Fluid Jet Simulations using Smoothed Particle Hydrodynamics

Erik Schnetter, Stefan Kunze, and Roland Speith

Institut für Astronomie und Astrophysik, Universität Tübingen, Germany,
Email: schnetter@tat.physik.uni-tuebingen.de,
URL: www.tat.physik.uni-tuebingen.de

Abstract. Our goal is to use Smoothed Particle Hydrodynamics to model the primary breakup of a Diesel jet as it is injected into the cylinder of an engine. We have performed two-dimensional simulations with parameters similar to those in a real Diesel injection process, and have identified some of the physical and numerical effects that have to be taken into account for a realistic simulation. We point out directions for future research.

1 Smoothed Particle Hydrodynamics

Smoothed Particle Hydrodynamics (SPH) is a grid free Lagrangian numerical method to solve the equations of hydrodynamics. The absence of a grid makes it particularly suited for simulation domains with highly irregular shapes, and its Lagrangian nature renders the treatment of advection terms almost trivial. Another key property is the ability to handle large density gradients well. A more in-depth discussion of the advantages and disadvantages of SPH is found in Monaghan [5].

In SPH, the matter is divided into small packets, known as *particles*. These particles follow the motion of the fluid while interacting with their neighbours. The particles do not exchange mass, but rather change their volume as the mass density of the fluid changes.

1.1 Basic Principles

In the SPH formalism, a continuum function $f^*(\mathbf{x})$ can be approximated by a function $f(\mathbf{x})$ using N particles that are located in space at the positions $\mathbf{x}_i$ and have the volumes V_i. This function $f(\mathbf{x})$ is defined through

$$f^*(\mathbf{x}) \approx f(\mathbf{x}) := \sum_i V_i \, f_i \, W(\mathbf{x} - \mathbf{x}_i) \quad .$$

This is a linear superposition of the particles' "sample values" f_i, weighted with the particle volumes V_i and a kernel $W(\mathbf{x})$. The kernel determines the shape of the particles. It is usually spherically symmetric and has compact support, i. e., it is zero for $|\mathbf{x}| \geq h$. The constant h is called *smoothing length* and determines the spatial resolution. Due to the compact support of the

kernel, only particles within a radius of h contribute to the value of f at any given point.

This locality is important for both physical and computational reasons. Physically, it renders particles independent when they are separated by more than h. Computationally, the algorithmic cost of SPH is proportional to the number of interactions. A limited number of interactions per particle is necessary to make simulations with large numbers of particles feasible.

From the above equation it follows that the dimension of the kernel W is an inverse volume. In order to be consistent, it is necessary that the sum of the individual particle volumes equals the total volume described by the particles. This can be expressed by the condition

$$1 \approx \sum_i V_i \, W(\mathbf{x} - \mathbf{x}_i) \quad .$$

This condition can, e. g., be satisfied by the choice

$$V_i := \frac{1}{\sum_j W(\mathbf{x}_i - \mathbf{x}_j)} \quad .$$

Another possibility is to choose consistent values for the V_i initially, and then evolve the particle volumes by

$$\frac{dV_i}{dt} = -V_i \sum_j V_j \, (\dot{\mathbf{x}}_i - \dot{\mathbf{x}}_j) \cdot \nabla W(\mathbf{x}_i - \mathbf{x}_j) \quad ,$$

which is justified by its similarity to the total time derivative of the previous equation.

Spatial derivatives of the continuum function $f^*$are approximated by the corresponding derivatives of f:

$$\nabla f^*(\mathbf{x}) \approx \nabla f(\mathbf{x}) = \sum_i V_i \, f_i \, \nabla W(\mathbf{x} - \mathbf{x}_i)$$

(where we use $\nabla f(\mathbf{x})$ as a short form for $\nabla f|_{\mathbf{x}}$). Because f depends spatially only on the kernel W, the continuity and smoothness of f and its derivatives are completely determined by the kernel. By choosing a suitable kernel, f can be made as often differentiable as necessary. In practice, however, it is preferable to only use first derivatives and express second derivatives as a sequence of two first derivatives using an intermediate quantity.

1.2 Hydrodynamics

The physical density field is approximated by $\rho(\mathbf{x}) = \sum_i V_i \, \rho_i \, W(\mathbf{x} - \mathbf{x}_i)$. After the substitution $\rho_i = m_i / V_i$ we get

$$\rho(\mathbf{x}) = \sum_i m_i \, W(\mathbf{x} - \mathbf{x}_i)$$

where m_i is the particle mass that is constant in time.

The equation of continuity in its Lagrangian form $D\rho/Dt = -\rho\nabla \cdot \mathbf{v}$ is translated into the SPH formalism by expressing the spatial derivative $\nabla \cdot \mathbf{v}$ as $\nabla \cdot \sum_j V_j \, \mathbf{v}_j \, W(\mathbf{x} - \mathbf{x}_j)$ and then replacing the continuum quantities ρ and $\mathbf{x}$ by the corresponding particle values ρ_i and $\mathbf{x}_i$.

Unfortunately, the resulting equation is not Galilei invariant. In order to cure this deficiency, one starts out with the slightly modified continuum equation $D\rho/Dt = -\rho\nabla \cdot \mathbf{v} + \rho\mathbf{v} \cdot \nabla 1$, which is mathematically correct because $\nabla 1 = 0$. Thus one gets the SPH equation

$$\frac{d\rho_i}{dt} = -\rho_i \sum_j V_j \, (\mathbf{v}_j - \mathbf{v}_i) \cdot \nabla W(\mathbf{x}_i - \mathbf{x}_j)$$

which yields zero when all particles have the same velocity.

In very much the same way the velocity field is approximated by

$$\mathbf{v}(\mathbf{x}) = \sum_i V_i \, \mathbf{v}_i \, W(\mathbf{x} - \mathbf{x}_i) \quad ,$$

and the Euler equation $D\mathbf{v}/Dt = -(1/\rho)\nabla p$ is translated into

$$\frac{d\mathbf{v}_i}{dt} = -\frac{1}{\rho_i} \sum_j V_j \, (p_j + p_i) \, W(\mathbf{x}_i - \mathbf{x}_j) \quad ,$$

in which the symmetry in the pressure term ensures conservation of momentum.

The equation of state $p(\rho, T)$ is realised particle-wise by setting $p_i = p(\rho_i, T_i)$. The addition of physical viscosity and an energy evolution equation is described, e. g., in Flebbe et al. [2] or in Speith [9]. Often an artificial viscosity is added as described in Monaghan & Gingold [6].

1.3 Separate Multi-Phase Fluids

We model multiple separate fluids as described in Ott [7]. We are mainly concerned with the simulation of two fluids with a large difference in density, as are Diesel oil and air. The total density and pressure of the SPH representation of the fluids are defined in the same way as described above. The basic new quantity needed to simulate multiple fluids is the *fluid fraction* $a(\mathbf{x})$, expressing which part of a certain volume is filled with which fluid. In our case, the fluid fraction expresses the volume fraction of Diesel. In the continuum it is obviously always $0 \leq a \leq 1$.

We define that each particle represents a packet of matter of only one kind of fluid, so that either $a_i = 0$ for an air particle or $a_i = 1$ for a Diesel particle. The approximate fluid fraction can then be calculated as $a(\mathbf{x}) = \sum_i V_i \, a_i \, W(\mathbf{x} - \mathbf{x}_i)$. It should be noted that this quantity can become slightly larger than 1 due to errors in the SPH approximation.

Luckily, the regions of space containing only one kind of fluid do not pose any particular problem that is not also found in the evolution of single fluids. The interface regions, however, require special attention.

In the neighbourhood of a Diesel-air interface one finds a region of width $2h$ (one smoothing length in every direction) where the total mass density is the sum of the two partial densities (see Fig. 1). While that might seem to indicate a physical mixture of both fluids, this effect is of a purely numerical nature. This is most easily shown by the fact that the width of that region depends on the resolution h, and that the whole region vanishes in the continuum limit $h \to 0$.

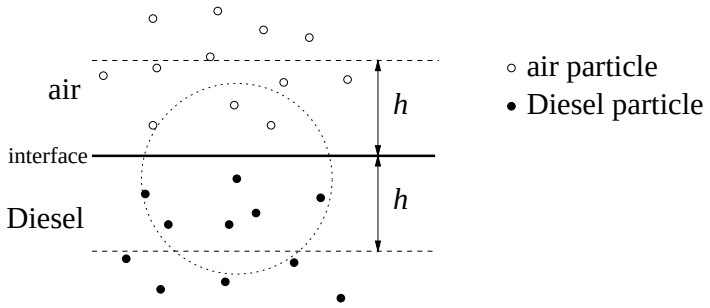


Fig. 1. Diesel-air Interface region. The dotted circle depicts the interaction radius h of a Diesel particle. This particle contributes to the density everywhere within that circle, even in the air region.

The total pressure in that region is also the sum of the two partial pressures. One particular problem in this region is the definition of a sensible equation of state $p(\rho, T, a)$. Ott finds that most naïve SPH formulations lead to instabilities in the interface regions. He shows that a stable evolution can be achieved by evaluating the two equations of state for the single fluids on the partial densities ρ_{air} and ρ_{Diesel} and taking the sum of the resulting partial pressures. The SPH equations presented above have been carefully formulated to take this principle into account and are thus applicable everywhere in the simulation domain, also in the interface region.

It might be conjectured that the instabilities found by Ott should also appear in cases where only a single fluid is present, but where the individual particle masses differ greatly. (Such a difference is often introduced to obtain a better spatial resolution in a certain region. The particles with smaller mass also have a smaller volume.) It would seem that the SPH formulation presented here should give better results in this case.

2 Implementation Details

The spatial resolution of SPH is determined by two key factors. One is the smoothing length h that has already been introduced above. It determines the size and shape of the individual particles and is thus a spatial lower limit to the features that can be represented numerically. The other key factor is the number of particles that overlap at a given point in space, often denoted as $N(\mathbf{x})$. The SPH particles move through the simulation domain without adhering to any particular local order, giving SPH a statistical component. Particles interacting at one time may become separated later on, while new particles move closer together and start interacting. This leads to a certain noise that is best counteracted by a smooth kernel W and a large number of interacting particles everywhere.

Increasing the number of interacting particles $N(\mathbf{x})$ while keeping the particle size h constant leads to an increase in the total number of particles needed and thus in computational resource requirements. Empirically one needs about 80 interacting particles in two dimensions and a good bit more than 100 in three dimensions. Exact numbers depend strongly on the details and have to be found in a case-by-case basis. One can think of it in the following way: A convergence study in SPH requires not one, but two parameters to be varied.

2.1 Resolution and Resources

A three-dimensional simulation requires at least 10^6 particles for a reasonable resolution; 10^7 or more particles are desirable. This means that for every evaluation of the right hand side about 10^9 particle-particle interactions have to be evaluated. These processing requirements could as of 1999 not be satisfied on a workstation. (In several years' time, when workstations with a comparable processing power will be available, we likely will have "adapted" our physical problems to scale with the available supercomputers.)

The memory requirements of an SPH implementation are relatively small when compared to the computing requirements. Storage for 10^6 particles requires about 500 MB of main memory, which is today already in the high end workstation range. The additional memory available on supercomputers can be used to speed up the search for interactions, as is described below.

2.2 The Heart of a SPH Implementation

A SPH simulation consists mainly of evaluating time derivatives of the form

$$\frac{df_i}{dt} = \sum_j C_{ij}\, V_j\, f_j\, W(\mathbf{x}_i - \mathbf{x}_j)$$

where C_{ij} is a rather simple expression depending on quantities of particles i and j only. These time derivatives are then integrated using a suitable

integrator; we usually prefer integrators of the Runge-Kutta type that are fast, easily available, and well tested.

For N particles there are of the order of N time derivatives to be evaluated, and each of the $\sum_j$ terms sums over N values, leading to an expensive N^2 algorithm. However, most of the terms in these sums will be zero because W has compact support. It is therefore crucial to evaluate only those terms where $|\mathbf{x}_i - \mathbf{x}_j| < h$ and to use an efficient algorithm that searches for the interacting particle pairs.

Our SPH implementation has to traverse all interactions up to three times per evaluation of the right hand side; once to determine the particle volumes (if so desired), once to calculate the first derivatives, and possibly a third time to calculate the second derivatives (if physical viscosity is used). The search for all interacting particles and the determination of the interaction kernels $W(\mathbf{x}_i - \mathbf{x}_j)$ and $\nabla W(\mathbf{x}_i - \mathbf{x}_j)$ is quite costly, and therefore the interaction information is stored and re-used. With the numbers mentioned above, a naïve implementation of such storing would raise the memory requirement by a factor of almost 100, as there are about 100 interactions per particle.

We have developed an algorithm as described in Kunze, Schnetter & Speith [3] that reduces these storage requirements substantially by introducing a stepwise evaluation. We traverse the simulation domain from left to right while evaluating the interactions. We do not store the whole set of interactions, but eventually drop the interaction information after having proceeded by some distance. This distance can be changed; by keeping it small, memory is saved, but the set of processors needs to be synchronised more often, resulting in a larger overhead and longer computing time. Depending of the size of the simulation domain and the shape of the particle distribution, only about 1 % of the interaction information might have to be kept in memory at a given time. This way, the memory requirements are again proportional to the number of particles. We usually keep the interaction information for a longer time than absolutely necessary and thus gain a higher speedup factor.

2.3 Parallelism

Thanks to MPI our code runs on many platforms without modifications. It has been tested on our local workstation cluster, a Beowulf cluster, the IBM SP/2, and the Cray T3E. It performs reasonably well; the load balancing takes only a negligible amount of time. Typical runs have 300,000 particles on about 30 nodes, where one right hand side evaluation takes a few seconds. The typical overall time spent waiting for communication is less than about 12 %.

Figure 2 shows the speedup and efficiency of our code on the Cray T3E in Stuttgart when run with varying numbers of particles. As expected, the number of processors used has to be adapted to the problem size when a reasonable efficiency is an issue. The lower bound on the number of processors

in these benchmark runs was given by the available memory per node. The upper bound was chosen according to the efficiency achieved and according to the batch queue congestion.

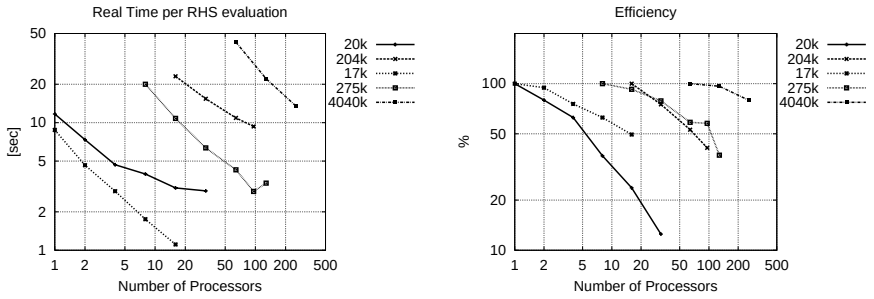


Fig. 2. Performance with varying numbers of particles. The top two runs are a 3D astrophysical application, the bottom three a 2D Diesel injection simulation as described below.

Runs with less then about 50,000 particles would for the user's convenience normally be placed on our local workstation cluster. They have been included in this benchmark for comparison only. We also normally prefer queues with fewer processors, because they tend to have shorter waiting times.

3 Simulation Setup and Results

A Diesel jet that is injected into the cylinder of an engine undergoes two stages of breakup. During the *primary breakup* the solid jet breaks up into several large drops and filaments, and during the *secondary breakup* these turn into a spray of droplets. While the secondary breakup is reasonably well researched and can be modelled as a spray, the primary breakup is mostly uncharted territory. The physical effects that might contribute to the primary breakup are the pressure forces of the air onto the surface of the Diesel jet, instabilities induced by surface tension, turbulence eddies, and cavitation bubbles created inside the injection nozzle.

Unfortunately, the injection is a high speed process occurring on small length scales under high pressure and temperature, and therefore experimental data are difficult to obtain. One goal of our numerical experiments is to study the physical effects mentioned above and determine their contribution to the primary breakup. We are currently still in the stage of modelling these effects.

3.1 Simulation Parameters

We decided to perform two-dimensional simulations, as they allow for a higher spatial resolution. One natural way to reduce an injection geometry to two dimensions is to postulate cylindrical symmetry. However, this leads to much dreaded $1/r$ terms at the axis of symmetry, and cylindrical coordinates destroy much of the beauty of SPH. We therefore decided to use a planar symmetry, i. e., a setup where all fields are constant along the z axis. Hence our simulations were performed in the x-y-plane.

In order to be able to compare the results of individual simulations we settled for a standard set of parameters, as listed in Table 1.

Table 1. Standard simulation parameters

simulation domain	:	$1\ \mathrm{mm}^2$
number of air particles	:	$250,000$
injected Diesel particles	:	$25,000$
initial particle volume	:	$4\,\mu\mathrm{m}^2$
nozzle width	:	0.1 mm
initial air pressure	:	$50,000\,\mathrm{hPa}$ (50 bar)
initial air density	:	$22.4\,\mathrm{kg/m}^3$
initial air temperature	:	$500\,^\circ\mathrm{C}$
air equation of state	:	adiabatic, $\gamma = 5/3$
air speed of sound	:	$610\,\mathrm{m/s}$
initial Diesel pressure	:	$50,000\,\mathrm{hPa}$ (50 bar)
initial Diesel density	:	$772.5\,\mathrm{kg/m}^3$
initial Diesel temperature	:	$100\,^\circ\mathrm{C}$
Diesel equation of state	:	tabulated
initial Diesel velocity	:	$400\,\mathrm{m/s}$
boundary conditions	:	fixed walls; axial symmetry at $y = 0$
artificial viscosity	:	none

We performed a parameter study varying the speed of the injected Diesel oil, its velocity profile, the coefficients of the physical and the artificial viscosity, the air pressure and density, and the spatial resolution. We typically ran a simulation until the tip of the Diesel jet got near the opposite wall, where the simulation becomes unrealistic: In reality, the cylinder is much larger than our simulation domain, and near our fixed wall the Diesel jet begins to slow down.

The figures below show snapshots of simulations at $t = 2 \cdot 10^{-6}$s. The Diesel jet is injected from the left. The lower boundary is a symmetry axis. The size of the injection nozzle is $1/10$ of the height of the simulation domain.

3.2 Standard Parameter Set

The most prominent feature of a run with the standard parameters, as shown in Fig. 3, is that the jet develops an anchor-like shape. Similar shapes have been found in experiments by Schugger [8], so we believe that this general feature is not a numerical effect. However, the jet does not break up, meaning that the simulation still lacks one or more crucial physical effects. The fact that the Diesel-air interface is dissolved does not indicate a breakup, as this dissolution happens on the length scale of the individual particles and is therefore resolution dependent. As shown further down, the boundary dissolution can be avoided by purely numerical means (which is also a sign of its non-physical nature). With a lower injection velocity of 200 m/s, the particle mixing becomes negligible.

Examining the pressure and the speed we find that the sonic waves created at the jet's tip to be propagated nicely. The first wave seems to propagate slightly faster than the speed of sound, indicating a very weak shock. At the time of Fig. 3 some waves have already been reflected from the opposite wall. While the reflection and the following superposition are physically correct, the opposite wall is not present in an engine. Thus the effects of the reflected waves on the jet's tip are interesting from a numerical point of view, but do not advance the study of the underlying problem of a primary jet breakup. We stopped the simulation at that point.

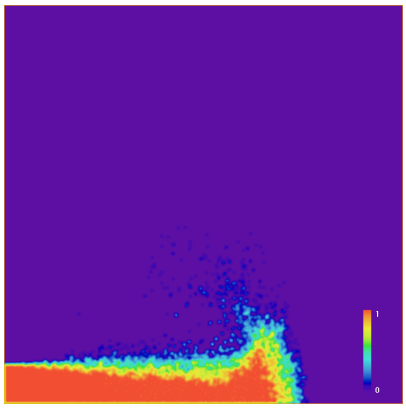
The interior of the Diesel jet also contains sound waves. These are initially perpendicular to the jet's velocity, i. e. parallel to the jet's tip, but get reflected from our injection boundary condition at the nozzle and start to interfere. The resulting interference pattern is visible in the pressure figure. Finally the speed figure shows an eddy in the air being created behind the jet's tip rotating counterclockwise with a velocity slightly faster than the Diesel jet.

3.3 Parabolic Velocity Profile

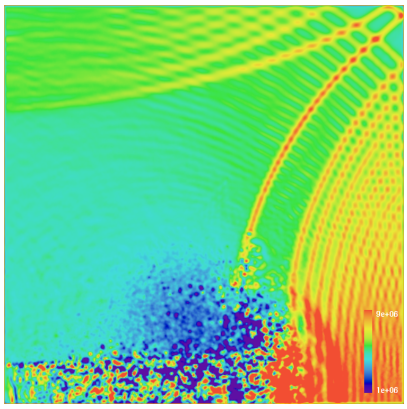
Simulations of the flow inside the injection nozzle undertaken by Bosch [1] show that the jet's velocity profile when it is injected is not radially constant, but decreases outwardly, while the velocity at the jet's outer boundary is nonzero. Qualitatively, a profile of this kind is to be expected from a turbulent flow. We changed the velocity profile of our injection boundary condition to resemble the measurements and chose a parabolic profile, where the velocity at the outer boundary is one half of the central velocity. A snapshot of a run with otherwise unchanged parameters is shown in Fig. 4.

In contrast to the run with a constant velocity profile shown above, the anchor-like shape does not form here. Photographs obtained by Schugger [8] show a large variety of shapes, leading us to expect a variety of shapes that depend on the exact details of the initial data.

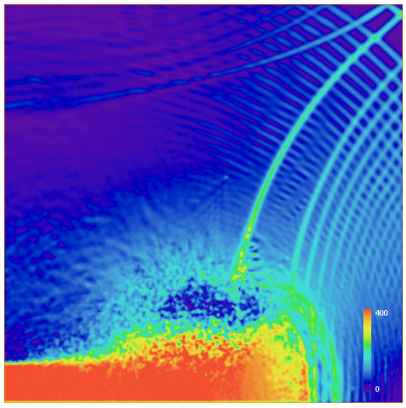
Unfortunately, the lower velocity at the jet's boundary does not preclude the unphysical particle mixing.



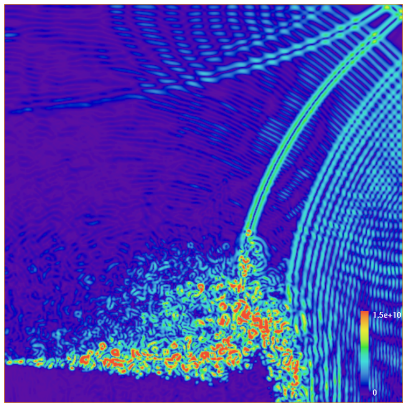
(a) Diesel volume fraction



(b) pressure

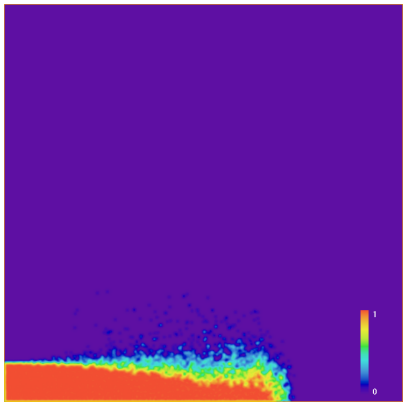


(c) speed

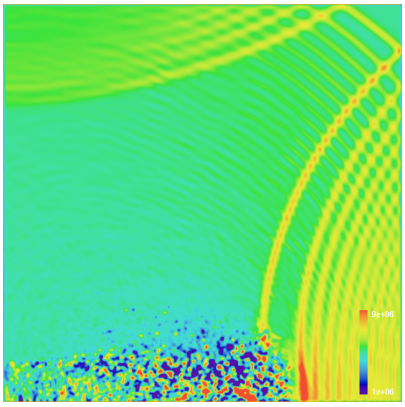


(d) acceleration

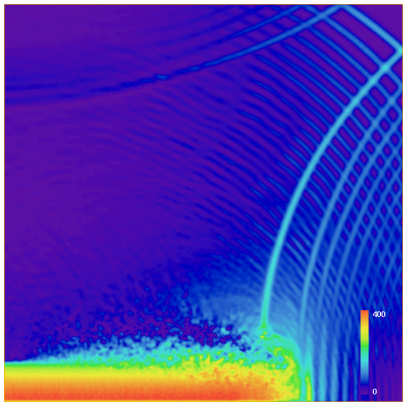
Fig. 3. Simulation with standard parameters



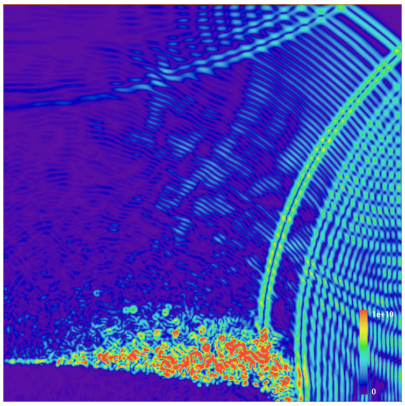
(a) Diesel volume fraction



(b) pressure



(c) speed



(d) acceleration

Fig. 4. Simulation with a parabolic velocity profile

3.4 The Role of Physical Viscosity

A unique feature of the brand of SPH formalism developed in our group is the inclusion of physical viscosity, which has come from a need in certain astrophysical simulations. The viscous time scales in this problem here are larger than our simulated physical time, so that one does not expect a change in the global features if physical viscosity is taken into account. Indeed, Fig. 5, which uses the same parameters as the previous simulation except that physical viscosity has been added, shows only slight changes. The edge of the jet's tip looks rounder, and the sound waves become somewhat blurred. The physical viscosity has only negligible effect on the unphysical particle mixing.

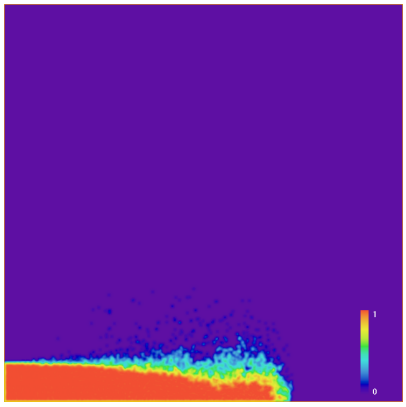
The need to evaluate the second derivatives in the Navier-Stokes equation almost doubles the execution time of our program. As mentioned above, we find that we get better results when the second derivatives are evaluated as a sequence of two first derivatives. This has a smoothing effect that helps to reduce the high frequency noise introduced by second derivatives (see Flebbe et al. [2]). The doubling of the run time is likely caused by the fact that the set of particles and the list of interactions have to be traversed more often, where the set of particles does not fit into the cache and has to be fetched from the main memory. Furthermore the boundary conditions have to be applied twice, leading to the processors being synchronised twice as often.

3.5 Artificial Viscosity

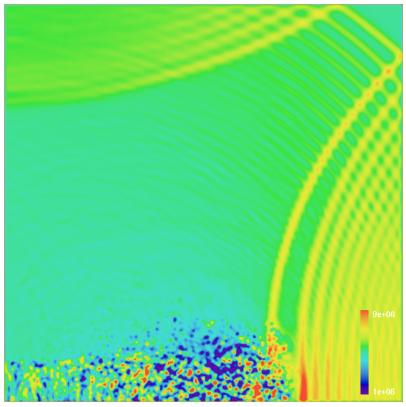
Artificial viscosity is a viscosity-like term that is added to the right hand side of the Euler equation. It dampens high frequency particle oscillations on the length scale of the spatial resolution. Due to its resolution dependence, artificial viscosity vanishes in the continuum limit. In the absence of physical viscosity, artificial viscosity is necessary to model shocks; by changing the Euler equations it removes the discontinuity across the shock. Additionally it has been found by Lattanzio & Monaghan [4] that artificial viscosity helps to prevent SPH particle penetration.

A simulation with a parabolic velocity profile and with an artificial viscosity coefficient of $\beta = 2$ successfully prevents the dissolution of the Diesel-air interface (see Fig. 6). A few small clumps of Diesel particles still leave the jet; this might be cured by the addition of surface tension terms.

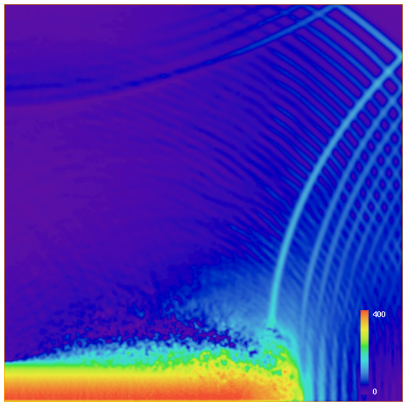
The artificial viscosity employed here also smoothes out the sound waves, leaving only the strongest first few wave trains in the air and inside the jet. Luckily sound waves are not believed to play an important role in the primary jet breakup, except maybe by triggering instabilities introduced by other effects, so that the dampening of the sound waves can be tolerated.



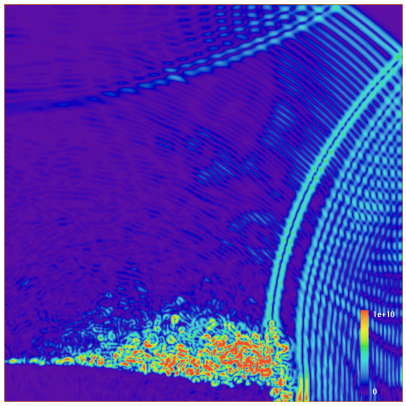
(a) Diesel volume fraction



(b) pressure



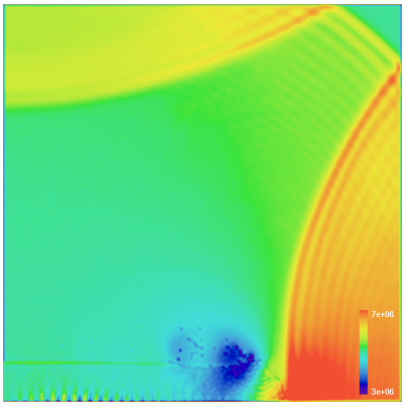
(c) speed



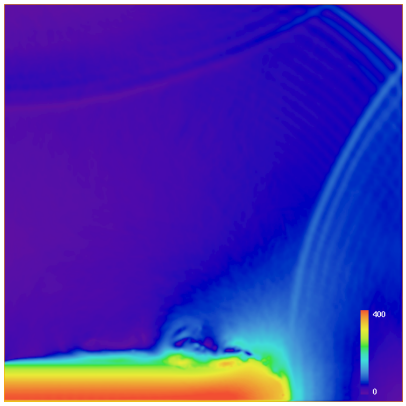
(d) acceleration

Fig. 5. Simulation with physical viscosity

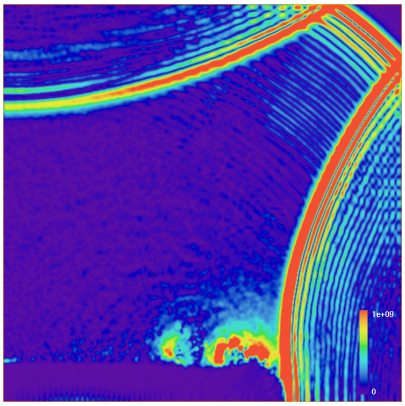
(a) Diesel volume fraction



(b) pressure



(c) speed



(d) acceleration

Fig. 6. Simulation with artificial viscosity

4 Future Work

None of the simulations performed by us showed a breakup of the jet as observed in reality. As mentioned above, likely breakup mechanisms include turbulence, cavitation, and surface tension. We have started to perform experiments with surface tension terms on workstations in our department. We are also planning to develop a turbulence model for SPH.

Our implementation of the SPH formalism was written with extensibility in mind. The basic part of the program, handling memory management, file I/O, and interprocessor communication, can remain unchanged while new physics is added to the code. This has enabled us to experiment with various physical effects, and will allow us to continue to do so without affecting the already optimised performance characteristics of the code. Our code is able to perform simulations in one, two, or three dimensions, and a parameter study in three dimensions is planned. (Recently, computing time for this project was granted on the HLRS Cray T3E.)

This work is based on the dissertation of Dr. Frank Ott, to whom we want to express our thanks. We also want to thank Dipl.-Ing. Ch. Schugger, Lehrstuhl für Wärmeübertragung und Klimatechnik RWTH Aachen, for his photographs, and the company Bosch, represented by Dr. B. Heine, for making available experimental and simulational data.

References

1. Firma Bosch. Simulation of the Velocity Profile in an Injection Nozzle. Internal note, 1998
2. O. Flebbe, S. Münzel, H. Herold, H. Riffert, H. Ruder: SPH: Physical Viscosity and the Simulation of Accretion Disks. *ApJ*, **431**, 754–760, 1994
3. S. Kunze, E. Schnetter, R. Speith: Development and Astrophysical Applications of a Parallel Smoothed Particle Hydrodynamics Code with MPI. In: E. Krause, W. Jäger (eds.): *High Performance Computing in Science and Engineering '99*, 52–61, Springer 2000
4. J. C. Lattanzio, J. J. Monaghan, H. Pongracic, M. P. Schwarz: Controlling penetration. *SIAM J. Sci. Stat. Comput.*, **7**, 591–598, 1986
5. J. J. Monaghan: Smoothed Particle Hydrodynamics. *Annu. Rev. Astron. Astrophys.*, **30**, 543–574, 1992
6. J. J. Monaghan, R. A. Gingold: Shock Simulations by the Particle Method SPH. *J. Comp. Phys.*, **52**, 374–389, 1983
7. F. Ott: Weiterentwicklung und Untersuchung von SPH im Hinblick auf den Zerfall von Dieselfreistrahlen in Luft. Dissertation, Universität Tübingen, 1999
8. Ch. Schugger: Photographs of injected dodecane from experimental results, Lehrstuhl für Wärmeübertragung und Klimatechnik, RWTH Aachen. Private communication, 1998
9. R. Speith: Untersuchung von Smoothed Particle Hydrodynamics anhand astrophysikalischer Beispiele. Dissertation, Universität Tübingen, 1998