FRIEDRICH-ALEXANDER-UNIVERSITÄT ERLANGEN-NÜRNBERG
INSTITUT FÜR INFORMATIK (MATHEMATISCHE MASCHINEN UND DATENVERARBEITUNG)

Lehrstuhl für Informatik 10 (Systemsimulation)

Simulation of bloodflow in aneurysms using the Lattice Boltzmann method and an adapted data structure

Jan Götz

Technical Report 06-6

Abstract

Unruptured intracranial aneurysms are a major public health issue in every developed nation. To diagnose and treat these diseases, doctors base their decision on modern imaging techniques and their knowledge. However, the hemodynamic situation in the vessels can hardly be measured or visualized, but might be essential for further treatment of the patient. A bloodflow simulation could gain additional information, but is memory and computational expensive. Here a method is investigated, which reduces the memory requirements and the computational time of this type of simulation. The fluid flow is calculated using the Lattice Boltzmann method with an adapted data structure, where a domain decoupling is used to erase unneeded areas. With this technique, the memory requirements are reduced about 90% in a standard aneurysm geometry. In addition to velocity and density values, which are the standard output of the Lattice Boltzmann method, the von Mises stress tensor can easily be calculated and visualized.

1 Introduction

Aneurysms are local dilations (baloonings) of the vessel wall and most commonly occur in the base of the brain or near the aorta. About 80% of these malfunctions are caused by arteriosclerotic diseases. The big danger of aneurysms is their rupture and intracranial hemorrhage into the subarachnoid space. This often results in a stroke. To diagnose these diseases, modern techniques like computer tomography angiography, magnetic resonance imaging, or digital subtraction angiography are used (see e.g. [DE06]). In recent studies it has been suggested that hemodynamic factors such as intraaneurysmal pressure and wall shear stress may be responsible for the development and growth of aneurysms [Ros98, NHea91].

In the field of computational fluid dynamics (CFD), several methods have been proposed based on the mathematical description of the flow field, namely the Navier Stokes equations. These methods, like the finite element method (FEM), or the finite volume method (FVM), are mostly used in todays commercial fluid solvers. With the development of computer power, these solvers were then ready to solve complex flows in complex geometries.

In the last two decades another approach was developed and became an attractive alternative to conventional CFD solvers. It is called the Lattice Boltzmann method (LBM) and has its origin in the Lattice Gas Automata (LGA). The LBM was used in a wide range of applications, e.g. simple fluid dynamic problems, flow through porous media [Thu03], or the movement of particles in a flow [Igl05, Fei06]. Furthermore, it seems to be a suitable method for handling complex geometries.

In contrast to standard methods, which solve the Navier-Stokes equations, the LBM deals with the solution of a special discretization of the kinetic Boltzmann equation. The most popular model was developed in 1992 by Bhatnagar, Gross and Krook (BGK model) and only uses first derivatives:

$$\frac{\partial \mathbf{f}}{\partial t} + \boldsymbol{\xi} \cdot \boldsymbol{\nabla} \mathbf{f} = \frac{1}{\lambda}(\mathbf{f} - \mathbf{f}^{(eq)})\,. \tag{1}$$

This model describes the evolution of a single particle distribution function $\mathbf{f}$ for particles that move with a microscopic velocity $\boldsymbol{\xi}$ and collide with relaxation time λ till they relax to the Boltzmann-Maxwellian equilibrium distribution function $\mathbf{f}^{(\mathbf{eq})}$. λ depends on the fluid viscosity. To obtain a solution from Equation 1, the velocity is discretized by a finite set of velocities $\xi_\alpha, \alpha = 0, 1, \cdots N$, where α is the direction and N is the number of directions, resulting in:

$$\frac{\partial f_\alpha}{\partial t} + \xi_\alpha \cdot \boldsymbol{\nabla} f_\alpha = \frac{1}{\lambda}(f_\alpha - f_\alpha^{(eq)})\,. \tag{2}$$

For the three-dimensional problem, several velocity models have been proposed. The D3Q19 model is a good compromise in terms of stability and computational efficiency [MSYL00] and thus used here. The velocity space in the $D3Q19$ model is discretized in 19 distribution functions. Each cell contains one stationary velocity in the center for the particles at rest, six velocity directions along the Cartesian axes and 12 velocities combining two coordinate directions (see Figure 1). Thus the fluid particles are only allowed to move along these 18 velocity directions, or stay at rest.

The equilibrium distribution for all models is of the form

$$f_\alpha^{(eq)} = \rho^* \cdot w_\alpha \cdot [1 + \frac{3}{c^2}\mathbf{e}_\alpha \cdot \mathbf{u}^* + \frac{9}{2c^4}(\mathbf{e}_\alpha \cdot \mathbf{u}^*)^2 - \frac{3}{2c^2} \cdot \mathbf{u}^* \cdot \mathbf{u}^*] \quad , \alpha = 1, \cdots N\,, \tag{3}$$

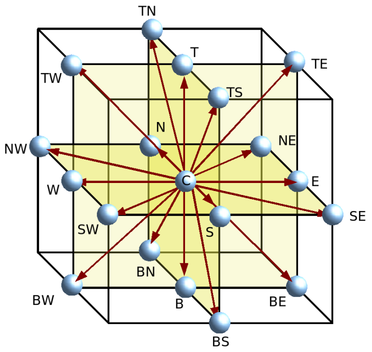


Figure 1: Model for three dimensions

where w_α is a weighting factor and $\mathbf{e}_\alpha$ is the given velocity vector. w_α and $\mathbf{e}_\alpha$ are dependent on the given model. For the D3Q19 model the weights are

$$w_\alpha = \begin{cases} 1/3 & \alpha = C \\ 1/18 & \alpha = N, S, E, W, T, B \\ 1/36 & \alpha = NW, NE, SW, SE, TN, TS, TE, TW, BN, BS, BE, BW \end{cases} .$$

To improve the readability, the numbers $\alpha = 0, \cdots 18$ were replaced by directional abbreviations, which consist of E (east), W (west), S (south), N (north), T (top) and B (bottom) and all relevant combinations of these abbreviations. The velocity vectors themselves are given by

$$\mathbf{e}_C = c(0,0,0)$$

$$\mathbf{e}_{T,B}, \mathbf{e}_{N,S}, \mathbf{e}_{E,W} = c(\pm 1,0,0), c(0,\pm 1,0), c(0,0,\pm 1)$$

$$\mathbf{e}_{TS,TN,BS,BN}, \mathbf{e}_{NE,NW,SE,SW}, \mathbf{e}_{TE,TW,BE,BW} = c(\pm 1,\pm 1,0), c(0,\pm 1,\pm 1), c(\pm 1,0,\pm 1)\,,$$

where $c = \Delta x^*/\Delta t^*$ denote the lattice speed with the lattice cell size Δx^* and the lattice time step Δt^*.

The hydrodynamic density value ρ^* and the velocity values $\mathbf{u}^*$ from Equation 3 are determined in terms of the particle distribution functions from:

$$\rho^* = \sum_\alpha f_\alpha \tag{4}$$

and

$$\rho^* \mathbf{u}^* = \sum_\alpha \mathbf{e}_\alpha f_\alpha\,. \tag{5}$$

Equation 2 is then discretized in space and time into the well-known lattice BGK equation [Art03]

$$f_\alpha(\mathbf{x}_i + \mathbf{e}_\alpha \Delta t^*, t + \Delta t^*) - f_\alpha(\mathbf{x}_i, t) = -\frac{1}{\tau}[f_\alpha(\mathbf{x}_i,t) - f_\alpha^{(eq)}(\mathbf{x}_i,t)]\,, \tag{6}$$

where τ is the lattice relaxation time. Equation 6 is typically solved in two steps:
Collide:

$$\tilde{f}_\alpha(\mathbf{x}_i, t + \Delta t^*) = f_\alpha(\mathbf{x}_i,t) - \frac{1}{\tau}[f_\alpha(\mathbf{x}_i,t) - f_\alpha^{(eq)}(\mathbf{x}_i,t)] \tag{7}$$

Stream:

$$f_\alpha(\mathbf{x}_i + \mathbf{e}_\alpha \Delta t^*, t + \Delta t^*) = \tilde{f}_\alpha(\mathbf{x}_i, t + \Delta t^*) \tag{8}$$

In the collide step, which can be interpreted as many particle collisions, new distribution functions are calculated according to the old time step. The equilibrium distribution functions, which are necessary to do the collide step, are calculated from Equation 3. After this step, all particles are streamed to their neighboring cells according to their velocity directions. This is done by just copying the values of the distribution functions to the neighboring cells.

The flow field and the shear stress play important roles in understanding, diagnosis and treatment of vessel malfunctions. Thus, in the following, the shear stress tensor is derived.

With the use of a Taylor expansion of Equation 6 up to the order $\mathcal{O}(\Delta t^2)$ and application of multiscale Chapman Enskog techniques by expanding f_α about $f_\alpha^{(0)}$ and introducing two timescales $t_0 = t$ and $t_1 = t\Delta t$ with the assumption $\lambda \sim \Delta t$, one can write the evolution equation as

$$(\partial t_0 + \mathbf{e}_\alpha \cdot \boldsymbol{\nabla}) f_\alpha^{(0)} = -\frac{1}{\tau} f_\alpha^{(1)} \tag{9}$$

to the first order and

$$\partial t_1 f_\alpha^{(0)} + (\partial t_0 + \mathbf{e}_\alpha \cdot \boldsymbol{\nabla})(1 - \frac{1}{2\tau}) f_\alpha^{(1)} = -\frac{1}{\tau} f_\alpha^{(2)} \tag{10}$$

to the second order. Equation 9 gives the continuity and momentum conservation equation to the first order

$$\partial t_0 \rho + \boldsymbol{\nabla} \cdot (\rho \mathbf{u}) = 0 \tag{11}$$

$$\partial t_0 (\rho \mathbf{u}) + \boldsymbol{\nabla} \cdot \boldsymbol{\Pi}^{(0)} = 0\,, \tag{12}$$

where $\boldsymbol{\Pi}$ is the momentum flux tensor. The tensor has the form

$$\boldsymbol{\Pi}_{ij}^{(0)} = \rho c_s^2 \boldsymbol{\delta}_{ij} + \rho \mathbf{U}_i \mathbf{U}_j\,, \tag{13}$$

where c_s is the speed of sound[1]. For the $D3Q19$ model the speed of sound is given by $c_s = 1/\sqrt{3}$. Following a similar way, the conservation equations can be derived from Equation 10 to the second order

$$\partial t_1 \rho = 0 \tag{14}$$

$$\partial t_1 (\rho \mathbf{u}) + \boldsymbol{\nabla} \cdot (1 - \frac{1}{2\tau}) \Pi^{(1)} = 0 \tag{15}$$

with

$$\boldsymbol{\Pi}_{ij}^{(1)} = \sum_\alpha f_\alpha^{(1)} \mathbf{e}_{\alpha i} \mathbf{e}_{\alpha j} = \Delta t c^2 \tau \left[(\frac{c_s^2}{c^2} - \frac{1}{3}) \delta_{ij} \boldsymbol{\nabla} \cdot (\rho \mathbf{u}) - \frac{1}{3} (\partial_i \rho \mathbf{U}_j + \partial_j \rho \mathbf{U}_i) \right]. \tag{16}$$

For incompressible fluids ($\boldsymbol{\nabla} \cdot (\rho \mathbf{u}) = 0$) Equation 16 simplifies to

$$\boldsymbol{\Pi}_{ij}^{(1)} = \Delta t c^2 \tau \rho \left[-\frac{1}{3} (\partial_i \rho \mathbf{U}_j + \partial_j \rho \mathbf{U}_i) \right] \tag{17}$$

or equivalently

$$\boldsymbol{\Pi}_{ij}^{(1)} = -\frac{2}{3} \Delta t c^2 \tau \rho \mathbf{S}_{ij}\,. \tag{18}$$

The strain tensor is therefore

$$\mathbf{S}_{ij} = -\frac{3}{2 \Delta t c^2 \tau \rho} \boldsymbol{\Pi}_{ij}^{(1)}\,. \tag{19}$$

It is customary to divide the total stress σ_{ij} into an isotropic part and a deviatoric part as in

$$\boldsymbol{\sigma}_{ij} = -P \boldsymbol{\delta}_{ij} + \boldsymbol{\tau}_{ij}\,, \tag{20}$$

where $\boldsymbol{\tau}_{ij} = 2\eta \mathbf{S}_{ij}$ is the viscous stress tensor and P is the (mechanical) pressure for an incompressible fluid. Substituting the previous formulas into Equation 20 and using the viscosity

$$\nu = \frac{1}{3} c^2 \Delta t (\tau - \frac{1}{2}) \tag{21}$$

results in

$$\boldsymbol{\sigma}_{ij} = -\rho c_s^2 \boldsymbol{\delta}_{ij} - (1 - \frac{1}{2\tau}) \sum_\alpha f_\alpha^{(1)} \mathbf{e}_{\alpha i} \mathbf{e}_{\alpha j}\,. \tag{22}$$

To simplify things, blood is treated as a Newtonian fluid, since to a fair approximation, the effect of non-Newtonian behavior of blood in arteries in the human brain is small and negligible [Kor03].

[1] Note, that the speed of sound mentioned here does not correspond to the physical speed of sound of the simulated fluid. It is the speed with which the information is transported through the grid.

2 Modified data structure

In a standard LBM solver two grids are needed, called the source and the destination grid with 19 distributions per cell. Additionally, macroscopic values like the pressure and the velocity components are stored. This results in an overall number of 42 floating point values per cell, which limits the simulation due to memory needs. To reduce the memory requirements, techniques like grid-compression can be used [Igl05], which has a reduction factor of about 50% compared to the original LBM solver. For a simulation of the human brain, a higher reduction can be achieved if unneeded cells are not stored. A very efficient data layout, which uses only related cells, is described in detail in [HKR05]. But this layout is not very flexible with regard to moving boundaries, thus another approach is used, which is discussed below.

Examples of 3D data sets of the human brain show, that only about 2-10% of the cells belong to the arterial blood vessel system. The rest of the cells are obstacle cells which are not needed for the simulation. To reduce memory requirements, the domain is split into equally sized boxes and only boxes are store, which contain fluid cells.
A 2D-example of this procedure is given in Figure 2. In a first step, the geometry is divided into boxes along Cartesian coordinate axis. These boxes contain an underlying lattice and information about their neighbors. If there are fluid cells in a box, it is marked and saved. All remaining boxes (colored gray in Figure 2c) are not necessary and thus not saved. In this example only 59% of the storage is needed.
The reduction factor regarding 3D-cases is even higher and in the range of $81 - 97\%$. For two exemplary data sets with a number of $256 \cdot 256 \cdot 400$ points and 16 boxes in each direction only $10-19\%$ of the memory compared to the standard layout were needed. To treat each box independently in the stream and collide step and to reduce data transfers, every box has an additional ghost layer which surrounds the box. The data of the ghost layers has to be updated between two time steps. The different types of connections are selected by the direction of the neighbor, which is saved in the data structure. Based on the direction of the neighbor, the corner, the edge, or the adjacent area of the two boxes is exchanged. For the D3Q19 model, only adjacent faces and edges have to be exchanged.

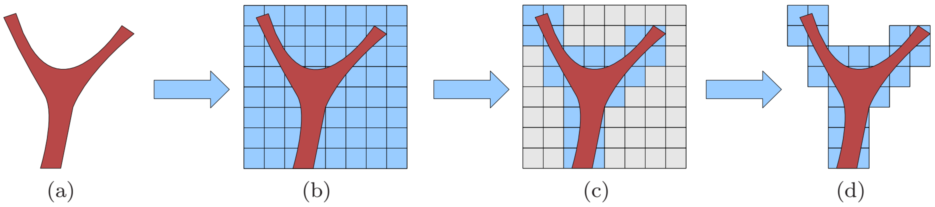


Figure 2: Domain uncoupling and reduction of a simple 2D-model of a bifurcation

3 Results

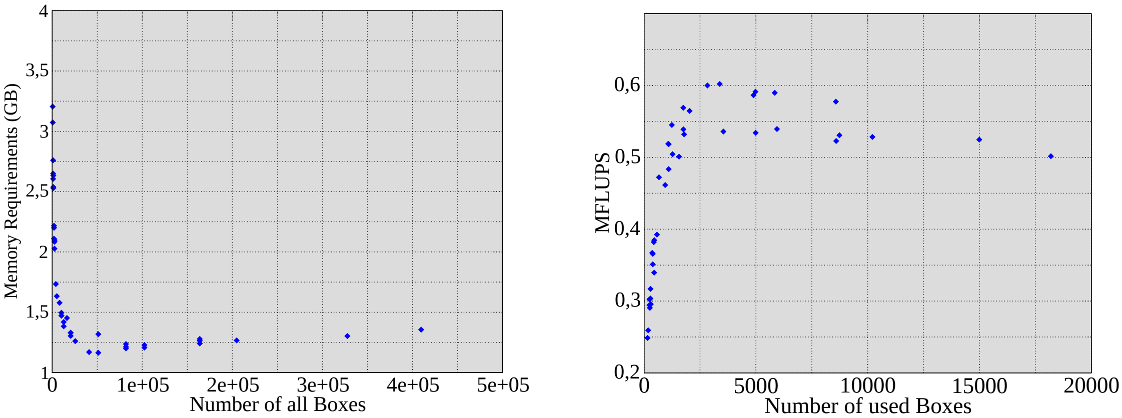
In the paragraph below, the memory requirements and the performance of the modified LBM are investigated. All simulations were done on one dual node of the LSS[2] cluster with an AMD Opteron CPU (2.2 GHz, 1 MB L2 cache, 4 GB DDR 333 RAM). Furthermore, the results of pressure and effective shear stress are presented for two relevant geometries.

The memory requirements for the simulation of a fixed domain size depend on the number of boxes that are used for the domain uncoupling. When the overall number of boxes increases, the size of the boxes is getting smaller, so the number of lattice cells in each box decreases. Hence, for a rising number of boxes more and more boxes are not necessary, because they do not include fluidal points. However, for the memory requirements not only the number of boxes is important. Also

[2] Lehrstuhl für Systemsimulation der Universität Erlangen-Nürnberg

the memory need of the ghost layer has to be taken into account. With increasing number of boxes the percentage of ghost cells surrounding the boxes also increases, which means higher memory requirements. Thus these two effects influence the overall demand, which is shown in Figure 3a. Up to a critical value of approximately 51,000 boxes, the memory requirements are decreasing, because they depend strongly on the percentage of used boxes. With more than 51,000 boxes, the increasing number of ghost cells is becoming dominant, resulting in higher memory needs. The minimum for this geometry ($\approx 1.16GB$) is reached with 51,200 boxes.
Different memory requirements for the same number of boxes come from different domain uncouplings. Depending on the geometry, there are more and less efficient uncoupling possibilities.

For performance issues, the MFLUPS[3] value was measured for a given geometry. For a rising number of used boxes the time for the stream collide step and the boundary treatment is reduced. This effect results from the fact, that every box is completely processed, even if it is nearly empty. With increasing number of boxes, every box gets densely filled with fluid cells, which means that mostly fluid cells are processed and no other cells, which are unnecessary for the simulation result. Unfortunately there exists an antithetic effect regarding time costs. Increasing the number of boxes leads to a higher percentage of interchanged ghost cells, which makes the ghost exchange time-consuming. These antithetic effects cause a maximum for the MFLUPS value of about 0.6 (see Figure 3b). This value is reached with 3,382 used boxes and a number of 51,200 overall boxes, which is also the most efficient splitting regarding memory costs. Compared to a standard LBM solver, only 10% of the memory is needed. As mentioned before, the efficiency regarding runtime depends on the domain uncoupling.



(a) Measurement of the memory requirement per number of boxes for a given geometry

(b) Measurement of the mega fluid updates per second per number of used boxes for a given geometry

Figure 3: Measurements of the memory requirements and the mega fluid updates

Stress is an important quantity regarding diagnosis and treatment of vessel malfunctions. With the Lattice Boltzmann method the nine Cartesian components of the stress tensor can be calculated directly without the need to estimate the velocity gradients. This reduces the computational costs significantly compared to standard FEM or FVM solvers. Moreover, a calculation of the stress using velocity gradients would introduce an approximation error due to the nonlinear velocity gradients at the wall. For visualization we use the von Mises effective stress, defined as

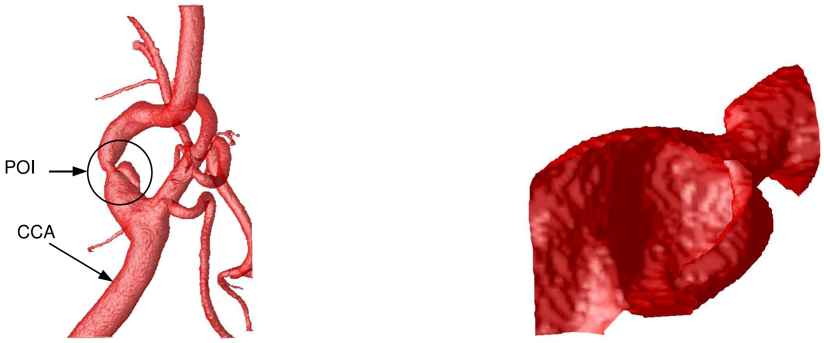
$$\sigma = \sqrt{\frac{(\sigma_{xx}-\sigma_{yy})^2+(\sigma_{yy}-\sigma_{zz})^2+(\sigma_{zz}-\sigma_{xx})^2+6(\sigma_{xy}^2+\sigma_{yz}^2+\sigma_{zx}^2)}{2}} \tag{23}$$

which is a second invariant of the stress tensor.

In the following, the results for a geometry are presented, where the common carotid artery (CCA) is inflow vessel. The CCA is one of the largest vessels in the human head in diameter. Figure 4

[3] Mega Fluid Updates per Second

shows the geometry with a stenosis and a small saccular aneurysm in front of the stenosis behind a bifurcation.



(a) complete artery

(b) Point of interest

Figure 4: Common carotid artery: Display of a stenosis and aneurysm.

The simulation is done with a time dependent inflow pulse of 1000ms (see Figure 5), the size of a LBM cell of $\Delta x = 0.12\ mm$, a fluid viscosity of $\nu = 4.0 \cdot 10^{-6} \frac{m^2}{s}$, a maximum inflow fluid velocity of $u = 0.96 \frac{m}{s}$, 80000 time steps and a domain size of $256 \cdot 256 \cdot 400$ cells, split in $32 \cdot 32 \cdot 50$ boxes in x,y and z-direction. This results in a physical time step of $1.25 \cdot 10^{-5} s$ and a τ value of 0.51042. The Reynolds Number, approximated for the diameter of the inflow vessel as relevant length scale, is 1,033.22. The simulation needs 3,382 of 51,200 boxes and uses about $1.16GB$ of memory. With a standard LBM solver the geometry would need about $8.6GB$.

The simulation is done on the complete geometry shown in Figure 4a, but the results are only visualized in the point of interest (POI) near the stenosis, which is shown in Figure 4b.

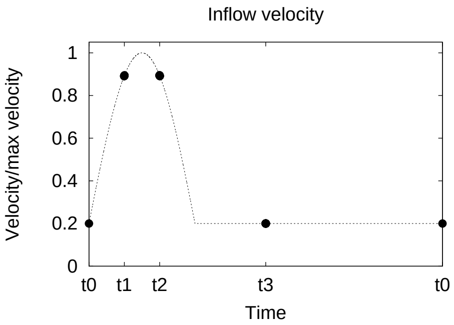
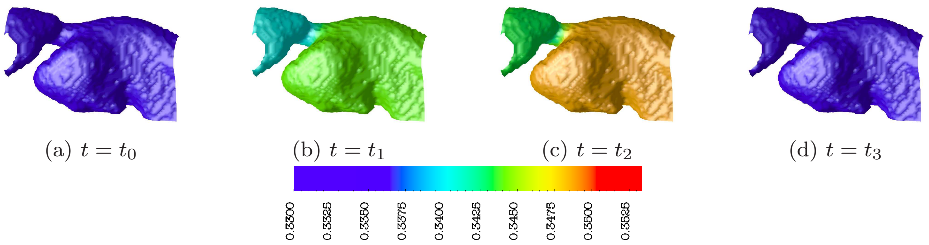


Figure 5: Relevant points at the inflow pulse

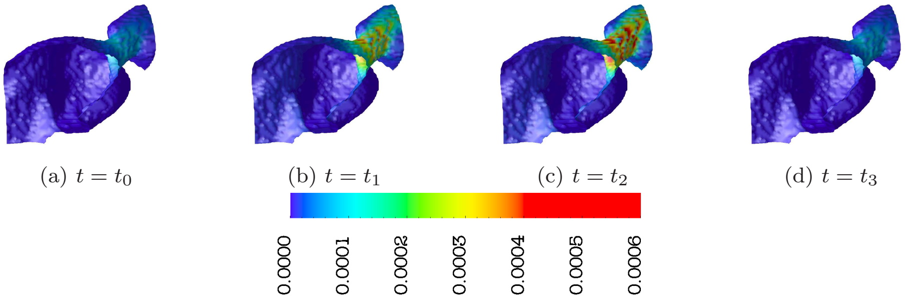
First experiments showed that a glyph representation of the pressure is difficult to understand and thus not useful. Hence, in Figure 6 the pressure is mapped to the arterial wall using a perspective, where the bloodflow is from right to left.

Analogous to the pressure, the von Mises effective stress is visualized by color-coding and mapping the values to the arterial wall. For the stress, a different perspective is used, which is shown in Figure 4b.



(a) $t = t_0$ (b) $t = t_1$ (c) $t = t_2$ (d) $t = t_3$

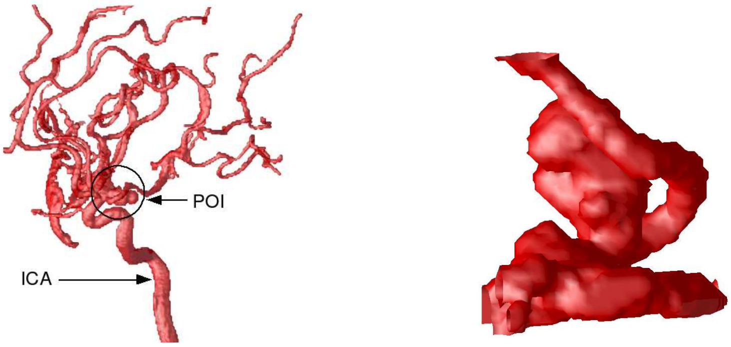
Figure 6: Time dependent pressure near the wall in lattice units at the point of interest



(a) $t = t_0$ (b) $t = t_1$ (c) $t = t_2$ (d) $t = t_3$

Figure 7: Time dependent von Mises effective stress near wall in lattice units at the point of interest

As a second example, a geometry of the internal carotid artery (ICA) is shown in Figure 8. This patient suffers from an aneurysm near the posterior communicating artery, which is the point of interest in this geometry. The flow enters the domain through the ICA and leaves it through several smaller vessels. As in the example before, a time dependent inflow pulse (see Figure 5) with a duration of 1,000 ms was used. The parameters of the simulation were $\Delta x = 0.12\ mm$ as the size of a LBM cell, a fluid viscosity of $\nu = 4.0 \cdot 10^{-6} \frac{m^2}{s}$, a maximum inflow fluid velocity of $u = 0.64 \frac{m}{s}$, 37000 time steps and a domain size of $660 \cdot 720 \cdot 672$ using $132 \cdot 144 \cdot 96$ boxes in x,y and z-direction. This results in a physical time step of $2.7 \cdot 10^{-5} s$ and a τ value of 0.51082. The Reynolds Number is 705.975, again approximated for the inflow diameter as relevant length scale. The simulation uses 19,458 of 1,824,768 boxes and has memory requirements of $2.95 GB$, which is less than 3% compared to a standard LBM solver.



(a) Complete artery (b) POI

Figure 8: Internal carotid artery: Display of an aneurysm

Figure 8b shows the interesting area of the geometry of the internal carotid artery. To get a better view on relevant values, the results are only shown for this area. The aneurysm occurs near the trifurcation of the ICA.

The pressure values are again color-coded and mapped to the arterial wall (see Figure 9) using the point of view illustrated in left figure of 8b. The results for the shear stress are shown in Figure 10).

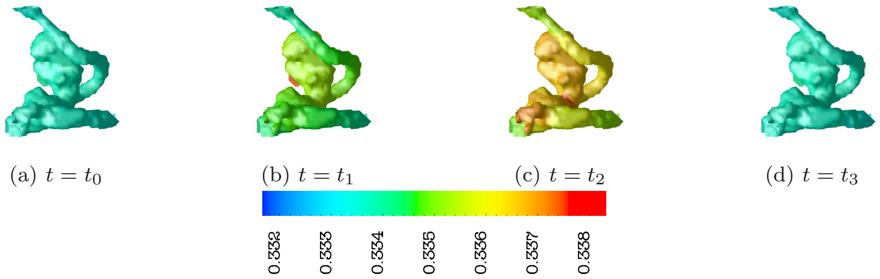

(a) $t = t_0$ (b) $t = t_1$ (c) $t = t_2$ (d) $t = t_3$

Figure 9: Time dependent pressure near the wall in lattice units at the point of interest

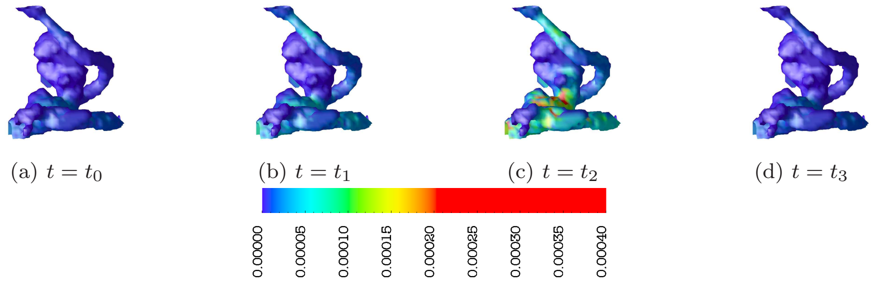

(a) $t = t_0$ (b) $t = t_1$ (c) $t = t_2$ (d) $t = t_3$

Figure 10: Time dependent von Mises effective stress near wall in lattice units at the point of interest

4 Conclusion

A hemodynamic simulation in the human brain is possible with techniques presented here. A Lattice Boltzmann method was implemented and adapted for the special geometry of the vessels in the brain. Due to a improved data structure, the requirements regarding memory and computational costs could be reduced significantly. Important geometries, based on real patient data, were simulated and the results were presented. The hemodynamic situation was visualized and discussed. However, quantitative physiological studies and the validation of the results are necessary to investigate the role of hemodynamics in the development and rupture of aneurysms. This could open the way for a clinical use.

References

[Art03] A. Artoli. *Mesoscopic Computational Haemodynamics*. PhD thesis, University of Amsterdam, 2003.

[DE06] A. Dörfler and T. Engelhorn. MR- und CT-Angiographie bei zerebrovaskulären Erkrankungen: Technik, Indikation und klinische Anwendung. *Radiologie up2date*, 6(101055S2006925018):49–64, 2006.

[Fei06] C. Feichtinger. Lattice-Boltzmann Simulation of Moving Charged Colloids. Master's thesis, University of Erlangen, 2006.

[HKR05] F. Hülsemann, M. Kowarschik, and U. Rüde, editors. *Optimizing Performance of the Lattice Boltzmann Method for Complex Structures on Cache-based Architectures.* ASIM, SCS Publishing House, 2005.

[Igl05] K. Iglberger. Lattice Boltzmann Simulation of Flow around moving Particles. Master's thesis, University of Erlangen, 2005.

[Kor03] Korea Advanced Institute of Science and Technology. *Blood Flow Simulation toward Actual Application at Hospital*, 2003.

[MSYL00] Renwei Mei, Wei Shyy, Dazhi Yu, and Li-Shi Luo. Lattice Boltzmann Method for 3-D Flows with Curved Boundary. *Journal of Computational Physics*, 161:680–699, 2000.

[NHea91] H. Nakatani, N. Hashimoto, and Y. Kang et al. Cerebral blood flow patterns at major vessel bifurcations and aneurysms in rats. *J Neurosurg*, 74:258–262, 1991.

[Ros98] S. Rossitti. Shear stress in cerebral arteries carrying saccular aneurysms: a preliminary study. *Acta Radiol*, 39:711–717, 1998.

[Thu03] N. Thuerey. A single-phase free-surface Lattice Boltzmann Method. Master's thesis, University of Erlangen, 2003.