AC 2012-4538: FLUID DYNAMICS SIMULATION USING CELLULAR AUTOMATA

Dr. Gunter Bischof, Joanneum University of Applied Sciences, Graz, Austria

Throughout his career, Gnter Bischof has combined his interest in science, engineering and education. He studied physics at the University of Vienna, Austria, and acquired industry experience as development engineer at Siemens Corporation. Currently, he teaches Engineering Mathematics and Fluid Mechanics at Joanneum University of Applied Sciences. His research interests focus on vehicle aerodynamics, materials physics, and engineering education.

Mr. Christian Steinmann, Joanneum University of Applied Sciences, Graz, Austria

Christian Steinmann has an engineer degree in mathematics from the Technical University Graz, where he focused on software quality and software development process assessment and improvement. He is Manager of HM&S IT-Consulting in Graz and provides services for SPiCE/ISO 15504 and CMMI for development as a SEI-certified instructor. He performed more than 100 process assessments in software development departments for different companies in the finance, insurance, research, automotive, and automation sector. Currently, his main occupation is a consulting project for process improvement at the Electrics/Electronics Development Department at Volkswagen in Wolfsburg, Germany. On Fridays, he is teaching computer science introductory and programming courses at Joanneum University of Applied Sciences in Graz, Austria and has for 14 years.

Fluid Dynamics Simulation using Cellular Automata

The idea to apply project-based learning as a didactical method in the freshman year was primarily driven by the need to motivate the students to apply theoretical knowledge in practice as early as possible. Faculty teaching in the areas of mathematics, science and information technology noted that students were not always enthusiastic in approaching the theoretical concepts involved in these disciplines, and that they frequently failed to recognize the interrelatedness of what they were studying as well as its applicability to their future professions.

In the last academic year a particularly challenging problem was posed on those first-year students, who have, according to their interests, chosen the Lattice-Gas Cellular Automaton project. Their task was the implementation of the FHP lattice-gas model in a computer program for the visualisation of two-dimensional fluid density and velocity distributions. In this model, particles can move with the same velocity at each site of the residing triangular lattice in any of six possible directions. An exclusion principle puts a stringent constraint on these velocities. Collision occurs synchronously at the lattice nodes, while the streaming takes place on the connection between each two nodes. The collision rules are chosen in such a way that mass and momentum are conserved. By multi-scale analysis it can be shown that the FHP model asymptotically goes over to the incompressible Navier-Stokes equation.

The students worked in teams of three, and four groups were assigned the same task in order to introduce a competitive aspect, which increased the students' motivation. The members of faculty who have proposed the project supervised and supported the students. The progress of the work was continuously evaluated in order to ensure a successful outcome of the projects.

In this paper the project task, the essentials of the lattice-gas model, the students' learning process and its assessment will be discussed, and the outcome of the project will be presented.

Introduction

At the Joanneum University of Applied Sciences, we have started to offer a four-year automotive engineering undergraduate degree program in 1996. The main aim of the program is to help students acquire professional and scientific qualifications both on a theoretical and a hands-on level. On that account faculty considers especially important to apply modern didactical methods in the degree program as early as possible to increase the efficiency of knowledge transfer and to fortify the students' motivation to learn and to co-operate actively.

A three-phase multi subject didactical method, based on the well-known methodology of project based learning (PBL), has been introduced about 12 years ago. It has proved to be an excellent method to demonstrate the need of basic sciences in professional engineering. Students are confronted, in addition to their regular coursework, with problems that are of a multidisciplinary nature and demand a certain degree of mathematical proficiency. A particularly suitable way of doing so turned out to be the establishment of interdisciplinary project work in the courses Information Systems and Programming, which forms the first phase of this didactical method in the second and third semester of the degree program. The results presented in this paper arose from this initial phase of our didactical approach. The second phase, which takes place in the junior year, and the third phase that comprises of an internship in industry are described elsewhere[1].

The courses Information Systems and Programming take place in the first three semesters. In the first semester the students are given an introduction into standard application software. In the second semester the programming language Visual Basic (VB) is introduced, which enables the students to develop graphical user interfaces (GUI) with comparatively little effort.

The students are offered a variety of project proposals at the beginning of the semester. They can choose their project according to their interests and skills. In general, two or more groups of three or four members work simultaneously on the same task. In this way competition is generated, which in turn increases the students' motivation. By having the option to select their own project, the students have the chance to delve into subjects of particular interest to them, but which are not taught in such depth and detail in regular lectures. One student is designated by the team as project leader and assumes the competences and responsibilities for this position. This structure promotes the development of certain generic skills, like the ability to work in teams, to keep records and to meet deadlines. The lecturers who propose a topic supervise and support the project groups by offering additional meetings and lectures. The projects' demands and the work load are continuously evaluated in order to avoid overburdening the students.

The project introduced in this paper was offered first-year students in their second semester, with the aim to demonstrate to them a typical application of computational methods in engineering and to stimulate their motivation and basic interest in informatics and mathematics. Although fluid mechanics is not part of the curriculum in the first year of study, automotive engineering freshmen naturally show a strong interest in this topic. Concepts like aerodynamic drag, uplift and downforce are often used in connection with vehicle design, and the visual perception of the flow around an airfoil or an automobile fosters the students' comprehension of fluid dynamics. Visualization bridges the quantitative information of fluid dynamics computation and the human intuition, which is necessary for a true understanding of the phenomena in question. The programming tool used in this project (VB) is perfectly suited for this purpose; it allows the students to design a GUI and to visualize the result of a computation quickly and easily.

The solution of the governing equations of fluid dynamics, as a matter of course, is way too advanced for first-year students. The cellular automata approach, however, renders it feasible even for freshmen to compute and visualize the two-dimensional flow of incompressible fluids through pipes, nozzles and around obstacles. And in contrast to the simple use of commercial tools for the demonstration of fluid flow, the derivation of flow fields from a self-made computer program based on comprehensible microdynamics is quite satisfying for the students, and leads to a much deeper insight into the subject matter.

The dynamics of incompressible fluids

The fundamental equations of fluid dynamics are based on the universal laws of conservation of mass, momentum and energy. The conservation of mass leads to the continuity equation, and the conservation of momentum, which is nothing more than Newton's Second Law, yields the so called Navier-Stokes equation. The conservation of energy is more or less identical to the First Law of Thermodynamics. A simplification of the resulting flow equations is obtained when considering an incompressible flow of a Newtonian fluid. Liquids are often regarded as incompressible because they require such high pressure to compress

them appreciably. However, it is quite legitimate in many applications to consider even a gaseous medium such as the atmosphere to be incompressible, in which case the incompressible flow assumption typically holds well at low Mach numbers up to about 0.3. The behavior of a viscous incompressible fluid is governed by the simplified Navier-Stokes equation, which can be written as

$$\frac{\partial \mathbf{v}}{\partial t} + (\mathbf{v} \cdot \nabla)\,\mathbf{v} = -\frac{1}{\rho}\nabla P + \nu\,\Delta \mathbf{v},$$

and by the continuity equation (under the incompressible assumption):

$$\nabla \cdot \mathbf{v} = 0,$$

where ∇ is the nabla operator, $\mathbf{v}$ is the flow velocity, P the pressure, ρ the constant mass density, Δ the Laplacian and ν the kinematic viscosity. The Navier-Stokes equation is a coupled system of nonlinear partial differential equations which prohibits its analytical solution except for a few cases. Numerical methods are required to simulate the time evolution of flows[2].

Flows with small velocities are usually smooth and are called laminar. At high velocities they tend to become turbulent. Nevertheless, the transition from laminar to turbulent flows does not only depend on the free stream velocity V; it also depends on the characteristic length L of an obstacle and on the kinematic viscosity ν. From these parameters one can form essentially one dimensionless number, namely the Reynolds number

$$\mathrm{Re} = \frac{V \cdot L}{\nu}.$$

The parameters V and L can be used to scale all quantities in the Navier-Stokes equation in such a way that it does not contain any scale and only one dimensionless quantity, namely the Reynolds number. Thus all flows of the same type but with different values of V, L and ν are described by one and the same non-dimensional solution if their Reynolds numbers are equal. This dynamic similarity provides the link between flows in the real world where length is measured in meters and the simulation of these flows with cellular automata over a lattice with unit grid length and unit lattice speed. In these models the viscosity is a dimensionless quantity. These dimensionless flows on the lattice are similar to real flows when their Reynolds numbers are equal[3].

The fact that different microscopic interactions can lead to the same form of macroscopic equations was the starting point for the development of cellular automata for the simulation of fluid flow, the so called lattice-gas cellular automata. These models are different from models such as finite difference, finite volume, and finite element which are based on the discretization of partial differential equations (top-down models). They render it possible to consider artificial micro-worlds of particles located on lattices with interactions that conserve mass and momentum. From them then the partial differential equations can be derived by multi-scale analysis (bottom-up models).

Cellular Automata

"Cellular automata are sufficient simple to allow detailed mathematical analysis, yet sufficient complex to exhibit a wide variety of complicated phenomena." Stephen Wolfram[4]

Cellular automata were originally introduced by John von Neumann and Stanislaw Ulam as an idealization of biological systems, with the particular purpose of modeling biological self-

replication[5,6]. One of the best-known cellular automata is Conway's *"Game Of Life"*, discovered by John Horton Conway[7] in 1970 and popularized in Martin Gardner's Scientific American columns[8].

Cellular automata can be regarded as mathematical idealizations of physical systems with discrete space and time and in which physical quantities take on a finite number of discrete values, such as "*on*" and "*off*". A cellular automaton usually consists of a regular uniform lattice with a discrete variable at each site. These variables completely specify its state. Cellular automata evolve in discrete time steps, with the value of the variable at one site being affected by the values of the variables in its neighborhood, which is typically taken to be the site itself and all immediately adjacent lattice sites. The variables at each site are updated simultaneously, based on the values of the variables in their neighborhood at the preceding time step, and according to a finite set of local rules[4]. In a nutshell, cellular automata can be characterized as follows:

- Cellular automata are regular arrangements of single sites of the same kind
- Each site holds a finite number of discrete states
- The states are updated simultaneously at discrete time steps
- The update rules are deterministic and uniform in space and time
- The rules for the evolution of a cell depend only on the local neighborhood of a site

In 2002 Stephen Wolfram published *"A New Kind of Science"* [9], in which he extensively argues that the discoveries about cellular automata are not isolated facts but are robust and have significance for all disciplines of science.

Lattice-Gas Cellular Automata

The first lattice-gas cellular automaton was proposed in 1973 by Hardy, de Pazzis and Pomeau (HPP model)[10]. Their model is based on a two-dimensional square lattice and is of interest today mainly for historical reasons. The HPP model does not obey the desired hydrodynamic equations (Navier-Stokes) in the macroscopic limit, which is due to the insufficient degree of rotational symmetry of the lattice.
In 1986 Frisch, Hasslacher and Pomeau[11] showed that a lattice-gas cellular automata model over a lattice with a larger symmetry group than for the square lattice yields the incompressible Navier-Stokes equation in the macroscopic limit (FHP model).

The essential properties of the FHP model are[3]:

- The underlying regular lattice shows hexagonal symmetry (see Figure 1).
- The nodes (sites) are linked to six nearest neighbors located all at the same distance with respect to the central node.
- The vectors $\mathbf{c}_i$ linking nearest neighbor nodes are called lattice vectors or lattice velocities

$$\mathbf{c}_i = \left(\cos\left(\frac{\pi}{3} i\right), \sin\left(\frac{\pi}{3} i\right) \right), \quad i = 1, \ldots, 6.$$

  with $|\mathbf{c}_i| = 1$ for all i (see Figure 1). More precisely, the lattice velocities are given by the lattice vectors divided by the time step Δt which is always set equal to 1.
- A cell is associated with each link at all nodes.
- Cells can be empty or occupied by at most one particle. This exclusion principle is characteristic for all lattice-gas cellular automata.

- All particles have the same mass, which is set equal to 1 for simplicity, and are indistinguishable.
- The evolution in time proceeds by an alternation of collision *C* and streaming *S* (also called propagation). So a full time step reads: in-state ⇒ collision *C* ⇒ out-state ⇒ streaming *S*. Collision occurs at the nodes, while streaming takes place on the connection between each two sites. The collision should conserve mass and momentum while changing the occupation of the cells.
- The collisions are strictly local, i.e. only particles of a single cell are involved.

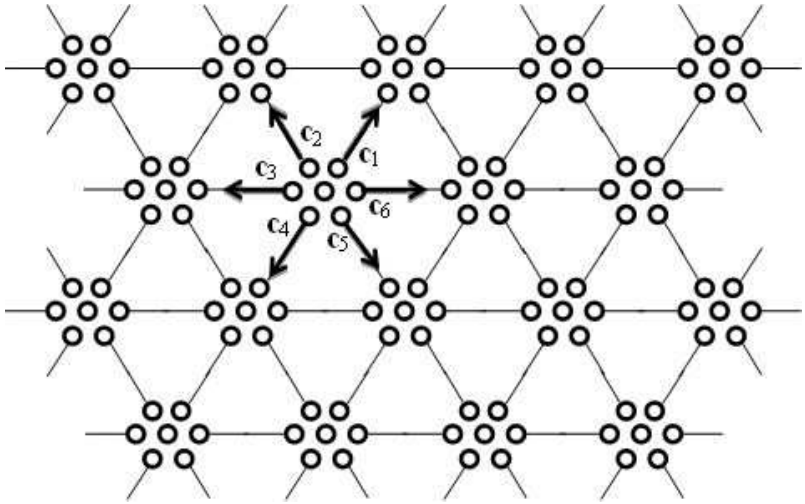

Figure 1: The triangular lattice of the FHP model shows hexagonal symmetry. The lattice velocities $\mathbf{c}_i$ are represented by arrows. The circles at each node represent the seven channels corresponding to particles moving along the six directions of the triangular lattice and to the rest particle (center) in FHP-II.

Several versions of the FHP model have been developed with the same geometrical structure but with different collision rules. The FHP-I model is a 6-bit model. There are $2^6 = 64$ different states for a node. Each node has six channels, corresponding to the six directions of the triangular lattice $\mathbf{c}_i$. The masses of all the particles are equal and usually taken as unity. Thus, the momentum $\mathbf{p}_i$ of each particle is equal to $\mathbf{c}_i$ and the kinetic energy is equal to ½. The evolution rule of the FHP-I model is a two-phase sequence: a propagation phase and a collision phase. During the propagation phase, a particle present at node $\mathbf{r}$ in channel i moves to node $\mathbf{r} + \mathbf{c}_i$, its nearest neighbor in the direction i. During the collision phase, pairs of particles arriving at the same node from opposite directions undergo a binary collision with an output state rotated by + 60° or – 60° with probabilities p and $1 – p$ respectively. Thus, some stochasticity enters the FHP microdynamics. Most commonly p is chosen to be 0.5 (Figure 2a). The two-particle collisions conserve not only mass and momentum but also the difference of the number of particles that stream in opposite directions. This "spurious invariant" is undesirable because it restricts to a certain degree the dynamics of the model and has no counterpart in the real world. It can be destroyed by deterministic triple collisions which conserve mass and momentum: three particles coming from three directions forming 120° angles between each other will be deflected by 60° (Figure 2b). The states in Figure 2 are the only ones among the 2^6 possible states that can undergo effective collision; all the other states remain unchanged. As a consequence, the collisional efficiency of FHP-I is therefore only 7.8 %.

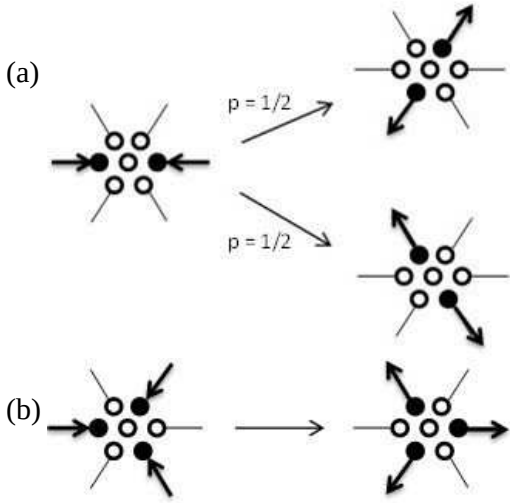

Figure 2: Collision rules for the FHP-I model, reduced by symmetry. Filled circles denote occupied cells and open circles empty cells. In-states are shown on the left hand side, out-states on the right hand side.

The FHP-II model is a variant of the FHP-I model that includes the possibility of one rest particle per node, in addition to the six moving particles of FHP-I. Each node than has seven channels, corresponding to particles moving along the six directions of the triangular lattice and to the rest particle. The channels associated with moving particles are labeled by integers from 1 to 6, and the channel corresponding to the rest particle is labeled 0. The propagation phase for moving particles is the same as for FHP-I since it does not affect the rest particle. The collision rules are similar to FHP-I with additional collisions coupling moving and rest particles (Figure 3).

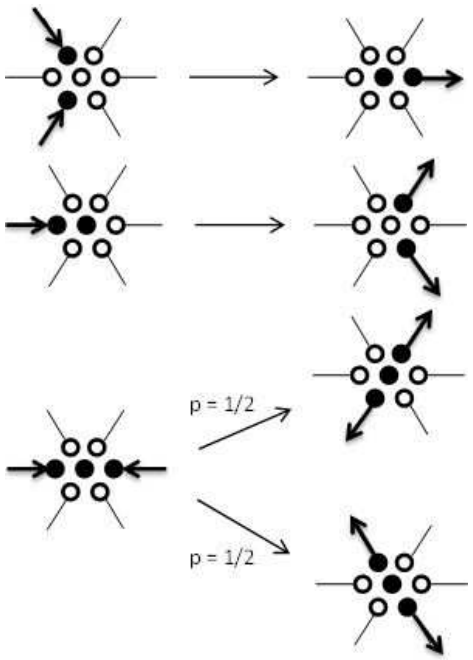

Figure 3: Additional collision rules for the FHP-II model, reduced by symmetry. The filled and open circles represent the bit-pattern of the node's state.

FHP-I and FHP-II have the same microscopic properties; the essential difference lies in the number of possible effective collisions which is larger for FHP-II, and yields a collisional efficiency of 17.2 %. This leads to a smaller kinematic viscosity for FHP-II than for FHP-I, which allows simulations at a higher Reynolds number.[12]

A further variant of the 7-bit FHP-II model is FHP-III where the collision rules are designed to include as many collisions as possible, under the constraint of having the same collisional invariants as with FHP-II. Figure 4 shows the additional collision rules of FHP-III. As in figures 2 and 3, the lattice-preserving isometries are used to reduce the number of collisions shown. By rotating, flipping, and combining the types of collisions shown in Figures 2 to 4, a full set of 128 collision rules (2^7) is derived. It can be shown that only 76 among the 128 possible states can undergo effective collision, which results in a collisional efficiency of 59.4 % [12].

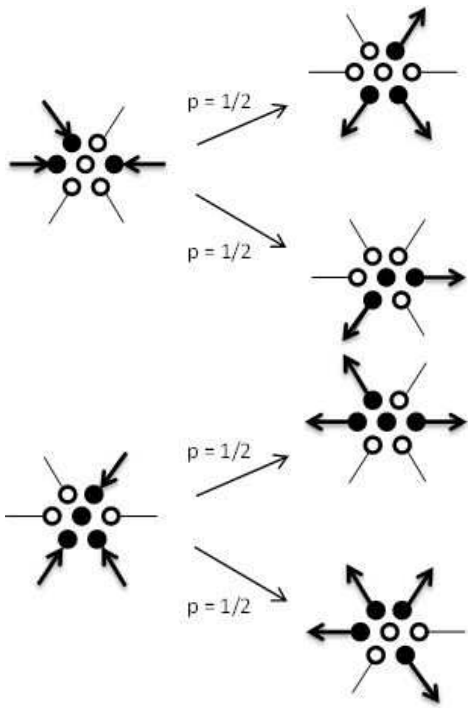

Figure 4: Additional collision rules for the FHP-III model, reduced by symmetry. The filled and open circles represent the bit-pattern of the node's state.

The corresponding macroscopic equations of the FHP models I, II and III all have the same form and differ only in their viscosity coefficients. As a rule of thumb the viscosity coefficient decreases with increasing number of collisions.

The FHP model was the first successful lattice-gas cellular automaton. Starting from the Boolean microdynamics the macroscopic equations can be derived up to first order (Euler equation) by a multi-scale expansion (Chapman-Enskog expansion). The second order expansion yields the Navier-Stokes equation[3].

Implementation of the Lattice-Gas Cellular Automaton

At the beginning of their second semester of study, students were introduced to the offered projects by one page task descriptions, detailed information on the scope of the projects, the deliverables, the timetable and deadlines and the evaluation criteria. Sufficient time was allotted for project groups to form and make their selections. This procedure was chosen because it generates usually a high degree of negotiation within the group as they vie for projects which interest them most. Experience has shown that the process of claiming ownership of a task as opposed to having it prescribed contributes greatly to the success of the projects[1].

Four three-member teams of students started to tackle the Lattice-Gas Cellular Automata project. After initial consultation with the supervisors and the assignment of duties within the team, students embarked on the main part of the project work. They typically went through the following stages:

- Acquiring relevant background knowledge and skills
- Finding technical solutions
- Designing and programming software
- Documenting the process from research to development and finally to output

The activity reports of the students, which are part of the project documentation, give an overview of the development process of such a project (see Table 1).

Table 1: Activity report of a team member

Date	Activity	Result	Hours
30.03.2011	Kick-off meeting	Explanation of the project and distribution of the tasks.	3
31.03.2011	Development of “RandDot” test routine	Comprehension of the task; performance test	2.5
11.04.2011	Team meeting, clarification of tasks	Implementation of a prototype	1
12.04.2011	Development of FHP1_simple	Simple grid, reflection boundary conditions, 2 byte cell variable	3
13.04.2011	Development of FHP1_simple	Real-time representation of particles	4
14.04.2011	2nd meeting with Dr. Bischof	Lattice structure, FHP model, look-up table	1.5
18.04.2011	Development of FHP1_evolution	Adapted grid, 1 byte cell variable, usage of look-up table	2.5
20.04.2011	Development of FHP1_evolution		3
26.04.2011	Development of FHP1_AVG	Coarse graining for noise suppression	3.5
27.04.2011	Development of FHP1_AVG		2
02.05.2011	Development of FHP2_AVG	Implementation of collision rules of FHP-II model	1.5
06.05.2011	Development of FHP3_AVG	Implementation of collision rules of FHP-III model	2.5
10.05.2011	Development of FHP3_AVG	Implementation of collision rules of FHP-III model	4
16.05.2011	Development of FHP3_AVG_Arrow	Visualization of particle velocities via arrows	1
18.05.2011	Development of FHP3_AVG_Arrow	Visualization of particle velocities via arrows	2.5
19.05.2011	Development of FHP3_AVG_Arrow	Visualization of particle velocities via arrows	1.5
30.05.2011	Modeling of compatible data types	Preparation of the code for linking with GUI	2
31.05.2011	Logics swapped in separate module	Boolean algebra separated from main code	3.5
01.06.2011	Data type and logics module integrated	Data and Boolean algebra linked with GUI	1.5
02.06.2011	Data type and logics module integrated	Data and Boolean algebra linked with GUI	1.5
06.06.2011	Team meeting	Discussion of GUI and paint function	2
08.06.2011	Bug fixing	Remedy problems with initialization of data pool	2
19.06.2011	Bug fixing	Remedy problems with start/stop of simulation	1
20.06.2011	Configuration “on the fly”	Enable change of parameters during simulation	2
21.06.2011	Extended functionality “initial pressure”	Initialization with rest particles enabled	1.5
25.06.2011	Integrate paint function	Translate GUI paint function results into data pool	2
26.06.2011	Integrate paint function	Translate GUI paint function results into data pool	1
28.06.2011	Link FHP3 with GUI	Transfer user settings from GUI to FHP3	2
29.06.2011	Final team meeting, documentation	Finishing of program and update of documentation	6

The role of the lecturers was to guide the students through these stages and give them the tools necessary for finding the missing pieces of what for them was often a great puzzle, at least in the early stages. In addition, two progress reports were required from each team, the first one after five weeks and the second one another four weeks later. Both reports were evaluated by the supervisors and a detailed feedback was given. The assessment of the project statuses used a traffic-light color scheme to categorize the risk of the failure of a project. Collaboration between the groups was not allowed during the entire process. Both the progress reports and the dissimilar outcome of the projects gave a clear indication that the four groups worked independently.

The biggest hurdle for all project teams turned out to be that in cellular automata solely information is handed over from node to node and no physical movement of particles takes place. This can be best realized with multi-spin coding since the exclusion principle makes it possible to describe the state of the cells of the lattice-gas cellular automata by the Boolean arrays $n_i(\mathbf{r},t)$: $n_i(\mathbf{r},t) = 1$ if the cell i is occupied, and 0 if the cell i is empty. $\mathbf{r}$ and t indicate the discrete points in space and time. In the FHP III model a seven-bit variable is enough to carry all the information at one site.

The students needed to create the array $M(j)$ that has the length of the total number of sites with each entry being that seven-bit variable, and j being the site number corresponding to the position $\mathbf{r}$. The compilation of the matrix was a bit aggravated by the fact that, due to the hexagonal grid, odd rows have one more site than even rows.

Another difficulty the students had to master was the definition of proper boundary conditions. Dirichlet, freestream and reflection boundary conditions were implemented. A bounce back scheme was used for no-slip conditions, while reflections led to a slip boundary. The Dirichlet boundary conditions were set as random variables on the boundary, with a probability distribution indicating the values at the boundary.

The routine that processes the collision phase refers to a collision look-up table, which is created in the initialization part of the program, finds out the out-state corresponding to the in-state for each site, and uses the one indicated by a random variable. For the generation of the random numbers a pseudo-random choice was used where the rotational sense in the out-states of the collisions changes by chance for the whole domain from time step to time step. Therefore, only one random number had to be generated per time step, which reduced the computation time tremendously. On the other hand, in this way the randomness of the collisions is somewhat reduced, which necessitates ensemble averaging for a proper interpretation of the simulation. Anyway, since noise is one of the biggest problems of the FHP model, ideally both space averaging and ensemble averaging should be used. The space average is simply achieved by averaging over a user defined number of neighboring nodes.

In the following, some results of one of the student projects are presented. In this context, it should be mentioned that while the minimum requirements were defined, in general no limits were placed on the students' creativity nor on the amount of time they should invest in order to complete the projects.

The best graded project team implemented a very user-friendly GUI with several tools for the intuitive usage of the program, such as sliders for the variation of coarse graining (spatial averaging), fluid pressure, wind speed and wind direction. For the illustration of the flow around objects some predefined obstacles can be chosen from the GUI. With the help of a

paint function implemented by the students the users can manually create obstacles of their choice. Flow direction and boundary conditions can also be selected in the GUI.

As to be expected, the soon-to-be automotive engineers provided a vehicle's contour as predefined obstacle (see Figures 5 and 6).

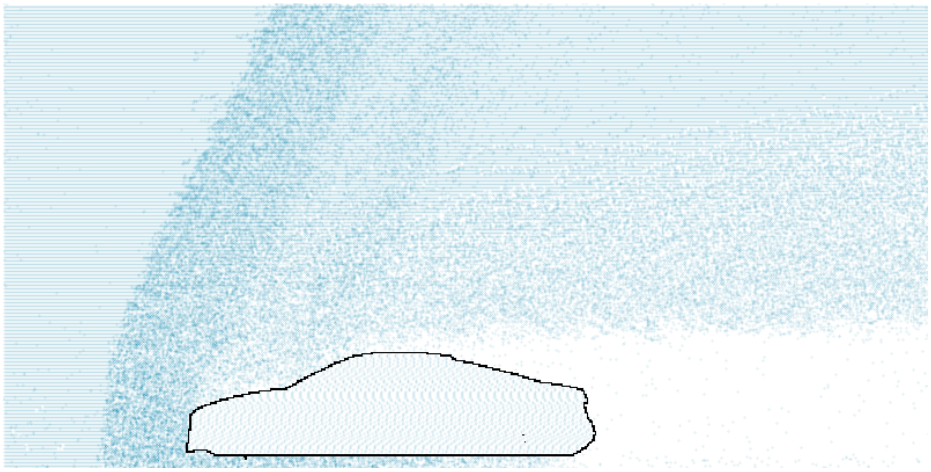

Figure 5: Uniform flow past a vehicle's contour. Mass density representation.[13]

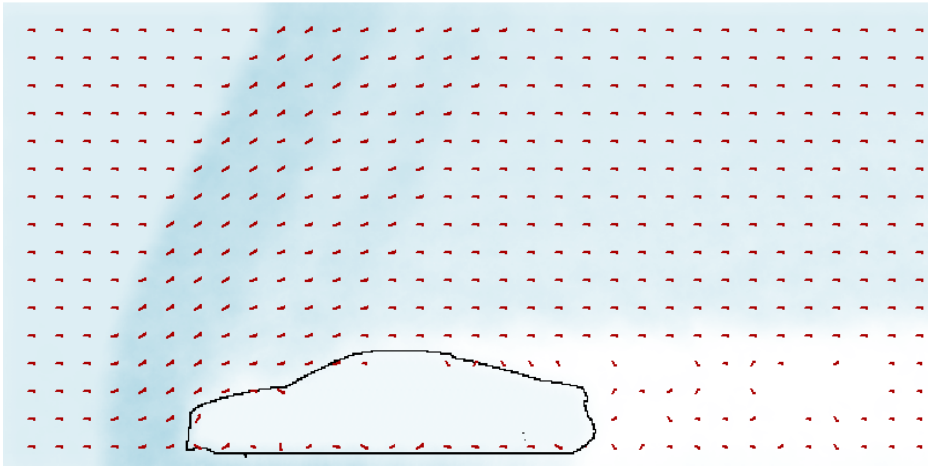

Figure 6: Uniform flow past a vehicle's contour. The mass density representation is superimposed by arrows illustrating the velocity field.[13]

In Figure 5 the mass density distribution of the flow is illustrated, and in Figure 6 a velocity direction field is superimposed. Both mass density and momentum density can be obtained from the mean occupation numbers N_i of the lattice-gas cellular automaton, which are calculated by averaging over neighboring nodes:

$$N_i(\mathbf{r},t) = \langle n_i(\mathbf{r},t) \rangle .$$

These mean occupation numbers are used to define the mass

$$\rho(\mathbf{r},t) = \sum_i N_i(\mathbf{r},t)$$

and momentum density

$$\mathbf{j}(\mathbf{r},t) = \sum_i N_i(\mathbf{r},t)\mathbf{c}_i .$$

These quantities are defined with respect to nodes and not to cells or area. Once the momentum density array (velocity field) is derived, other quantities of interest (such as flow rate or drag force) can be found.

Other predefined objects of study were four-digit NACA airfoils[14]. These early airfoil series were generated using analytical equations that describe the curvature (camber) of the geometric centerline of the airfoil section as well as the section's thickness distribution along the length of the airfoil. The first digit specifies the maximum camber in percentage of airfoil length, the second indicates the position of the maximum camber in tenths of chord, and the last two numbers provide the maximum thickness of the airfoil in percentage of the chord. Different airfoil geometries can be chosen from the GUI and put into the virtual wind tunnel of the cellular automaton (see Figure 7).

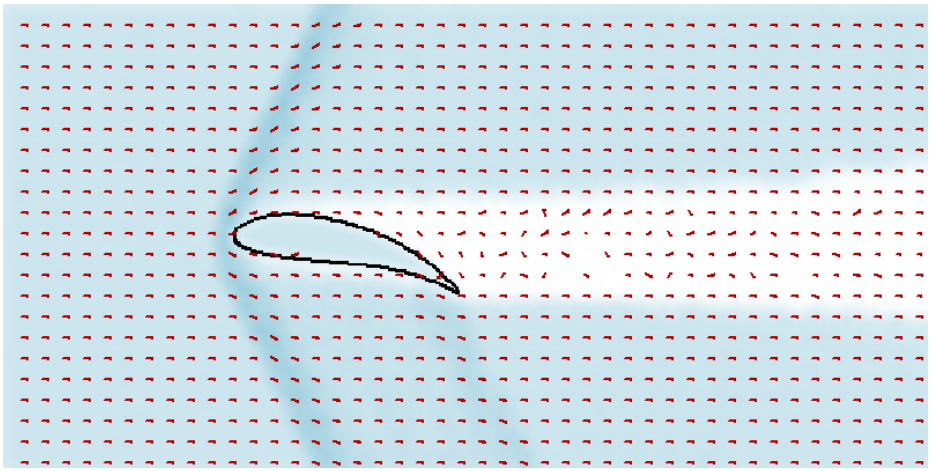

Figure 7: Uniform flow past a four-digit NACA airfoil. The mass density representation is superimposed by arrows illustrating the velocity field.[13]

As another example, the flow through a throttle valve of a carburetor is illustrated in Figure 8.

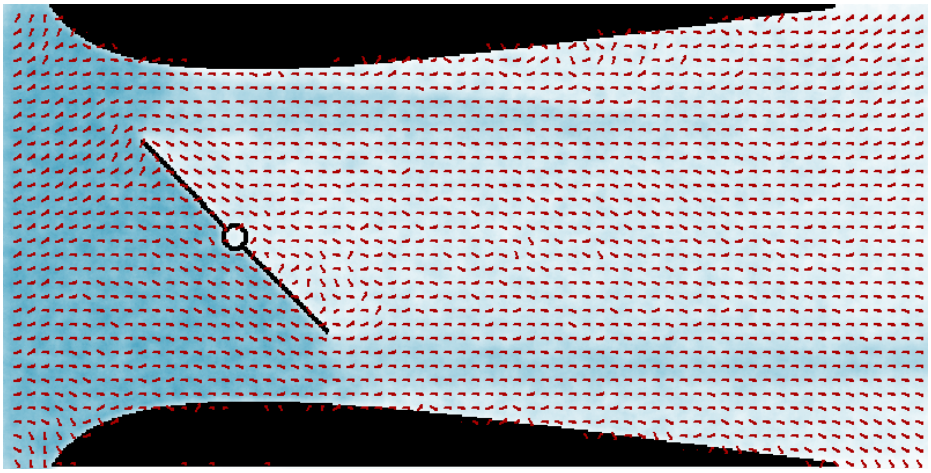

Figure 8: Uniform flow through a throttle valve. The mass density representation is superimposed by arrows illustrating the velocity field.[13]

In addition, the students provided a tool for the consecutive buffering of the flow field after each time step, which renders it possible to make a movie. In total, the members of this project team invested 179 hours for the completion of their task.

Conclusions

Starting from their freshman year, automotive engineering students at the Joanneum University of Applied Sciences are involved in project work within the framework of PBL. Software projects supplemental and complementary to the lectures exemplify the applicability of the just learned methods and mathematical algorithms, thus increasing the students' attentiveness and their appreciation for the currently learned topics.

In the last academic year, the development of lattice-gas cellular automata for the simulation and visualization of incompressible fluid flows was offered, among other topics, within the scope of such software projects. Four groups of first-year students in their second semester have accepted the challenge and developed independently and competitively computer programs that simulate and visualize the flow of incompressible fluids through and around various two-dimensional geometries.

The participating students were confronted, complementary to their regular courses, with a problem that was of a multidisciplinary nature and demanded profound skills and endurance. Additionally, by assigning this ambitious task the students were given the chance to look way beyond the standard curriculum of undergraduate engineering education.

One of the project teams has decided to continue with this topic in their sophomore year, and is currently programming an improved version of a FHP lattice-gas cellular automaton with the C programming language. Three other groups are simultaneously implementing a finite difference scheme for the solution of the two-dimensional macroscopic (Navier-Stokes) equations governing the dynamics of incompressible fluids within the framework of project based learning.

Bibliography

1. Emilia Bratschitsch, Annette Casey, Günter Bischof, and Domagoj Rubeša, 3-Phase Multi Subject Project Based Learning as a Didactical Method in Automotive Engineering Studies, ASEE Annual Conference Paper 1020 (2007)
2. John C. Tennehill, Dale A. Anderson, and Richard H. Pletcher, Computational fluid mechanics and heat transfer, Taylor & Francis (1997)
3. Dieter Wolf-Gladrow, Lattice gas cellular automata and lattice Boltzmann models: an introduction, Lecture notes in mathematics 1725, Springer (2000)
4. Stephen Wolfram, Statistical mechanics of cellular automata, Rev. Mod. Phys., Vol. **55**, No. 3 (1983)
5. John von Neumann, Theory of Self-Reproducing Automata, edited by A. W. Burks, University of Illinois, Urbana (1966)
6. Stanislaw Ulam, Some ideas and prospects in biomathematics, Ann. Rev. Bio., **255** (1974)
7. John Horton Conway, unpublished (1970)
8. Martin Gardner, Mathematical Games, Sci. Amer. **223**, Oct. 120-123 (1970)
9. Stephen Wolfram, A New Kind of Science, Wolfram Media Inc. (2002)
10. J. Hardy, Y. Pomeau, and O. de Pazzis, Time evolution of a two-dimensional model system: Invariant states and time correlation functions, J. Math. Phys. 14 (1973)

11. Uriel Frisch, Brosl Hasslacher, and Yves Pomeau, Lattice-gas automata for the Navier-Stokes equation, Phys. Rev. Lett. **56** (14) (1986)
12. Jean-Pierre Rivet and Jean Pierre Boon, Lattice Gas Hydrodynamics, Cambridge University Press (2001)
13. Peter Apolloner, Gerhard Hofer, and Benjamin Huber, CA-Sim, Technical Report, Joanneum University of Applied Sciences, Graz, Austria (2011)
14. Eastman N. Jacobs, Kenneth E. Ward, and Robert M. Pinkerton, The Characteristics of 78 Related Airfoil Sections from Tests in the Variable-Density Wind Tunnel, NACA Report No. 460 (1933)