A 6th order parallel CFD code based on an ENO-Padé scheme

Håkan Kihlström
hakki470@student.liu.se

Kristofer Lindberg
krili389@student.liu.se

Mark E. Dieckmann
mardi@itn.liu.se

Matt Cooper
matco@itn.liu.se

Norrköpings Visualiserings och Interaktionsstudio (NVIS), ITN, Linköpings Universitet

1 Introduction

Contemporary computer resources place the rendering of complex scenes for the professional film- computer games- and animation industry within range for physically accurate simulations. Models involving rigid body simulations, simulations of deformable objects, fluids and gases are increasingly being used to replace more traditional methods in animation and modeling. An important type of simulation is that for modeling fluids. Potential application areas are the simulation of water flow through the realistic modeling of flames to explosions. Simulations of liquid flow are an integral part of many recent movies and the real-time simulation of water is finding its way also into computer games like Half-life 2. However, most of these simulations solve the equations for incompressible fluids which excludes fluid compression and thus shock solutions and explosions. Without a physically accurate simulation we have to resort to procedural methods or geometric modeling to obtain animations of explosions. Explosions are however difficult to model since it is not generally possible to decouple the dynamical motion of the medium over multiple length- and time scales.

2 Exposition

Here we present our recent work on developing a 6^{th} order accurate simulation code that solves the Euler equations for compressible fluids which is discussed in [Wang and Huang 2002]. The Euler equations are a set of three coupled differential equations describing the transport of mass, momentum and energy in fluids. Specific properties of fluids are set by physical constants. The Euler equations are non-linear and the simultaneous occurence of processes on multiple time- and spatial scales complicates their analytic and their numerical solution. Typically, large spatio-temporal grids are required to resolve the large-scale structure of the fluid while ensuring that any waves with significant power that may be produced by physical processes have wave lengths larger than the simulation cell size. The required grid sizes pose challenges even to contemporary computer hardware. Our code is thus custom-written for parallel computers using the Message Passing Interface (MPI). The scheme also uses a fully implicit time discretization which will set more relaxed conditions on the time step chosen for the simulation. Further more we will discuss physical results and show simulation benchmarks obtained on Monolith, the massively parallel Linux cluster at Linköpings National Supercomputer Centre (NSC).

3 Results

To describe the performance of our numerical fluid solver we present a test case. The example shows a simulation that uses 560 cells and is evolved over 2000 time steps. To test the physical accuracy and the stability we have chosen initial conditions that lead to a shock-solution. The results are shown in figure 1. The figure show the fluid density in the system as a function of space and time. We have initialized the simulation with a rectangular density profile that moves at a high speed to the left relative to the surrounding

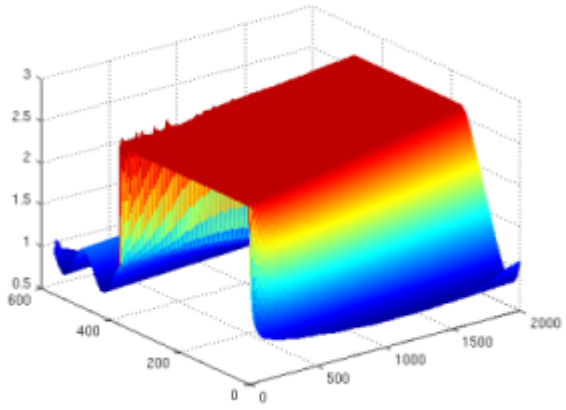

Figure 1: Plot of density over time.

medium. Ahead of the shock we have introduced some further sine-shaped oscillations. As you can see in the figure the backside of the density profile flattens. It reflects the suction force that the medium behind the density jump exerts on the fast-moving medium. The shock front shows a few wiggles but its slope constitutes a discontinuity without destabilizing. The ENO scheme has prevented the shock to smear out and the Pade scheme has provided excellent numerical accuracy across areas outside the shock discontinuity. This is in line with our expectations. We have also done efficiency tests on the code by comparing the simulations with identical number of grid cells and time steps for which we vary the number of processor nodes. If the code would scale perfectly we would expect a linear speed-up of the system with the number of processors. Our speed-up is not ideal. We obtain for example a speed-up of 2.25 for 8 processors instead of the ideal 8. These simulations do however only use a low number of cells per processor. This gives a problematic ratio of computations to communications. We can not yet extend the simulations to more realistic number of cells because we have not yet distributed the memory over individual processor nodes. We have also used an un-optimized matrix inversion scheme.

4 Future work

Currently our solver handles only one spatial dimension. As we commence our work of extending the solver to the fully three dimensional case we will also distribute the memory over the individual processor nodes and optimize the code's performance to allow for large simulation grids.

References

Wang, Z. Huang, G. P. 2002. An essentially nonoscillatory high-order Padé-type (ENO-Padé) scheme, J. Comput. Phys. 177, 37-58.