Flow Visualization on Hierarchical Cartesian Grids

Stefan Roettger[1], Martin Schulz[2], Wolf Bartelheimer[3], and Thomas Ertl[1]

[1] Institut für Informatik (VIS), Universität Stuttgart, Germany
[2] Science+Computing GmbH, Tübingen, Germany
[3] BMW AG, München, Germany

Abstract. Due to the limitations of the traditional finite volume CFD approach modern Lattice-Boltzmann methods are becoming more and more widespread. The results of developing an efficient visualization and exploration tool based on the Lattice-Boltzmann solver PowerFlow are summarized here to give the reader a basic insight into the pros and cons of such an approach. Also, the implementation of an automotive soiling simulation, which has been incorporated into the visualization tool, is presented here.

1   Introduction and Motivation

In the flow visualization community many solutions have been presented to embed flow visualization techniques into interactive and immersive environments, which allow an intuitive exploration and manipulation of a CFD data set [6,5,3,1,2,10]. In order to solve the Navier-Stokes equations most of todays available simulation applications are based on the finite volume approach, which is well known and established in the automotive industry. More recent approaches such as PowerFlow from Exa Corporation [4] use Lattice-Boltzmann simulation algorithms based on hierarchically refined cartesian grids (see Figure 1), which allow the simulation mesh to be generated automatically. Also, these methods are well suited for massive parallelization. Wind tunnel validations at BMW AG, Munich have shown that the simulated flow coincides very well with real-world measurements, so with PowerFlow now being used as a standard flow solver at BMW, there was the demand for a visualization system that could take advantage of the special properties of the hierarchical data. The cartesian representation of the CFD data, however, implies several restrictions. In particular, the car geometry has to be stored and handled explicitly, since the simulation grid is derived by voxelization. For the same reason special care has to be taken when modeling near surface effects in the visualization environment.
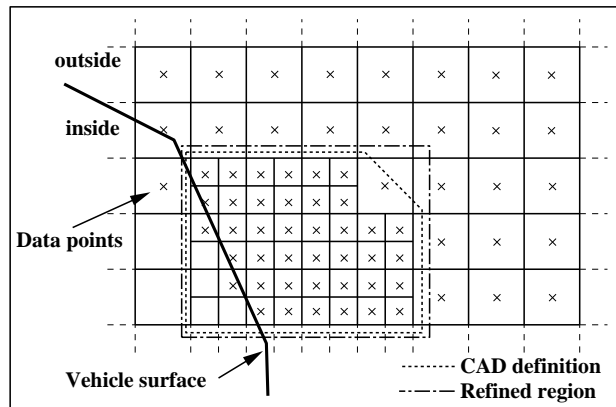
2   Hierarchical Representation

In our visualization application the hierarchical cartesian grid is stored in a compact way so that each voxel is primarily consuming memory for the scalar and vector components assigned with it. Each voxel can be accessed by a simple tree traversal
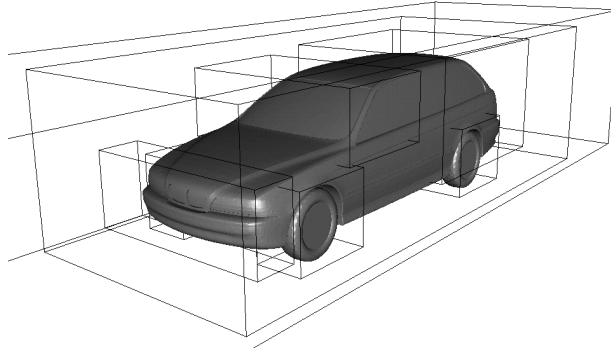
Fig. 1. Hierarchical cartesian grid, which is refined at the most interesting regions of a BMW 5 series for a Lattice-Boltzmann type CFD simulation.

of the hierarchy, which is illustrated in Figure 2. A typical data set consists of about 4 nested hierarchies containing approximately 3 million voxels. With one vector and 5 scalars loaded these account for a total of 120 megabytes.


Fig. 2. Hierarchical grid representation with explicit vehicle surface.

3   Visualization Methods

Because of the inherent hierarchical structure of the data set many of the established flow visualization techniques have to be adapted to the special properties of the hierarchical grids. For example, we are using an efficient Runge-Kutta particle tracer with adaptive step size control. The particle tracer utilizes an evaluation scheme of order 4(3), which has been shown to be optimally suited for streamline integration in cartesian grids [8,9]. For the purpose of particle tracing a method is also needed

to detect the hit points of a particle trace on the surface of the car. To achieve this aim we are using an octree representation of the car surface as presented in [7], which reduces the total number of visited triangles to an average of 8 to 15 triangles per octree search for a model with approximately 70.000 triangles. We can speed up collision detection even further by storing an additional "geometry bit" per voxel, which denotes the presence of geometry inside each voxel. With these fundamental algorithms we can now compute stream lines, ribbons and glyphs interactively. An example of all three methods combined with a slice probe that shows the pressure inside the data set is given in Figure 3. The stacked slice probe, as shown in Figure 4, is an efficient extension to traditional isocontouring methods exploiting 2D texture mapping and transparency clipping. These stacked slice probes are well suited for exploring the whole extent of a vortex and not just a single plane that slices through it.
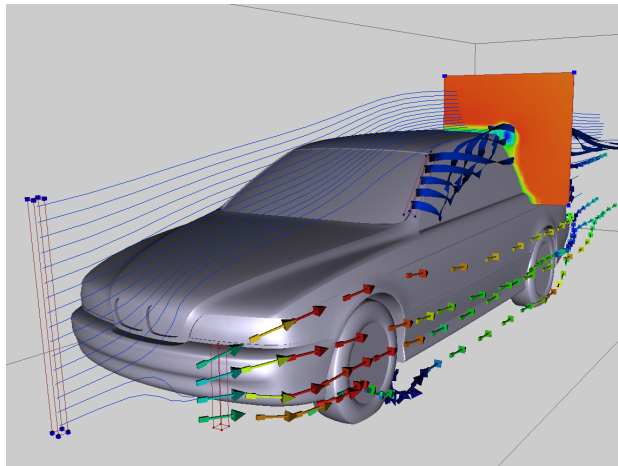

Fig. 3. Stream lines, ribbons, glyphs and a small pressure slice probe.

Investigating scalar values on the surface is of great importance to the fluid flow engineer. In Figure 5 for example the pressure on the car body is shown by means of coloring the so called surface elements (surfels). These surfels are computed by PowerFlow along with the voxels to compensate for the limited accuracy of the voxelization of the CAD geometry.

4 Interactive and Immersive Environment

All the aforementioned techniques can be used in an immersive environment like the PowerWall or the CAVE (see Figure 6). Space mouse and a variety of tracking devices are supported to allow intuitive three-dimensional orientation and manipulation of objects. A CAD geometry that consists of about 70.000 triangles can be
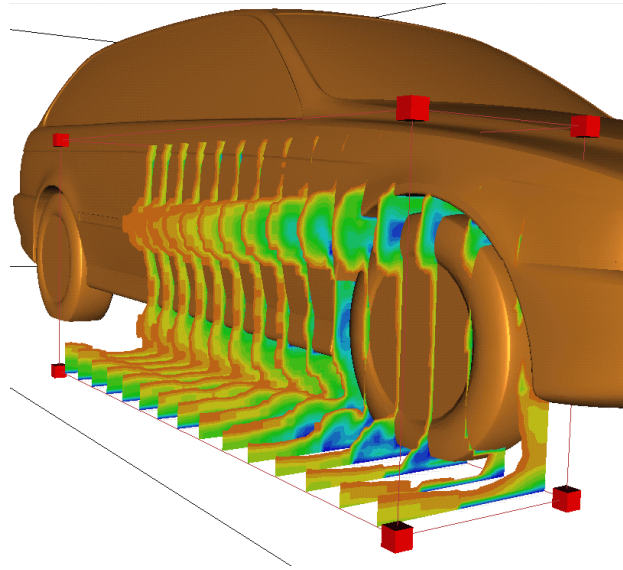
Fig. 4. Stacked slice probe.

Fig. 5. Surface pressure (bright colors depict high pressure).

displayed at a frame rate of approximately 15 Hz on an SGI Octane MXI. The frame rate drops to still interactive 12 Hz when using a PowerWall and an SGI InfiniteRe-ality2 with a single pipe for stereo rendering. The update times of the particle and slice probes depend on the specific settings but are interactive as well.
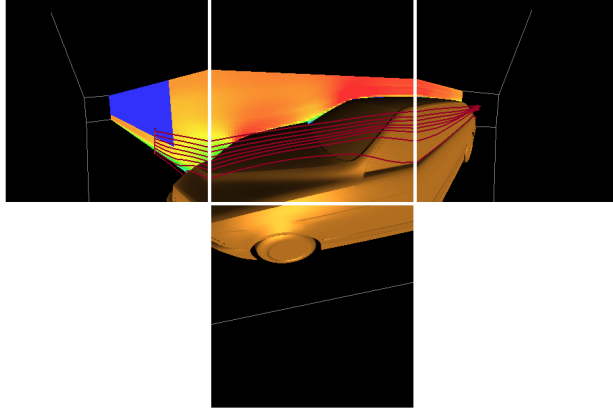
Fig. 6. The 4 side projections of a CAVE.

5 Particle Animation

Reconsidering the basic visualization toolkit we thought that introducing interactively animated particle probes would give the viewer a better physical and three-dimensional understanding of the flow properties. Therefore an animated particle system with user manipulable particle emitters was developed. The emerging stream of particles is visualized by means of a virtual camera with constant exposure time, such that particles with higher velocities leave longer traces on each frame of the animation (see Figure 7).
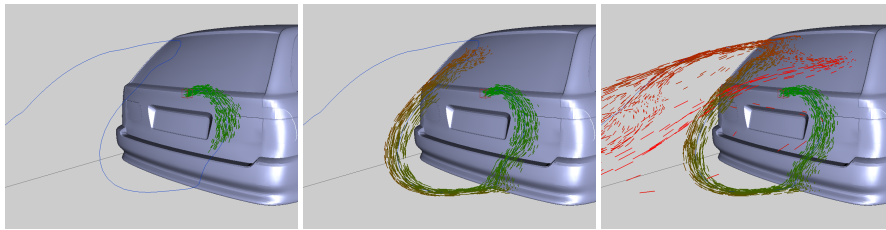

**Fig. 7.** Steps 1-3 of interactive particle animation.

Furthermore, we extended the adaptive Runge-Kutta particle tracer for the integration of point masses, which lead to an embedded particle tracer of order 4(3). Here our main goal was to trace and simulate real-world dust particles in a soiling simulation. Each particle is emitted in the extent of a box probe with stochastically varying diameter, specific mass and initial velocity. The forces, which drive the point masses depend on the velocities relative to the flow stream and the c_w-value of each particle. Assuming that the particles can be idealized as spheres the c_w-value of a dust particle can be approximated by the formula of *O'Seen*. This formula is suffi-

ciently exact for the low *Reynolds* numbers, which are encountered in automotive flow simulations.

With such a particle tracing system the accumulation of dirt on a car body can be visualized easily by coloring the hit points on the surface of the car. The adhesion probability of each particle depends mainly on the speed by which it is approaching the surface. Faster particles are more likely to hit the surface. Since flow velocity is always zero on the surface, the adhesion of each particle is driven by its momentum and by electrostatic effects. The latter effect is not yet completely understood and is subject to further research activities.

There are two other effects that introduce systematic errors when running a soiling simulation. First of all we are using a time-averaged flow field, which smoothes out the transient flow components and reduces the probability by which particles hit the surface. Secondly, the limited accuracy of the cartesian grid representation causes the flow velocity to be non-zero on the surface in contrast to the correct physical model. In order to compensate for that we define a small distance range around the car, in which the flow velocity is attenuated by a square root term depending on the distance to the surface of the car. Again, the distances can be calculated efficiently by using an octree representation of the surface triangles.

Our particle system is capable of tracing about 1000 point masses simultaneously on an SGI Octane MXI. With its dual 250 MHz R10K processors the refresh rate of the animation is approximately 20 Hz. The result of a dirt accumulation simulation employing the techniques mentioned above is shown in Figure 8.
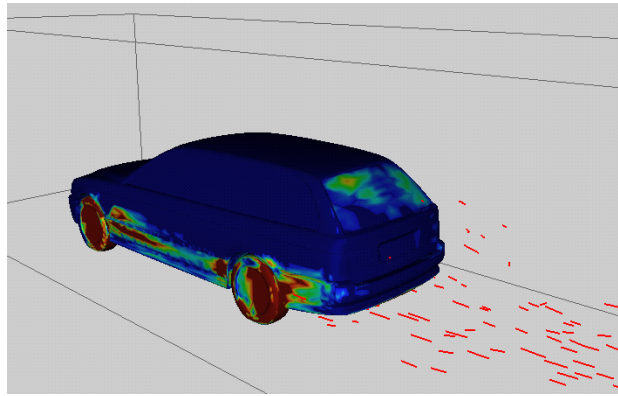
Fig. 8. Accumulated dirt after simulating dust particles.

6   Conclusion

We have presented a flow visualization environment that is optimized for the hierarchical grid structure of the Lattice-Boltzmann CFD solver PowerFlow. The development of a prototype of the visualization environment began in the second period

of FORTWIHR, the *Bavarian Consortium for High Performance Scientific Computing*, and was sponsored subsequently by FORTWIHR III. By utilizing efficient memory structures, fast particle tracers and hardware texture mapping we are able to explore the hierarchical CFD data sets in real-time and in immersive environments. Furthermore, a particle tracing system has been developed that is utilized for soiling simulations.

References

1. S. Bryson and C. Levit. The Virtual Windtunnel. In *IEEE Computer Graphics and Applications, 12(4):25-34*, 1992.
2. Steve Bryson. Approaches to the Successful Design and Implementation of VR Applications. In *Proc. SIGGRAPH'94*, 1994.
3. Steve Bryson and Steven Feiner. Virtual Environments in Scientific Visualization. In *Virtual Reality for Visualization, Course Notes of Tutorial 5 at Visualization 95*, 1995.
4. Exa Corporation. *http://www.exa.com*.
5. D. A. Lane. Scientific Visualization of Large-Scale Unsteady Fluid Flows. In G. Nielson, H. Hagen, and H. Mueller, editors, *Scientific Visualization*, pages 125–145. IEEE Computer Society, 1997.
6. F. Post and T. van Walsum. Fluid Flow Visualization. In H. Hagen, H. Mueller, and G. Nielson, editors, *Focus on Scientific Visualization*, pages 1–40. Springer Berlin, 1997.
7. M. Schulz, F. Reck, W. Bartelheimer, and Th. Ertl. Interactive Visualization of Fluid Dynamics Simulations in Locally Refined Cartesian Grids. In *Proc. Visualization '99*. IEEE, 1999.
8. C. Teitzel, R. Grosso, and T. Ertl. Efficient and Reliable Integration Methods for Particle Tracing in Unsteady Flows on Discrete Meshes. In W. Lefer and M. Grave, editors, *Visualization in Scientific Computing '97*, pages 31–41, Wien, 1997. Springer.
9. S.K. Ueng, C. Sikorski, and K.L. Ma. Efficient Construction of Streamlines, Streamribbons and Streamtubes on Unstructured Grids. *IEEE Transactions on Visualization and Graphics*, 2(2):100–109, June 1996.
10. S. P. Uselton. exVis: Developing A Wind Tunnel Data Visualization Tool. In R. Yagel and H. Hagen, editors, *Proc. Visualization '97*. IEEE, 1997.