with viscosity ν.

The number of directions i is determined by choosing a discretisation model. Qian et al. [43] developed the $DxQi$ model, which discretises the space in x dimensions with i-many directions. Examples in three dimensions are the $D3Q15$, $D3Q19$, and the $D3Q27$ models, where the former two models have been shown to not preserve rotational invariance [23, 56]. The last PPDF, i.e., $i \in \{15, 19, 27\}$ is always reserved for the rest-particle distribution. Figure 6.13 shows the directions for the different models in a Cartesian mesh setup, cf Ch. 6.3.

The derivation of f^{eq} employs a small Mach number approximation, which leads to the limitation of the LB method to quasi-incompressible flow and a decoupling of the energy equation from the equations of momentum conservation. To simulate thermal flow, the thermal LB (TLB) method is often used, which solves an additional scalar transport equation for the temperature T [16]

$$g\left(\mathbf{x}+\xi_i\delta t, t+\delta t\right) = g_i\left(\mathbf{x},t\right) + \omega_{x,T}\,\delta t \cdot \left(g_i^{eq}\left(\mathbf{x},t\right) - g_i\left(\mathbf{x},t\right)\right), \tag{6.6}$$

where the equilibrium is given by $g_i^{eq} = T t_p \chi$ and the collision frequency $\omega_{x,T}$ is a function of the heat conduction coefficient given by the PRANDTL number.

All macroscopic variables can be derived from the moments of the PPDFs, i.e.,

$$\text{density:} \quad \rho = \sum_i f_i(\mathbf{x},t), \tag{6.7}$$

$$\text{temperature:} \quad T = \sum_i g_i(\mathbf{x},t), \tag{6.8}$$

$$\text{momentum:} \quad \rho v_a = \sum_i \xi_i^a f_i^a(\mathbf{x},t), \tag{6.9}$$

where ξ_i^a are directions having a component $\xi_{i,a} \neq 0$ and f_i^a are the corresponding PPDFs. The pressure can be obtained from the density via $p = (1/3)\cdot\rho$.

From a algorithmic point of view, Eqs. 6.3 and 6.6 are split into a collision step by solving their RHSs and storing the results in a temporary array, and into a propagation step (LHS) in which this information is forwarded to all neighbouring elements.

Multiple Relaxation Time Method

For higher REYNOLDS number flows, the SRT method might suffer from instabilities [24]. Therefore, MRT methods [8] have been developed, which relax, in contrast to the SRT methods, in moment space. In vector notation, the MRT equation is given by

$$\mathbf{f}(\mathbf{x}+\xi_i\delta t, t+\delta t) = \mathbf{f}(\mathbf{x},t) - \underline{\mathbf{M}}^{-1}\underline{\mathbf{K}}_{MRT} \cdot \left[\mathbf{m}\left(\mathbf{x},t\right) - \mathbf{m}^{eq}\left(\mathbf{x},t\right)\right], \tag{6.10}$$

with $\mathbf{f}$ being the vector of the PPDFs. The moment coefficient and relaxation matrices $\underline{\mathbf{M}}$ and $\underline{\mathbf{K}}_{MRT}$ contain the different moments and the moment-specific collision frequencies. The vectors $\mathbf{m}^{eq}$ and $\mathbf{m}$ define the moments at equilibrium and non-equilibrium. Similar to the SRT method, the algorithm can be split into a collision and propagation step. For more information, especially for more details on the setup of the various matrices and vectors, the reader is referred to [8].

Boundary conditions

At inlets and outlets, von Neumann and/or Dirichlet conditions are frequently prescribed. In a simple case, the velocity is prescribed via a Dirichlet condition and the density is extrapolated from the next inner computational elements via a von Neumann condition. This creates a source and is commonly used for inlets. A simple outlet condition employs a Dirichlet condition for the density and a von Neumann condition for the velocity. In both cases, the equilibrium can be computed via Eq. 6.4. More complex boundary condition combinations [30] iteratively lower the pressure in the sense that the density is reduced, and extrapolate the velocity at the outlet. At the same time the momentum is extrapolated at the inlet via solving the equation of Saint-Vernant/Wanzel [47] in LB context. This yields a realistic inspiration behaviour if the former boundary condition is prescribed at the pharynx and the latter at the nostrils.

To realize no-slip conditions at tissue walls, interpolated bounce-back conditions [5], which perform a reflective collision of the wall-pointing PPDFs, are frequently used. They are of second-order accuracy and reflect the PPDFs weighted by the distance of the element center to the wall in direction i. There exist further popular schemes such as those from Ginzburg and D'Humières [14] and Yu et al. [57], which also deliver highly accurate results.

Large-Eddy Simulations

To perform LES computations, the governing equations are filtered in space and time and a modelling assumption for non-resolved scales is introduced. For simplicity, the following describes modelling via the Smagorinsky approach [19, 51]. In LB context, the viscosity ν in ω_x is replaced by $\nu_{LES} = \nu + \nu_t$, where

$$\nu_t = (C_s \delta x)^2 \sqrt{\underbrace{2\left(\frac{\omega_x}{2\rho c_s^2} \sum_i \left[f_i(\mathbf{x},t) - f_i^{eq}(\mathbf{x},t)\right] \xi_{i,a}\xi_{i,b}\right)^2}_{\text{squared filtered strain rate}}} \tag{6.11}$$

is the turbulent viscosity with mesh resolution δx and Smagorinsky constant C_s.

Mesh Refinement

If computations are to be employed on refined meshes, some modifications to the algorithm are necessary [9]. Missing incoming PPDFs need to be reconstructed from neighbours on a different level. To obey the laws of physics, it is required to keep the viscosity constant across mesh level interfaces. This goes a long with an adaption of the time step δt and yields transformations

$$f_{i,\lambda+1}(\mathbf{x},t) = \tilde{f}_i^{eq}(\mathbf{x},t) + \Theta_{\lambda+1} \cdot (\tilde{f}_{i,\lambda}(\mathbf{x},t) - \tilde{f}_i^{eq}(\mathbf{x},t)) \tag{6.12}$$

$$f_{i,\lambda}(\mathbf{x},t) = f_i^{eq}(\mathbf{x},t) + \Theta_{\lambda} \cdot \left(f_{i,\lambda+1}(\mathbf{x},t) - f_i^{eq}(\mathbf{x},t)\right) \tag{6.13}$$

for levels λ and $\lambda + 1$. The PPDFs with a tilde $<\tilde{}>$ are interpolated from the other mesh hierarchy. The transformations are functions of Θ_λ and $\Theta_{\lambda+1}$ given by

$$\Theta_\lambda = \frac{\delta x_\lambda}{\delta x_{\lambda+1}} \cdot \frac{\omega_{x,\lambda+1}}{\omega_{x,\lambda}} \tag{6.14}$$

$$\Theta_{\lambda+1} = \frac{\delta x_{\lambda+1}}{\delta x_\lambda} \cdot \frac{\omega_{x,\lambda}}{\omega_{x,\lambda+1}} \tag{6.15}$$

Performance of LB Methods

To analyze the parallel performance of LB methods, strong scaling experiments are performed for the SRT method on a cubic periodic domain with uniform mesh refinement, consisting of $1.1225 \cdot 10^9$ mesh elements [31, 32]. The experiments are run

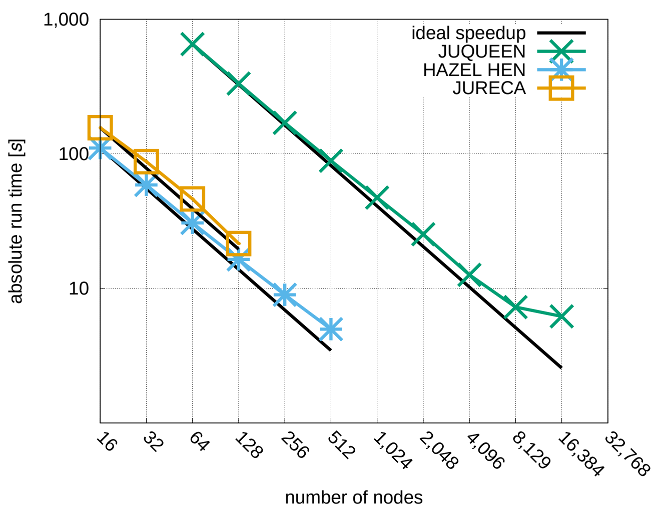


Figure 6.14: Results of strong scaling experiments on the three different HPC systems JURECA, JUQUEEN, and HAZEL HEN. The mesh is uniformly refined and consists of $1.1225 \cdot 10^9$ mesh elements. [32].

on the systems JURECA, JUQUEEN, and HAZEL HEN HPC systems located at JSC and HLRS. From Fig. 6.14 it is obvious that the LB method shows a very good strong scaling behaviour on all three systems, i.e., on JURECA an almost linear behaviour is visible up to 128 nodes, on HAZEL HEN the code scales well up to 512 nodes, and on the massively parallel system JUQUEEN a good scalability up to $8,192$ nodes with a slight decrease in parallel efficiency up to $16,384$ nodes is visible.

6.5.3 Summary and Conclusions on Using Lattice-Bolztmann Methods

Lattice-Boltzmann methods are a powerful alternative to conventional CFD methods such as finite volume, finite difference, finite element, and discontinuous Galerkin methods. Derived from the Boltzmann equation the single relaxation time and the multiple relaxation time methods are great options to simulate intricate flow in complex geometries. Especially their ability to implicitly compute the pressure and not to solve a Poisson equation in the incompressibility limit excels them. Furthermore, their high parallel efficiency and their straightforward implementation on hierarchical Cartesian meshes suit them well for large-scale simulations on HPC systems. Complex geometries can be represented by accurate boundary conditions and there exist modelling approaches for LES computations and refinement strategies. All these features, together with their great applicability for low Mach number and Reynolds number flows, make them an extraordinary method for the simulation of respiratory flows, especially when applied in combination with the meshing tools presented in Ch. 6.3.

6.6 Advanced Mesh Applications: High Resolution Simulations Without Any Turbulence Modelling

The computational simulation method in conjunction with the available computational resources determine a maximum resolution that can be achieved. Depending on this resolution, different approaches with increasing levels of modelling can be used to simulate flow.

A DNS resolves all scales of the flow and introduces no additional modelling. In contrast, an LES solves the spatially and temporally filtered governing equations and uses sub-grid scale (SGS) models such as the Smagorinsky [51] or dynamic Smagorinsky [13] models to reconstruct scales that are not resolved by the mesh, see Ch. 6.5.2. A computationally less expensive but also less accurate method than DNS and LES is to solve the RANS equations and to employ additional turbulence models such as the k-ϵ [25], k-ω [37], or the k-ω *SST* [15] models. These models introduce additional equations to solve for the turbulent kinetic energy k, the rate of dissipation ϵ, and the specific dissipation of the turbulence kinetic energy into internal thermal energy ω. The flow in the nasal cavity is at respiration at rest laminar and in some regions transitional but not necessarily turbulent [17, 18, 30]. RANS was developed for fully turbulent flow and the Reynolds averaging assumes decent fluctuations around a mean flow. Furthermore, model constants need to be individually tuned per case delivering inconsistencies in the solutions. The results obtained by this method hence allow for only a tentative evaluation of the flow in the nasal cavity.

It is obvious that DNS and even LES computations are out of reach for researchers without any access to HPC systems and that hence RANS simulations, e.g., with commercial software, remain their only option to simulate flow. Considering, however, the trend in hardware and parallel software development it is assumed that LES and DNS will become available sooner or later as desktop applications.

In the following, results for highly resolved simulations of the flow in the nasal cavity are presented that do not employ any kind of turbulence modelling, i.e., the equations of fluid mechanics are solved as is by means of an LB method on fine meshes that capture all important flow scales, see Ch. 6.3 and 6.5. The results are based on the findings in [28–30, 35, 55] and make use of the framework Zonal Flow Solver (ZFS) [31]. The interested reader is referred to these publications for more details.

6.6.1 Mesh Resolution Required for Fully-resolved Simulations

The mesh resolution is an important parameter for the accuracy of a simulation (see Ch. 6.4). It is of crucial interest to accurately determine various parameters that determine the quality of a nasal cavity, e.g., the pressure loss and the temperature increase as measures of the respiratory and heating capabilities. Furthermore, the wall-shear stress, which helps to identify regions of irritation, is of interest. Especially regions with high velocity gradients such as in the vicinity of the wall require a high resolution to accurately predict the aforementioned parameters.

To analyse the dependency of the mesh resolution on the simulation result, it is in most cases sufficient to perform grid convergence studies. This is done by solving the same case on successively refined meshes and to determine the change of the

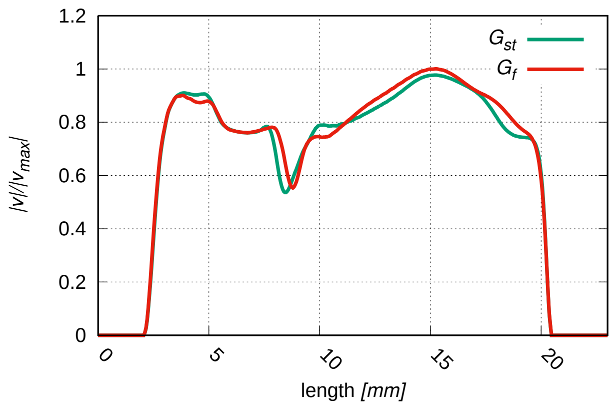


Figure 6.15: Grid convergence analysis. Two profiles at the pharynx for mesh resolutions of $\delta x_{st} = 93.569 \cdot 10^{-3} mm$ (G_{st} with $92.6 \cdot 10^6$ elements) and $\delta x_f = \delta x_{st}/2$ (G_f with $724 \cdot 10^6$ elements) are juxtaposed. The velocity magnitude $|v|$ is normalised by the overall velocity magnitude $|v_{max}|$ along the profile line [30].

solution with respect to the essential flow parameters. Figure 6.15 shows a comparison of the temporally averaged velocity profiles at the pharynx for two resolutions, i.e., for a case G_{st} with $92.6 \cdot 10^6$ elements and $\delta x_{st} = 93.569 \cdot 10^{-3} mm$, and a case G_f with $724 \cdot 10^6$ elements and $\delta x_f = \delta x_{st}/2$. Although some smaller differences are visible in the pharynx center region, especially in the vicinity of the walls, the curves of both profiles coincide. Considering furthermore the total pressure loss $\delta p_t = (p_s + p_d)_{pharynx} - (p_s + p_d)_{nostril}$, i.e., the difference of the sums of the static pressure p_s and the dynamic pressure $p_d = (\rho/2)|\mathbf{v}|^2$ at the nostrils and pharynx, the difference between G_f and G_{st} is only 2.1‰ and 0.8‰ for the left and right nasal cavity. The temperature difference $\delta T = T_{pharynx} - T_{nostril}$ can be evaluated similarly to the pressure loss as the difference in temperature between the nostrils and pharynx cross-sections. This yields a small difference of only $0.1985K$ between G_f and G_{st}. To summarise, the grid convergence analysis renders the resolution of G_{st} to be sufficient to simulate flow at respiration at rest.

Another method to evaluate a resolution is considering the power spectral density of the velocity components for various resolutions. Details on this method can be found in Lintermann et al. [30].

6.6.2 Examples of Simulation Results

In the following, several example results are presented that were generated by the highly scalable LB flow solver that operates on hierarchical Cartesian meshes (see Ch. 6.3 and 6.5). All simulations were run on supercomputers and the resolutions are in the sub-millimeter range to cover all essential flow phenomena. Three different cases N_1, N_2, and N_3 are considered that underwent an evaluation by medical experts beforehand. Figure 6.16 shows frontal CT-cross-sections of the different cases. Case

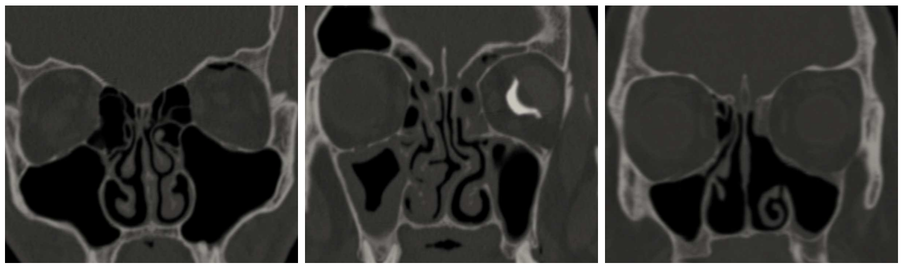
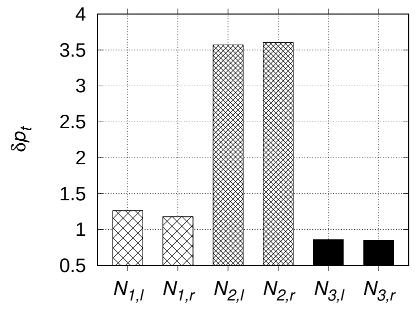
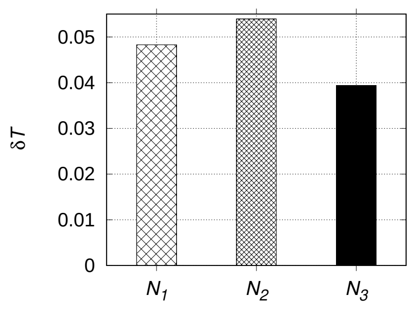


Figure 6.16: Frontal CT-cross-sections of the nasal cavities N_1, N_2, and N_3 from left to right [30].



(a) Non-dimensional total pressure loss across the nasal cavities.



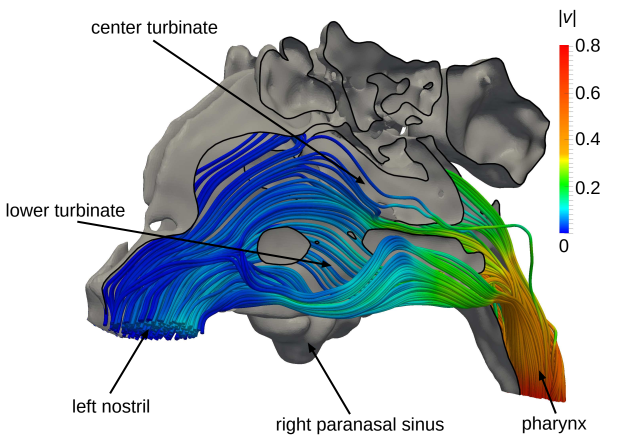
(b) Non-dimensional temperature increase across the nasal cavities.

Figure 6.17: Total pressure loss and temperature increase for three different cases N_1, N_2, and N_3 for the left (subscript l) and right (subscript r) cavities [30].

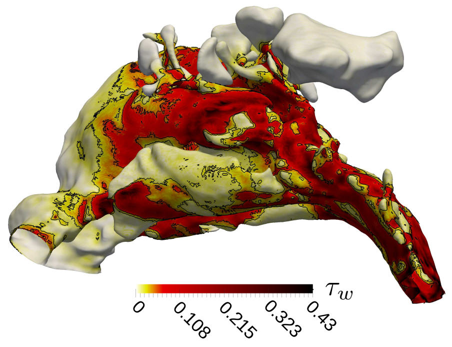
N_1 was considered a healthy nasal cavity with only slightly swollen turbinates on the left side. Case N_2 suffered from a bent septum, and swollen lower and center turbinates on the right side. Nasal cavity N_3 previously underwent a surgery in which the lower turbinate on the right side and the center turbinate on the left side were removed. Furthermore, a large orifice to the paranasal sinus on the left side existed. For further details, the interested reader is referred to the original publications [30, 35].

There are various results of a respiratory flow simulation that are of interest. As mentioned before, the difficulty to inhale can be quantified by considering the total pressure loss in the nasal cavity.

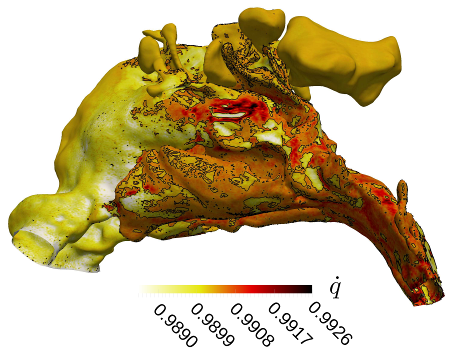
This is also a good parameter to juxtapose nasal cavities, i.e., to classify nasal cavities by their respiratory capabilities. Such a juxtaposition is shown in Fig. 6.17a for three nasal cavities N_1, N_2, and N_3. Obviously, case N_3 has the lowest pressure loss on both sides as the respiratory tract misses several turbinates. However, compared to N_1 the advantage is not excessive. In contrast, the swollen turbinates in N_2 lead to a high pressure loss and hence to impaired respiration. To furthermore find local phenomena responsible for the pressure loss, the total pressure along a stream line can be considered. The distribution of the mass flux in the nasal cavity can be analysed by



(a) Streamlines in the nasal cavity coloured by the non-dimensional velocity magnitude [33].



(b) Non-dimensional wall-shear stress mapped to the surface of a nasal cavity [30].



(c) Non-dimensional heat flux mapped to the surface of a nasal cavity [30].

Figure 6.18: Results of highly resolved simulations from Lintermann [30, 33].

measuring it in different cross sections or by visualising stream lines (see Fig. 6.18a). Another interesting parameter is the capability of a nasal cavity to heat the inhaled air. Figure 6.17b shows an evaluation for the same nasal cavities as in Fig. 6.17a. Undisturbed respiration in case N_3 comes with a diminished heating capability compared to cases N_1 and N_2, which almost heat up the air up to body temperature. To also analyse the impact of the flow on the tissue, i.e., to quantify the forces acting on the tissue, which might lead to irritations or even to inflammations, the wall-shear stress can be evaluated. Figure 6.18b shows a visualisation of the wall-shear stress mapped to the nasal cavity surface of case N_2. Obviously, the swollen turbinates and the bent septum lead to strong wall-shear stresses. Similar to the wall-shear stress, the heat flux can be evaluated by considering $\dot{q} = \kappa \partial T / \partial x_w$ with heat conduction coefficient κ. A result for case N_2 is shown in Fig. 6.18c. From both the wall-shear stress and the heat-flux it is clear that in narrowed geometries the less tempered inhaled flow passes close to the tissue and is thereby heated up quite well.

Further in-depth details on respiratory flows can be obtained by analysing vortical structures, e.g., by means of visualisation techniques (Δ-, Q-criterion, or λ_2 contours), the Reynolds stress tensor, the turbulent kinetic energy, root-mean square values, cross- and autocorrelations, eddy tunraround times, and power spectral densities. These are quantities that directly have an impact on the important parameters such as the pressure loss, the temperature increase, the wall-shear stress, and the heat flux but are not necessarily crucial to classify and evaluate nasal cavities in preparation of a surgery. Further details can be found in [28–30, 35, 55].

6.6.3 Recapitulation: Using Highly-Resolved Meshes

Simulations in the nasal cavity can be run with varying grades of detail with DNS delivering results with the highest accuracy and RANS with the lowest. Although DNS are quite expensive in terms of computational costs they deliver an extraordinary variety of data that is to both the fluid-mechanics expert and the medical doctor of interest. Concerning the physics of respiration, a doctor might be interested in phenomena causing certain functional degradations, e.g., local constrictions in the airways, which lead to strenuous respiration. The pressure loss and the wall-shear stress are good indicators to find such locations that are potential candidates for a surgery. The latter might also help to find locations with an increased risk of dry-out, irritation, or inflammation. The capability of the nasal cavity to increase the temperature contributes to the well-being of the patient and is hence another key component of the fluid mechanics of respiration. While the global increase along the airways delivers rather superficial information on the heating capability, an analysis of the local heat flux distribution allows to locate regions with low heat flux and hence regions that have a low impact on the temperature increase.

The expert in fluid mechanics is furthermore interested in the finest details of the flow, i.e., in the formation of recirculation zones, the energy distribution, frequency analyses of fluctuating flow, Reynolds stresses, and so forth, to advance fundamental research. Obviously, experts from the medical community, engineering, and HPC need to work together in an interdisciplinary context to extract the maximum of information from highly resolved simulations that are important to treat patients and to advance the fluid-mechanical understanding of respiration and computer science research.

6.7 Summary and Conclusion

In this chapter, an introduction to meshing complex shapes such as the human respiratory tract has been given. Different kinds of meshes, i.e., unstructured and structured meshes with different kinds of mesh elements have been considered and their advantages and disadvantages have been discussed. Using structured meshes, although they allow for an efficient computation and require only a small amount of memory, has shown not to be feasible for complex geometries. Despite their disadvantages from an efficiency point of view, unstructured meshes can be generated fully automatically and are applicable to intricate shapes. However, in this case one has to take care that the quality of the mesh elements is guaranteed. This can be realised by avoiding skewed cells or elements with a too high or too low aspect ratio, and by guaranteeing a sufficient resolution. The latter can be analysed with the help of mesh independence studies.

In addition to fundamentals in meshing, a parallel grid generator has been presented. It is capable of creating large scale hierarchical Cartesian meshes fully automatically on HPC systems in a short amount of time. The grid generator scales up to several hundred thousands of processes. Boundary and patch refinement methods enable high resolutions where necessary to capture of all key flow features. The parallel meshing algorithm is hence a reasonable tool to construct meshes for highly-resolved flow simulations, e.g., in the human airways. Other than that, there also exist other solutions to generate unstructured meshes. They are frequently part of commercial software packages or are free to use and can be used in combination with commercial solvers.

Furthermore, the lattice-Boltzmann method to simulate complex flows in intricate geometries has been presented. It shows high flexibility and high scalability on state-of-the-art supercomputers and is hence well suited to run fully-resolved and highly accurate simulations in the human respiratory tract. This method has been applied to different nasal cavities and a mesh independence study has been presented. It shows to be a suitable tool to evaluate nasal cavities from a fluid-mechanics point of view. Analyses of the pressure loss, the heating capability, the wall-shear stress, and the heat flux are in line with clinical findings.

References

[1] Bass, K., and Worth Longest, P. Recommendations for simulating microparticle deposition at conditions similar to the upper airways with two-equation turbulence models. *Journal of Aerosol Science 119* (2018), 31–50.

[2] Benzi, R., Succi, S., and Vergassola, M. The lattice Boltzmann equation: theory and applications. *Physics Reports 222*, 3 (dec 1992), 145–197.

[3] Bhatnagar, P. L., Gross, E. P., and Krook, M. A Model for Collision Processes in Gases. I. Small Amplitude Processes in Charged and Neutral One-Component Systems. *Phys. Rev. 94*, 3 (1954), 511–525.

[4] Bonet, J., and Peraire, J. An alternating digital tree (ADT) algorithm for 3D geometric searching and intersection problems. *International Journal for Numerical Methods in Engineering 31*, 1 (jan 1991), 1–17.

[5] Bouzidi, M., Firdaouss, M., and Lallemand, P. Momentum transfer of a Boltzmann-lattice fluid with boundaries. *Physics of Fluids 13*, 11 (2001), 3452–3459.

[6] Burstedde, C., Wilcox, L. C., and Ghattas, O. p4est : Scalable Algorithms for Parallel Adaptive Mesh Refinement on Forests of Octrees. *SIAM Journal on Scientific Computing 33*, 3 (jan 2011), 1103–1133.

[7] Cetin, M. O., Pauz, V., Meinke, M., and Schröder, W. Computational analysis of nozzle geometry variations for subsonic turbulent jets. *Computers & Fluids 136* (sep 2016), 467–484.

[8] D'Humières, D., Ginzburg, I., Krafczyk, M., Lallemand, P., and Luo, L.-S. Multiple-relaxation-time lattice Boltzmann models in three dimensions. *Philosophical transactions. Series A, Mathematical, physical, and engineering sciences 360*, 1792 (mar 2002), 437–51.

[9] Dupuis, A., and Chopard, B. Theory and applications of an alternative lattice Boltzmann grid refinement algorithm. *Physical Review E 67*, 6 (jun 2003), 1–7.

[10] Filipovic, N. *Chapter 8 - Computational modeling of dry-powder inhalers for pulmonary drug delivery*. Academic Press, 2020.

[11] FOLK, M., AND POURMAL, E. Balancing performance and preservation lessons learned with HDF5. In *Proceedings of the 2010 Roadmap for Digital Preservation Interoperability Framework Workshop on - US-DPIF '10* (2010), pp. 1–8.

[12] FRANK-ITO, D. O., WOFFORD, M., SCHROETER, J. D., AND KIMBELL, J. S. Influence of mesh density on airflow and particle deposition in sinonasal airway modeling. *J Aerosol Med Pulm Drug Deliv.* (2015).

[13] GERMANO, M., PIOMELLI, U., MOIN, P., AND CABOT, W. H. A dynamic subgrid-scale eddy viscosity model. *Physics of Fluids A: Fluid Dynamics 3*, 7 (jul 1991), 1760–1765.

[14] GINZBURG, I., AND D'HUMIÈRES, D. Multireflection boundary conditions for lattice Boltzmann models. *Physical Review E 68*, 6 (dec 2003), 066614.

[15] GROTJANS, H., AND MENTER, F. Wall functions for industrial applications, pages 1112–1117. JohnWiley & Sons, 1998. In *Computational Fluid Dynamics '98, ECCOMAS* (1998), K. Papailiou, Ed., John Wiley & Sons, pp. 1112–1117.

[16] GUO, Z., SHI, B., AND ZHENG, C. A coupled lattice BGK model for the Boussinesq equations. *International Journal for Numerical Methods in Fluids 39*, 4 (jun 2002), 325–342.

[17] HÖRSCHLER, I., MEINKE, M., AND SCHRÖDER, W. Numerical simulation of the flow field in a model of the nasal cavity. *Computers & Fluids 32*, 1 (jan 2003), 39–45.

[18] HÖRSCHLER, I., SCHRÖDER, W., AND MEINKE, M. On the assumption of steadiness of nasal cavity flow. *Journal of Biomechanics 43*, 6 (apr 2010), 1081–5.

[19] HOU, S., STERLING, J., CHEN, S., AND DOOLEN, G. D. A Lattice Boltzmann Subgrid Model for High Reynolds Number Flows. *Pattern formation and lattice gas automata 6* (jan 1994), 1–18.

[20] INTHAVONG, K., CHETTY, A., SHANG, Y., AND TU, J. Examining mesh independence for flow dynamics in the human nasal cavity. *Computers in Biology and Medicine 102* (2018), 40 – 50.

[21] ISAAC, T., BURSTEDDE, C., AND GHATTAS, O. Low-Cost Parallel Algorithms for 2:1 Octree Balance. In *2012 IEEE 26th International Parallel and Distributed Processing Symposium* (may 2012), IEEE, pp. 426–437.

[22] ISHIDA, T., TAKAHASHI, S., AND NAKAHASHI, K. Efficient and Robust Cartesian Mesh Generation for Building-Cube Method. *Journal of Computational Science and Technology 2*, 4 (2008), 435–446.

[23] KUWATA, Y., AND SUGA, K. Anomaly of the lattice Boltzmann methods in three-dimensional cylindrical flows. *Journal of Computational Physics 280* (jan 2015), 563–569.

[24] Lallemand, P., and Luo, L.-S. Theory of the lattice Boltzmann method: Dispersion, dissipation, isotropy, Galilean invariance, and stability. *Physical Review E 61*, 6 (jun 2000), 6546–6562.

[25] Launder, B. E., and Spalding, D. B. The numerical computation of turbulent flows. *Computer Methods in Applied Mechanics and Engineering 3*, 2 (mar 1974), 269–289.

[26] Li, J., Zingale, M., Liao, W.-k., Choudhary, A., Ross, R., Thakur, R., Gropp, W., Latham, R., Siegel, A., and Gallagher, B. Parallel netCDF: A High-Performance Scientific I/O Interface. In *Proceedings of the 2003 ACM/IEEE conference on Supercomputing - SC '03* (New York, New York, USA, 2003), ACM Press, p. 39.

[27] Lintermann, A. Efficient Parallel Geometry Distribution for the Simulation of Complex Flows. In *Proceedings of the VII ECCOMAS Congress 2016* (Athens, 2016), Technical University of Athens (NTUA) Greece, pp. 1277–1293.

[28] Lintermann, A., Eitel-Amor, G., Meinke, M., and Schröder, W. Lattice-Boltzmann Solutions with Local Grid Refinement for Nasal Cavity Flows. In *New Results in Numerical and Experimental Fluid Mechanics VIII*. Springer, 2013, pp. 583–590.

[29] Lintermann, A., Meinke, M., and Schröder, W. Investigations of the Inspiration and Heating Capability of the Human Nasal Cavity Based on a Lattice-Boltzmann Method. In *Proceedings of the ECCOMAS Thematic International Conference on Simulation and Modeling of Biological Flows (SIMBIO 2011)* (Brussels, Belgium, 2011).

[30] Lintermann, A., Meinke, M., and Schröder, W. Fluid mechanics based classification of the respiratory efficiency of several nasal cavities. *Computers in Biology and Medicine 43*, 11 (nov 2013), 1833–1852.

[31] Lintermann, A., Meinke, M., and Schröder, W. Zonal Flow Solver (ZFS): a highly efficient multi-physics simulation framework. *International Journal of Computational Fluid Dynamics* (mar 2020), 1–28.

[32] Lintermann, A., Pleiter, D., and Schröder, W. Performance of ODROID-MC1 for scientific flow problems. *Future Generation Computer Systems 95* (jun 2019), 149–162.

[33] Lintermann, A., Schlimpert, S., Grimmen, J., Günther, C., Meinke, M., and Schröder, W. Massively parallel grid generation on HPC systems. *Computer Methods in Applied Mechanics and Engineering 277* (may 2014), 131–153.

[34] Lintermann, A., and Schröder, W. Simulation of aerosol particle deposition in the upper human tracheobronchial tract. *European Journal of Mechanics - B/Fluids 63* (jan 2017), 73–89.

[35] LINTERMANN, A., AND SCHRÖDER, W. A Hierarchical Numerical Journey Through the Nasal Cavity: from Nose-Like Models to Real Anatomies. *Flow, Turbulence and Combustion 102*, 1 (jan 2019), 89–116.

[36] LOSEILLE, A., MENIER, V., AND ALAUZET, F. Parallel Generation of Large-size Adapted Meshes. *Procedia Engineering 124* (2015), 57–69.

[37] MENTER, F. Zonal two equation k-ω turbulence models for aerodynamic flows. In *24th AIAA Fluid Dynamics Conference* (Orlando, FL, USA, 1993), pp. AIAA paper 93–2906.

[38] NAKASHIMA, T., TSUBOKURA, M., VÁZQUEZ, M., OWEN, H. C., AND DOI, Y. Coupled Analysis of Unsteady Aerodynamics and Vehicle Motion of a Heavy-Duty Truck in Wind Gusts. *Computers & Fluids 80* (2012), 1–9.

[39] NISHIKAWA, H., AND DISKIN, B. Development and Application of Parallel Agglomerated Multigrid Methods for Complex Geometries. In *20th AIAA Computational Fluid Dynaimcs Conference* (Honolulu, Hawaii, 2011), pp. 27–30.

[40] PERIC, M. Flow simulation using control volumes of arbitrary polyhedral shape. *ERCOFTAC Bulletin, No. 62* (2004).

[41] POGORELOV, A., MEINKE, M., AND SCHRÖDER, W. Large-Eddy Simulation of the Unsteady Full 3D Rim Seal Flow in a One-Stage Axial-Flow Turbine. *Flow, Turbulence and Combustion* (jul 2018).

[42] POGORELOV, A., SCHNEIDERS, L., MEINKE, M., AND SCHRÖDER, W. An Adaptive Cartesian Mesh Based Method to Simulate Turbulent Flows of Multiple Rotating Surfaces. *Flow, Turbulence and Combustion 100*, 1 (jan 2018), 19–38.

[43] QIAN, Y. H., D'HUMIÈRES, D., AND LALLEMAND, P. Lattice BGK Models for Navier-Stokes Equation. *Europhysics Letters (EPL) 17*, 6 (feb 1992), 479–484.

[44] RICHARDSON, L. On the approximate arithmetical solution by finite differences of physical problems involving differential equations, with an application to the stresses in a masonry dam. *Proceedings of the Royal Society of London. Series A 83* (1910), 335–336.

[45] ROACHE, P. Perspective: A method for uniform reporting of grid refinement studies. *Journal of Fluids Engineering 116* (1994), 405–413.

[46] SAGAN, H. *Space-Filling Curves*, 1 ed. Universitext. Springer New York, New York, NY, 1994.

[47] SAINT-VERNANT, B., AND WANTZEL, L. Mémoire et expérience sur l'écoulement déterminé par des différences de pressions considérables. *Journal de l'École Polytechnique H.27* (1839), 85ff.

[48] SCHNEIDERS, L., GRIMMEN, J. H., MEINKE, M., AND SCHRÖDER, W. An efficient numerical method for fully-resolved particle simulations on high-performance computers. In *Proceedings in Applied Mathematics and Mechanics* (Lecce, Italy, 2015), GAMM, Ed., Wiley-VCH.

[49] SCHNEIDERS, L., GÜNTHER, C., MEINKE, M., AND SCHRÖDER, W. An efficient conservative cut-cell method for rigid bodies interacting with viscous compressible flows. *Journal of Computational Physics 311* (apr 2016), 62–86.

[50] SCHNEIDERS, L., MEINKE, M., AND SCHRÖDER, W. Direct particle–fluid simulation of Kolmogorov-length-scale size particles in decaying isotropic turbulence. *Journal of Fluid Mechanics 819* (may 2017), 188–227.

[51] SMAGORINSKY, J. General Circulation Experiments with the Primitive Equations. *Monthly Weather Review 91*, 3 (mar 1963), 99–164.

[52] THOMPSON, J. F., SONI, B. K., AND WEATHERILL, N. P. *Handbook of grid generation*. CRC Press, Taylor & Francis Inc., 1998.

[53] THOMPSON, J. F., WARSI, Z., AND MASTIN, C. W. *Numerical grid generation : foundations and applications*. Elsevier Science Pub. Co., New York, North-Holland, 1985.

[54] VINCHURKAR, S., AND LONGEST, P. W. Evaluation of hexahedral, prismatic and hybrid mesh styles for simulating respiratory aerosol dynamics. *Computers & Fluids 37* (2008), 317–331.

[55] WALDMANN, M., LINTERMANN, A., CHOI, Y. J., AND SCHRÖDER, W. Analysis of the Effects of MARME Treatment on Respiratory Flow Using the Lattice-Boltzmann Method. *New Results in Numerical and Experimental Fluid Mechanics XII* (2020), 853–863.

[56] WHITE, A. T., AND CHONG, C. K. Rotational invariance in the three-dimensional lattice Boltzmann method is dependent on the choice of lattice. *Journal of Computational Physics 230*, 16 (jul 2011), 6367–6378.

[57] YU, D., MEI, R., LUO, L.-S., AND SHYY, W. Viscous flow computations with the method of lattice Boltzmann equation. *Progress in Aerospace Sciences 39*, 5 (jul 2003), 329–367.