Three dimensional visualization of simulations of liquids and solids

Joan Adler, Eden Segal and Grisha Zeltyn
Physics Department, Technion-IIT, Haifa, Israel, 32000
E-mail: phr76ja@technion.ac.il

Abstract. Visualization in three dimensions is invaluable for understanding the nature of condensed and fluid systems, but it is not always easy. In nature it is hard to view sample interiors, but on computers it is possible. We describe and contrast two opposite approaches - “smoke” visualization for viewing interiors of liquid samples and interactive WebGL for solids and molecules. Both are extensions of earlier Technion Computational Physics group projects and complement and are interoperable with the recent SimPhoNy Fp7 project. They require only desktop hardware and software accessible to students. Examples and standalone instructions for both are presented, starting with sample creation and concluding with image galleries.

1. Introduction
Let us ask “what is the best way to look into our three-dimensional world? Virtual reality is cool: for movies, gaming and research/teaching. Fortunately the first two spur software and hardware development that we can use for research and teaching. But they also spoil us with heavy expectations. There are many visualization codes for research/teaching - for a recent discussion of several that can interface well with simulation codes see [1]. The SimPhoNy project provided a way to connect simulations at different scales with visualization software, and provides an approach to integrating simulation codes with visualization ones in a standardised, repeatable framework. Its novelty was in the interoperability, rather than in new visualization approaches. Good documentation was also a focal point and wrappers to pass data between different codes were presented. We demonstrated interoperability [1] by showing the same .xyz data for electronic density with three different codes. This demonstration showed that concepts of color use, perspectve and so on are the crucial aspects, and code selection secondary.

We now present two further developments in a similar spirit based on the .xyz datafiles of the AViz code [2, 3, 4] One is an approach extending the AViz code to density simulation such as that generated from 3D Lattice Boltzman (LB) fluid simulations [5], and the other an implementation of WebGL [6] for nanotubes and crystals, extending earlier AViz studies. Both can be carried out on commodity boxes. Supplementary material relating to these projects is presented on [7].

Very large LB simulations [8] were part of SimPhoNy, but their visualization was carried with Computational Fluid Dynamics (CFD) codes and an HPC center visualization expert. We now describe a much simpler 3D LB simulation and its visualization with AViz which enables insight into the flow process from every desktop [9]. In the next section we will discuss our
previous approaches and then introduce a specific LB implementation that enables a novel approach to the 3D visualization of the flow. We will implement in AViz, but again the data files could be viewed with other codes. Following this, we will describe a new WebGL approach to nanotube modelling, again demonstrating interoperability with earlier AViz visualizations. Final discussion will center on the question “is it worth the effort?”

2. LB simulations and their websites
LB is an innovative approach to solving the Boltzmann equation, a partial differential equation governing the mass distribution of different fluid particles as a parameter of time, position and velocity. The equation is based on the assumption that different fluid particles are independent of each other except for collisions between them. The LB simulation quantizes each of the parameters: time is incremented in discrete steps and the positions can only fall on a grid of points. It is an example of a cellular automaton. A website simulation/visualization approach to cellular automata was presented in [10].

The novelty in this project is an educational approach with reproducible code for a commodity box. There are more sophisticated calculations in the literature [8]. Our local efforts have been focused on instructional websites and public codes [11, 12], details in the supplementary material [7] In [12], a GPU was also utilised in the visualization leading to an elegant, fast code that was somewhat demanding in terms of hardware and software requirements. The project code is not split into simulation and visualization parts; and the specific system demands (GPU and OpenGL, GLUT and GLEW) limit its future implementation, although the code is perfectly reproducible on suitable hardware and has been checked on different LINUX implementations. It includes a text-file method to create the porous structure in any desired shape. That hint of turbulence led to the idea of improved visualization techniques in order to observe this more closely, and in three dimensions.

3. Three dimensional density visualization with AViz
There are several approaches to 3d density visualization. Assuming that we do not have a convenient immersive cave; one good approach used for LB [8] is to show contour plots at selected locations and leave it to the viewer to interpolate between these. Another is to use “smoke” visualization with and without a stereo option. A third is to combine “smoke” with strategic slicing. We have explored the latter two choices for LB.

The separation of simulation and visualization enables a choice of visualization code. As noted above, in [1] we created visualizations for the same electronic density data in three different codes with the crucial issue being approach (as in selecting which parts of the data to show, rather than in the code selection). We direct the interested reader to electronic density visualizations in Figures 8, 9, and 10 of [1, 7].
4. Three dimensions for LB

The present LB simulation [5] overcomes the size and non-public domain limitations of the first [11], and the independent data access issues of the second [12] as well as the dimensional limitation of both but does not have a general porous shape selection option - only several preselected geometrical shapes. It is three dimensional; and although it works best on specific hardware (SIMD, details below) it can also be run on simpler systems and even has a Windows capability.

These present calculations follow the paradigm of splitting the effort into simulation and visualization, as per our group preferences and we present details of both. Downloadable code and all details can be found on the LB project website [5]. Like other LB simulations there are several stages in the simulation algorithm. The first is collision, including calculation of velocity and density, stored for later use in visualization. Then “streaming” or propagation to the next cell. Less straightforward is the fluid wall interaction, where particles reverse when they hit a wall. Boundary conditions, both at walls and inlet/outlet is the hardest part, the present code uses Zho-He conditions with details on the website.

The 3 dimensional algorithm’s choice of 3 dimensions and 19 velocity values is given the name Lattice Boltzman D3Q19 or D3Q19 for short. The velocities for this case take one of 19 different values, which together with implementing inlet and outlet boundary conditions even for $\tau \neq 1$ leads to a somewhat complicated implementation, but it is documented in detail on the website. $\tau$ is an inverse weight.

Hardware and software details and downloadable code are given on the website. The simulation has been designed to run on two levels of hardware specifications; the simpler non-SIMD one for all processors requires either gcc4.8 or gcc4.9 and the one suited for SIMD processors needs gcc4.9 Both use cpp support. SIMD (Single Instruction for Multiple Data) is present on Intel architectures newer than Haswell (from 2012) and AMD newer than Piledriver (from 2013). Multithreading with the OpenMP library that is part of the compiler makes the simulation even faster, because it distributes both the streaming and collision parts to the different threads in the processor. In the collision step, the distribution is carried out based on the z axis for the collision step, each thread is given some number of identical z planes. On the streaming step the distribution is based on the direction.

Another factor that increases the throughput is the memory layout which is an SOA (structure of arrays) approach. For each cell in our simulation there are 19 different values needing to be stored, one for each direction. We chose to store the values in 19 different arrays. This approach was used for the 2 simulation buffer, one for post streaming and one for post collision, and for the velocity buffers (which use only 3 arrays).

This SOA approach is very helpful in the SIMD case. SIMD means that each instruction works on a vector register of 256 bits, which are 8 floats, 4 doubles or some number of integers based on the size. Each instruction is performed on each float of the array independently on any other vectors element. For those instruction to be effective, the programmer needs to make each vector element be as similar to the other elements as much as possible. Because each of the pixels in the collision performs the same instruction irrespectively of the other pixels, making each vector element be a different pixel is a perfect candidate for this instructions. Loading such registers is best done by loading adjacent values in memory, and that is the reason we use the SOA order. The collision step operates on 8 pixels at the same time, each of those share the same y and z coordinates but differ in their x, and are then stored in the same way. The streaming step is done in exactly the same way. Detailed descriptions, sourcecode and pseudocode are given on the website.

The splitting of the simulation and visualization steps means that a fast processor can be used relatively easily for the heavier simulation part, and the hardware/software demanding visualization part be implemented on older but well-installed desktops at leisure and repeated
Figure 2. Nanotube, diamond, C_{440}, NaCl

as needed. The code generates data files in the .xyz format every 10 steps as a default.

As described above, for visualizations decisions on what to show, emphasize, manipulate, rather than which code is used are the most important so w.l.o.g. we use AViz. We aim to view “into” a semitransparent sample implementing both color binning [13] and slicing [14] where appropriate. We also use techniques to aid three dimensional insight including viewing an initial empty sample to establish perspective, and animations. An example of the empty box is given at left in the Figure 1, and three later shots center and right. AViz has several object options. For the entire sample shot we used the spin option for the fluid that shows orientation and direction with objects small enough to enable transparency. For the slicing we selected a “ball” object option so that the plane of the cut could be clearly viewed when sliced, although opaque when closed. The static cells are smaller balls. In both situations the same color bar is used, with static elements in blue or absent.

For these runs the sample dimension is 64*60*60 nodes, with a ball of radius 15 in the middle, and walls in the four enclosing planes perpendicular to the y and z axis. The boundary condition is that both the inlet speed and the outlet speed is 0.02 nodes/iteration time with $\tau=1.0$.

5. WebGL for nanotubes and crystals
This is an extension of a WebGL site [16] (http://phony1.technion.ac.il/~lsharir) that provides square and cubic lattice percolation visualizations for nanotubes and other structures with uniform bond lengths. The nanotube generation is based on an AViz nanotube website [17] and includes a generalization to all lattices with nearest neighbour bonds. WebGL is a version of the graphic library OpenGL viewed thru browsers that avoids the complications of OpenGL installation and can be viewed with browsers on smartphones and tablets. It is based on javascript, rather than the c/c++ basis of OpenGL and AViz. In common with AViz (that a uses OpenGL) our WebGL sites are also require/create .xyz files, and the limitations on WebGL (relative to the OpenGL/mesa implementation used for AViz) do not affect the visualizations presented below.

The AViz nanotube creation site is based on the [18] nanotube polyhedral method that correctly describes nanotube curvature. The code on the site enables creation of three types (Carbon, Boron and Silicon) of .xyz files for nanotubes with any desired parameters. These files have to be transformed for the WebGL input using the ChemDoodle Web Components Library [20] and a MATLAB script is provided for this. A translation to c/c++ is in preparation. Some still images [6] are presented in Figure 2 but even more so than for AViz images the beauty is in the rotate and zoom possibilities, which require a browser. They can be viewed on the website [7].
6. Conclusions

These, and other visualization techniques purport to enable relatively effortless three-dimensional visualization into sample interiors, the former for fluids, the latter for molecules and crystals. Considerable effort was made to develop both websites with clear instructions for users. For the former LB + AViz code, (AViz has validated instructions for systems programmer installation, [2, 3]) and a robust makefile for the LB code on a LINUX desktop are the sole requirements. The WebGL site is self-contained but for now requires MATLAB, an internet connection to the ChemDoodle site and a source of .xyz files. The images can be shown in any browser, including that on smartphones. The .xyz files are a bit more elusive. Nanotubes [17] can be self-generated on demand, a set of links for other materials [22] and a request page [23] are provided. There are also many .xyz files searchable on the web.

Given the potential benefit from these types of visualizations and the simplicity of the desktop orientation, they are well-worth the additional small effort. A selection of animated movies are presented in [5] and in our supplementary material, mirrored at [7].

Acknowledgments

The educational websites were part of the SimPhoNy project which was funded by FP7 under NMP-2013-1.4-1 call with grant agreement no: 604005. We thank Keijo Mattila, Liran Sharir and Simon Brandon for discussions and advice.

References

[1] Adler J et al. 2018 Visualization in the integrated SimPhoNy multiscale simulation framework Computer Physics Communications 231 45-61
[2] http://phony1.technion.ac.il/~aviz
[3] https://github.com/simphony/AViz
[4] Adler J 2003 Visualization in atomistic and spin simulations, Computers in Science and Engineering 5 61-65
[5] http://philafel.technion.ac.il/~edensegal
[6] http://philafel.technion.ac.il/~gzeltyn
[7] http://phony1.technion.ac.il/~phr76ja/vin3iop/mirror.html
[8] Mattila K, Puurtinen T, Hyvärinen J, Surmas R, Myllys M, Turpeinen T, Robertsén F, Westerholm J and Timonen J 2016 A prospect for computing in porous materials research: Very large fluid flow simulations Journal of Computational Science 12 62-76
[9] Peled D, Silverman A and Adler J 2013 3D visualization for atomistic simulations on every desktop IOP Conference Series 454 012076
[10] http://philafel.technion.ac.il/~idokesten/pro/about/about.htm
[11] http://philafel.technion.ac.il/~droden/project
[12] Adler J, Nissim G and Kiswani A 2017 GPUs in a computational physics course", J. Phys.: Conf. Ser. 905 012017
[13] Adler J, Fox J, Kalish R, Mutat T, Sorkin A and Warszawski E 2007 The essential role of visualization for modeling nanotubes and nanodiamond, Computer Physics Communications 177 19-20
[14] Adler J and Pine P 2009 Visualization techniques for modelling carbon allotropes Computer Physics Communications 180 580-2
[15] Grosso B, Cooper V R, Pine P, Hashibon A, Yaish Y and Adler J 2015 Visualization of electronic density Computer Physics Communications 195, 1-13
[16] Adler J, Ellenbaum S and Sharir I 2017 Eur. Phys. J. Special Topics 226 737-747
[17] Mazvovsky D, Halioa G and Adler J 2012 Physics Procedia 34 1-5
[18] Lee R K F, Cox B J and Hill J M 2010 Nanoscale 2 859-872
[19] Adler J, Artzi Y, ben Bashat L, Izraeli T Y, Kreif M, Lavi I, Leibenzon A, Levi A, Schlesinger I, Toledano E, Peretz U, Weisler Y and Alon Yagil 2014 Journal of Physics: Conference Series 510 012003
[20] http://web.chemdoodle.com
[21] http://phony1.technion.ac.il/~aviz/xyzfiles.html
[22] http://phony1.technion.ac.il/~phr76ja/plugfest/
[23] http://phony1.technion.ac.il/~aviz/requests.html