Internal Combustion Engines on MPP

Dr. Monika Wierse
Silicon Graphics Computer GmbH
Allmandring 30
D-70569 Stuttgart
mwierse@sgi.com

Harald Echtle
Daimler-Chrysler AG
HPC: E222
D-70546 Stuttgart, Germany
echtle@daimlerchrysler.com

Prof. Dr. A. David Gosman
Computational Dynamics Ltd.
200 Shepherds Bush Rd.,
London W6 7NY, England
d.gosman@cd.co.uk
Abstract:
The design of internal combustion engines still requires a lot of testing and many prototypes are required within a design cycle. This is a very cost and time consuming process, which has to be complemented and replaced partly by numerical simulations of the flow and the combustion. Using complex 3D simulation within the design cycle allows the evaluation of new concepts in an early stage of the development which allows a reduction of the number of prototypes that must be built and tested. On the other hand the joint usage of sophisticated testing and complex 3D flow and combustion simulation improves the understanding of the physical phenomena considerably, which is a prerequisite for the tuning of a basic design. Unfortunately in the past the numerical simulation of engines including grid generation usually took months and within a typical engine design cycle only a very small number of 3D simulations could be done. The EC Esprit project HPS-ICE (High Performance Simulation of Internal Combustion Engines, Esprit Contract 20184) was set up with the partners Computational Dynamic Ltd., Daimler Chrysler AG, SGI, HLRS and VOLVO. The objective of this collaborative work was to be able to apply 3D transient simulation of fluid flow, mixing and combustion in internal combustion engines to the design process. This required a reduction of the time to perform a typical simulation cycle from several months to one week (including pre-processing, simulation and post-processing).

Keywords:
CFD, combustion, grid generation, visualization, HPC, VR, parallel computing, simulation, engine simulation, spray, load balancing

Table of contents
1 Introduction
2 Sketch of the Entire Engine Simulation Process
3 Parallelization of the Numerical Simulation
4 Parallelization of the Postprocessing
5 Results
6 Conclusions
7 Acknowledgements
8 References

1 Introduction
In other areas of engineering design, numerical simulation already could replace experimental simulation entirely or at least partly. Examples of those areas in the automotive industry are for instance coolant jackets of internal combustion engines, under-hood flows and passenger compartments. For an increasing number of application fields numerical simulation is faster and cheaper than experiments and at least sufficiently accurate.

In the development process of internal combustion engines, computer simulations play only a minor role compared to experimental methods. The description of the physical process in a reciprocating piston engine including combustion and fuel injection requires sophisticated physical models, which can only be described by complex mathematical equations; this is in some areas still a research topic. Beside this the numerical simulation of those equations requires sometimes such a high grid density, that it is hard (if not impossible) to solve those problems with the computer power available today. In order to end up with a manageable computing model in some areas (e.g. combustion and mesh density) some compromises have to be accepted to keep the model as simple as possible. For instance turbulence and combustion modelling are used to keep the model sizes as small as possible. Clearly is has to be proven that these compromises are a good representation of the real process.
It is obvious that there is a huge demand for numerical simulation. First of all for the validation of the chosen models and second for the engineering design process. Here especially with parameter studies of valve and inlet shapes, changes in the geometry, changes of boundary conditions, optimization of fuel injectors and fuel flow rates. The simulation of the fuel-air mixing process in the intake port and in the cylinder is a vital prerequisite for the correct prediction of combustion engines. Charge motion is currently described in CFD engine simulation codes by the solution of compressible Navier-Stokes equations using for instance k-e turbulence models. In the numerical simulation code STAR-CD which is applied in this project a Lagrangian formulation of the fuel spray is used with droplet-air interaction, droplet wall interaction and wall film flow with appropriate sub-models. The Lagrangian particle tracking model required special attention for the parallel implementation since it is not mesh-based. Moving meshes and scalability of the numerical simulation up to many processors is another difficulty in this application.

In this paper we will shortly describe the entire process chain from grid generation, numerical simulation and post processing which has to be shortened to one week to make the numerical simulation acceptable. We will mainly focus on the parallelization aspects in this chain: numerical simulation and post processing.

2 Sketch of the Entire Engine Simulation Process

In the following figure the different steps of the entire engine simulation are depicted with the according names of the programs (in grey boxes) of the simulation software that have been used. STAR-HPC is the parallel version of the numerical simulation code STAR-CD [1] from Computational Dynamics (CD). The programs ProICE and ProSTAR are preprocessing tools from ADAPCO. The visualization package COVISE is developed at the University of Stuttgart and by VirCinity.

![Figure 1: Simulation Process Chain](image)

In the yellow boxes possible changes by the user are mentioned and an arrow indicates the necessary restart of the process chain.

2.1 Grid Generation

One major success in the HPS-ICE project was the time reduction of the grid generation process (the step from the CAD geometry to the surface mesh and the volume mesh including the information of the mesh movement). Before the HPS-ICE project started this process normally needed 2-3 weeks. Today this is possible in one day in a nearly automatic manner. To create an engine geometry with a complex shape of the piston top, valves and dome the program ProICE [2] was written which is based on a set of parameterized templates of typical engines. In the two following pictures the template is mapped to the CAD surface of the real cylinder.
The resulting unstructured mesh consists mainly of hexahedra, however prisms, pyramids and tetrahedra are also supported.

### 2.2 Moving Mesh Capabilities

To model the moving piston and valves vertices of the mesh could be moved accordingly. But this can only be done in a restricted manner since the mesh will be compressed/stretched heavily during the piston/valve motion. Since the aspect ratio of the cells has to stay in a certain range for accuracy reason of the numerical method cell layers normal to the piston/valve movement have to be deleted/added during mesh movement. By using arbitrary interfaces, even complex geometries (e.g. Diesel engine intake ports) can be meshed fairly rapidly. The description of the mesh motion in the valve region can be simplified by using arbitrary sliding interfaces in a similar way. As long as no local adaptation (nonstationary refinement and coarsening of the mesh) is used, the work to be done for the mesh movement can be performed by a separate program. To support all the complicated moving mesh capabilities mentioned above this is done in the serial program ProICE, which also can prepare the grid in advance. The deletion and addition of cell layers can be performed very fast, but the vertex movement can take a while, since a smoothing algorithm has to be used which tries to optimize the grid quality to avoid distorted cells (aspect ratio, warpage etc.). During the HPS-ICE project this coupling of the numerical simulation software program STAR-HPC running massively parallel and the serial ProICE was optimized such that the moving mesh preparation only marginally influences the simulation time. This allows speedups as in the stationary case, if the work of each processor is well balanced.

### 3 Parallelization of the Numerical Simulation

#### 3.1 Mathematical Method and Discretization

The implicit finite volume method which is used in STAR-HPC discretizes the three dimensional unsteady compressible Navier-Stokes equations describing the behaviour of mass, momentum and energy in space and time. All investigations done within this project were done with
- a k-ε turbulence model with a wall function to model the turbulent behaviour of the flow,
- a premixed version of the 2-equation Weller model [3] [4],
- two scalar transport equations to track the mixture of fresh and residual gas,
- a space conservation law to model the timedependent behaviour of the finite volumes (e. g. hexahedra). The fuel injection is modeled by a large number of droplet parcels formed by droplets of
different diameter. The number of parcels has to be large enough to represent the real spray in a statistical sense. An ordinary differential equation for every droplet trajectory has to be solved as function of the droplet and flow properties (mass, momentum, energy, drag, heat conduction). Each droplet is considered as a sphere and based on this geometric simplification droplet drag and vaporisation rates are evaluated. In addition collision and breakup models for droplet-droplet and droplet-wall interaction are used to describe the spray and its feedback on the flow as realistic as possible.

### 3.2 Domain Decomposition and Load Balancing

To get scalability of a parallel application for a high number of processors (8) it is necessary to equalize the load and restrict the memory address space locally for each processor. Therefore domain decomposition is applied where the grid is decomposed in different parts which are then assigned to the different processors. At the boundary of each processor domain an update of the data has to take place. In STAR-HPC the message passing library MPI is used to communicate between the processors. The choice of the domain decomposition is a difficult task in our application where
- the number of elements in the grid is changing due to the mesh movement,
- droplets are moving around in an unpredictable concentration,
- a huge linear system of equations has to be built up and solved,
- different boundary conditions and wall functions have to be evaluated.

#### 3.2.1 Scalability of a Non Moving Mesh Simulation

In the case of a non moving mesh application and with no droplet modelling a standard domain decomposition method can be applied. This domain decomposition method distributes the finite volumes equally over the processors. The aim is to minimize the amount of data to communicate between the processors to update values at the processor domain boundaries. That means the different domains has to be as compact as possible. The public domain software package METIS can be used for instance. The numerical simulation code STAR-HPC is scaling quite well for this type of application as we see in the following table. In this simulation the flow around an E-class car of Mercedes was simulated. The grid consists of 3.1 million cells. The compressible Navier-Stokes equations in 3D with a k-e model are solved. The calculations are performed on an SGI-Origin 2000 with 195 MHz R10000 processors. The times are elapsed time in seconds. The SGI Origin 2000 has 64 KB first-level and 4 MB second-level cache.

<table>
<thead>
<tr>
<th># of CPUs</th>
<th>Time [s]</th>
<th>Speedup</th>
<th>Par. Eff.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>11181</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>4</td>
<td>2668</td>
<td>4.2</td>
<td>1.05</td>
</tr>
<tr>
<td>8</td>
<td>1334</td>
<td>8.4</td>
<td>1.05</td>
</tr>
<tr>
<td>16</td>
<td>632</td>
<td>17.7</td>
<td>1.1</td>
</tr>
</tbody>
</table>

#### 3.2.2 Scalability of a Moving Mesh Simulation

In a moving mesh simulation with a changing number of cells due to deletion and addition the load on the processors can change heavily and has to be rebalanced. But it has to take into account that rebalancing can also be time consuming. In the HPS-ICE project some tests with the load balancer JOSTLE were performed [5]. JOSTLE is a parallel program that distributes cells with a diffusion method. However mostly an initial decomposition was used such that by the cell
deletion/addition the number of cells on the processors are changing equally. This is achieved by decomposing the engine cylinder and valve region in pie-slices and the inlet and outlet with a standard tool like METIS. Each processor gets one of those pie slices and one of the no moving parts. Although this decomposition is of course not optimal with respect to the minimization of the amount of data that has to be exchanged between the processors to update the data in the boundary of each processor domain, it was hard to beat it by using dynamic load-balancing. The following figure shows a decomposition for four processors.

![Figure 4: Decomposition for 4 processors](image)

In the next table we see first results of the moving mesh case: A transient in-cylinder cold flow in a typical four valve engine of Daimler-Chrysler. The grid, consisting of 300,000 cells, was decomposed for 2,4,8,16 and 32 processors. In the time frame considered 170,000 of the cells are activated, which is the reason for the decreasing parallel efficiency for a large number of CPUs. The simulations were done on an SGI-Origin 2000 (with 14 195 MHz R10000 CPUs with 4 GB main memory, 4 MB secondary cache), on a T3E-900 (512 DEC EV5 processors with 450 MHz clock speed, 128 MB memory per processor, 96 KB secondary cache) and on a T3E-1200 (740 DEC EV5 processors with 600 MHz clock speed, 256 MB memory per processor, 96 KB secondary cache).

<table>
<thead>
<tr>
<th>SGI Origin 2000</th>
<th>Cray T3E-900</th>
</tr>
</thead>
<tbody>
<tr>
<td># of CPUs</td>
<td>Time [s]</td>
</tr>
<tr>
<td>2</td>
<td>5156.40</td>
</tr>
<tr>
<td>4</td>
<td>2617.00</td>
</tr>
<tr>
<td>8</td>
<td>1378.28</td>
</tr>
<tr>
<td>12</td>
<td>1072.68</td>
</tr>
<tr>
<td>16</td>
<td>833.12</td>
</tr>
<tr>
<td>32</td>
<td>742.98</td>
</tr>
</tbody>
</table>

<sup>1</sup> relative to 2 Processors

<table>
<thead>
<tr>
<th>Cray T3E-1200</th>
</tr>
</thead>
<tbody>
<tr>
<td># of CPUs</td>
</tr>
<tr>
<td>8</td>
</tr>
<tr>
<td>16</td>
</tr>
<tr>
<td>32</td>
</tr>
</tbody>
</table>

<sup>3</sup> relative to 8 Processors

The numbers in the following table are for a Daimler-Chrysler test case with 600,000 cells; combustion is included in the simulation. In the time frame considered between 260,000 and
300,000 cells are activated during the simulation.

<table>
<thead>
<tr>
<th># of CPUs</th>
<th>Cray T3E-900</th>
<th># CPUs</th>
<th>Time [s]</th>
<th>Speedup (^1)</th>
<th>Par. Eff.</th>
</tr>
</thead>
<tbody>
<tr>
<td>64</td>
<td>8063</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>96</td>
<td>6376</td>
<td>1.46</td>
<td>0.84</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\(^1\) relative to 64 Processors

All moving-grid cases were calculated with a static decomposition and without dynamic load-balancing. Not all the cases could be run on the T3Es due to memory limits on each processor. In the following figure the waiting time on ProICE is depicted at different time steps. The huge peaks come from the time steps in which the smoother is activated to optimize the moving grid. This was the situation at the beginning of the project. It is clear that this waiting time would reduce the scalability of the parallel simulation. In the latest version of ProICE the situation has improved, because those time steps can be considered in advance during the simulation process. An entire simulation for 400 degree crank angle with combustion can be performed for this test case in less than two days on 64 CPUs (compared to 120 days on a single processor workstation).

### 3.2.3 Spray Simulation and Load Balancing

In the next table we see results of an engine simulation including spray calculation. In this example every time step 10 particles are injected into an engine geometry consisting out of 176494 elements. The calculation lasts for 1000 time steps and the grid topology is not changing in time. The SGI Origin 2000 used for this benchmark had a 300 MHz R12000 processor with 8 MB second-level cache.
The distribution of the particles is very unequal for the different processors. Into some processors never a particle enters and the gas phase calculation can only continue after the spray calculation has finished for all processors. But nevertheless speedups can be obtained without rebalancing the workload of the processors since the tracking of particles is only 10-20% of the overall computing time. It has to be considered to use a multiconstrained loadbalancer as it is developed in PARMETIS. But again the effort to determine the way of redistributing the work and the rebalance of the data over the processors has to be taken into account.

4 Parallelization of the Postprocessing

4.1 Challenge

The data of the numerical simulation has to be analysed by an engineer to understand the flow and find possible improvements. There are two major difficulties in the visualization of data sets of the application described above:

- the flow inside the engine is a highly complex three-dimensional transient data set. Cutting surfaces and isosurfaces are standard methods to visualize data in a twodimensional way. The timedependency of the data requires a fast determination of the cutting planes and isosurfaces such that the user can get a good impression of the evolution of the data.
- the amount of data to visualize is too much to be handled by a standard graphics workstation. For the time-dependency, at least 100 time-steps have to be handled in memory to get a visual impression of the flow behavior interactively. For particle tracing even higher time resolutions are required.

4.2 The visualization software package COVISE

In the HPS-ICE project the collaborative visualization system COVISE [6] *(Collaborative visualization and simulation environment)* was extended such that the requirements mentioned above are fulfilled: To speed up the determination of cutting planes and isosurfaces the domain decomposition of the geometry and the calculated data of the simulation can now be used to achieve parallelization in the visualization process. These modules of COVISE have to run directly on the supercomputer that keeps the geometry and the simulation data. The results of these modules are sets of triangles which represent the cutting planes or isosurfaces. This information can then be passed to the visualization workstation to render it. But before sending this data it can be reduced by 'data reduction' methods. The aim of these tools is to reduce the amount of data that is necessary to represent a surface. Those methods are available for instance in SGI's OpenGL Optimizer [7] and work on geometrical but also on simulation data. In the HPS-ICE project those methods were ported to the T3E. In this project the data reduction algorithm is applied to different pieces of the surfaces in parallel. These pieces are produced by the corresponding parallel visualization module. A decreased number of triangles helps to lower network congestion between the supercomputer and the workstation and also increases the frame rates of the rendering hardware.

In the following figure we see an example of data reduction for an isosurface in a data set of Daimler-Chrysler. The number of triangles was reduced from 17.000 to 5.000 in this case.
6 Results

In the following figures some typical simulation results are shown. In Fig. 8-10 are shown the temporal variation of the grid and the turbulent kinetic energy at crank angles 332 when the intake valves are still closed, 390 when the exhaust valve is closed and the intake valves are opened and 540 when the piston has reached its bottom dead center position. The same color scale is used in all the figures and it can be seen how the intensity of the turbulent kinetic energy is decaying, from the beginning to the end of the suction phase (360 - 540).
Figure 8: Crank angle = 332°
Figure 9: Crank angle = 390 °
A contour plot of the fuel concentration at various crank angles is given in Fig. 11 and the corresponding velocity field is shown in Fig. 12. A comparison of the predicted in cylinder pressure for homogeneous combustion and the measurement is shown in Fig. 13. The agreement in the fine mesh (300,000 cells at BDC) is already acceptable.
Figure 11: Fuel concentration at various crank angles

Figure 12: Velocity field at various crank angles
Figure 13: Comparison numerical simulation and experiment

Figure 14 shows the propagation of a fuel spray in a port injected gasoline engine. The spray is injected at the left with a nozzle in the intake port and it propagates through the slit between intake valve and cylinder head into the cylinder. At the beginning the droplet velocity (magnitude and direction) and the droplet diameter are prescribed in a way that further downstream these droplets can vaporize or interact with each other, the wall and the flow. This leads to a change in diameter and temperature during their way down to the cylinder. Mainly in the valve region the averaged droplet diameter is changing rapidly for the droplets are splashed when they hit the valve surface. In addition the hot valve surface leads to an increased vaporisation rate of the droplets which additionally causes a decrease of the droplet diameter.

Figure 14: Spray injection
7 Conclusions

The HPS-ICE project is considered as highly successful by all partners. At the end of the project the major achievements of this project with respect to mesh generation, simulation and visualization can be summarised as follows:

- the mesh generation effort could be reduced from 4 weeks to 2-4 days by automatic the process as far as possible,
- the numerical simulation with STAR-HPC turns out to be highly scalable up to 100 processors such that the engine simulation time could be reduced from months/weeks to a few days,
- the visualization with the parallel version of COVISE allows an efficient analysis of transient engine simulation results.

Due to these improvements the 3D engine simulation is now integrated into the design cycle.

8 Acknowledgements

The HPS-ICE project was funded by the European Commission in the ESPRIT program. The authors would like to thank all project partners for the collaboration inside the project. We also would like to thank Dr. Herbert Ohno of SGI who supported all the special requests for the T3E at HWW from the HPS-ICE project.

9 References

Star-HPC, ProICE


JOSTLE


COVISE


OpenGL Optimizer

