Fluid-structure interaction simulation with the Virtual Test Facility

Ralf Deiterding

ORNL & Caltech

deiterdingr@ornl.gov

May 21, 2007

Outline

- Introduction
  - *Ideas, software*
- Fluid solver framework
  - *Adaptive mesh refinement, boundary embedding*
- Fluid-structure coupling
  - *Algorithmic concept, main components*
- Examples from different computational solid dynamics solvers
- Simple 2d template application
  - *Main codes*
  - *Demonstration*

The Virtual Test Facility

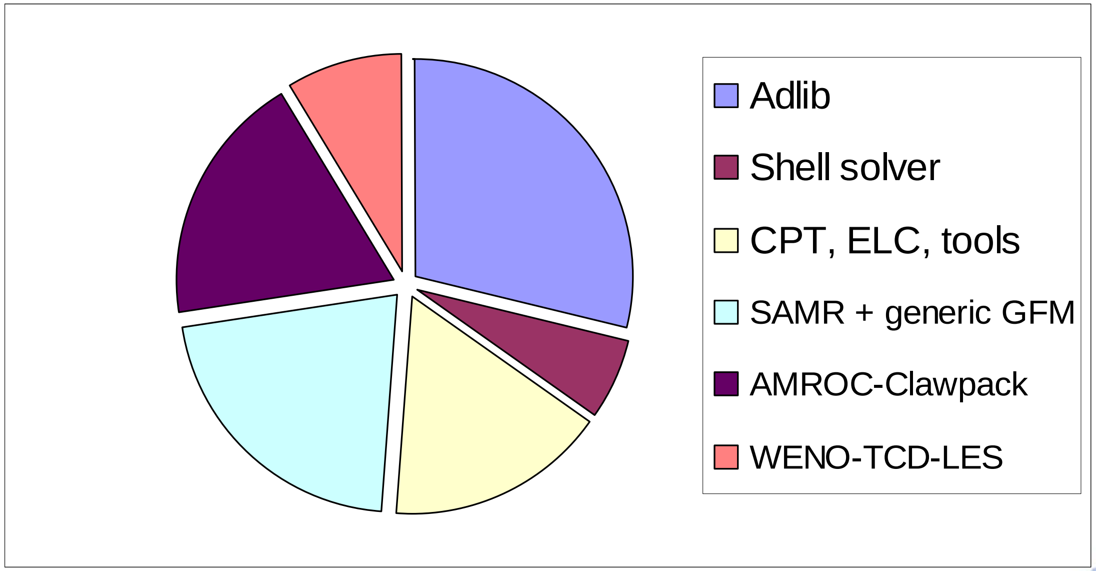
Infrastructure for fluid-structure interaction simulation of shock- and detonation-driven solid material deformation

Concept:

- Use a level-set-based approach to couple Lagrangian solid mechanics solvers to Eulerian fluid mechanics solvers
- Level set stores the distance to closest point on solid body surface in each Eulerian mesh point
- Distance information is updated on-the-fly as the solid evolves
- Use distance information to consider geometrically complex boundary conditions in a ghost fluid method for Cartesian fluid solvers
- Use block-structured mesh adaptation to mitigate boundary approximation errors
- Eulerian-Lagrangian inter-solver communication library synchronizes the boundary data exchange between coupled solver modules
- Implement all components for distributed memory systems with non-blocking MPI communication routines

The Virtual Test Facility software

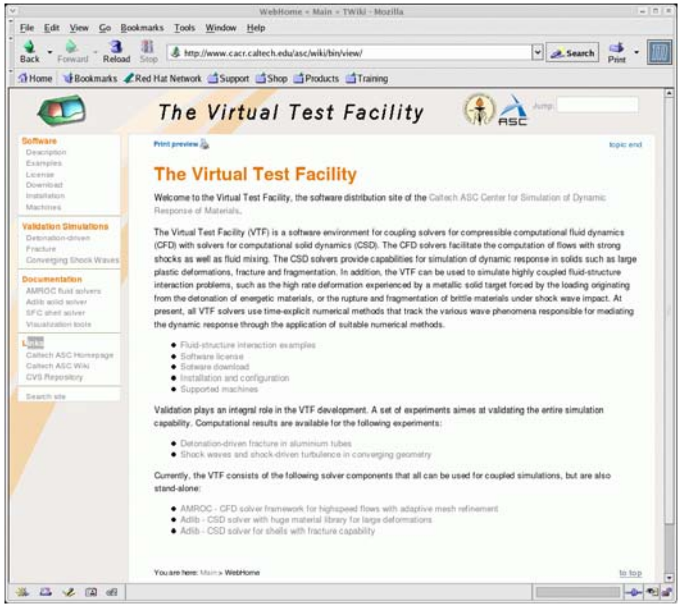
- Freely available infrastructure for fully coupled fluid-structure simulations in 3D and 2D on distributed memory machines
  - *Shock-capturing Cartesian finite volume methods for Euler and Navier-Stokes equations (WENO-TCD-LES, AMROC-Clawpack) with SAMR*
  - *Lagrangian finite elements methods (Adlib, SFC shell solver) with standard material models or rigid body motion for solid simulation*
  - *Implicit boundary representation with level-set functions (e.g. through CPT), considera-tion in Cartesian scheme with generic ghost-fluid method (GFM) fully incorporated into SAMR algorithm*
- Fully coupled fluid-structure simulations in 3D on distributed memory machines
  - *Eulerian-Lagrangian coupling (ELC) module*
- Language: object-oriented C++ with components in C, F77, F90. Size ~12MB
- ~430,000 lines of source code (ANSI)
- autoconf/ automake environment with full support for all ASC platforms



Current portion of source code size of VTF subcomponents

The Virtual Test Facility software

- Freely available infrastructure for fully coupled fluid-structure simulations in 3D and 2D on distributed memory machines
  - *Shock-capturing Cartesian finite volume methods for Euler and Navier-Stokes equations (WENO-TCD-LES, AMROC-Clawpack) with SAMR*
  - *Lagrangian finite elements methods (Adlib, SFC shell solver) with standard material models or rigid body motion for solid simulation*
  - *Implicit boundary representation with level-set functions (e.g. through CPT), considera-tion in Cartesian scheme with generic ghost-fluid method (GFM) fully incorporated into SAMR algorithm*
- Fully coupled fluid-structure simulations in 3D on distributed memory machines
  - *Eulerian-Lagrangian coupling (ELC) module*
- Use an online content management system to create the documentation necessary for the release of the VTF software
  - *Installation, configuration, examples*
  - *Scientific and technical papers*
  - *Archival of key simulation and experimental results*
- http://www.cacr.caltech.edu/asc



The Virtual Test Facility

Welcome to the Virtual Test Facility, the software distribution site of the Caltech ASC Center for Simulation of Dynamic Response of Materials.

The Virtual Test Facility (VTF) is a software environment for coupling solvers for compressible computational fluid dynamics (CFD) with solvers for computational solid dynamics (CSD). The CFD solvers facilitate the computation of flows with strong shocks as well as fluid mixing. The CSD solvers provide capabilities for simulation of dynamic response in solids such as large plastic deformations, fracture and fragmentation. In addition, the VTF can be used to simulate highly coupled fluid-structure interaction problems, such as the high rate deformation experienced by a metallic solid target forced by the loading originating from the detonation of energetic materials, or the rupture and fragmentation of brittle materials under shock wave impact. At present, all VTF solvers use time-explicit numerical methods that track the various wave phenomena responsible for mediating the dynamic response through the application of suitable numerical methods.

- Fluid-structure interaction examples
- Software license
- Sotware download
- Installation and configuration
- Supported machines

Validation plays an integral role in the VTF development. A set of experiments aimes at validating the entire simulation capability. Computational results are available for the following experiments:

- Detonation-driven fracture in aluminium tubes
- Shock waves and shock-driven turbulence in converging geometry

Currently, the VTF consists of the following solver components that all can be used for coupled simulations, but are also stand-alone:

- AMROC - CFD solver framework for highspeed flows with adaptive mesh refinement
- Adlib - CSD solver with huge material library for large deformations
- Adlib - CSD solver for shells with fracture capability

You are here: Main > WebHome

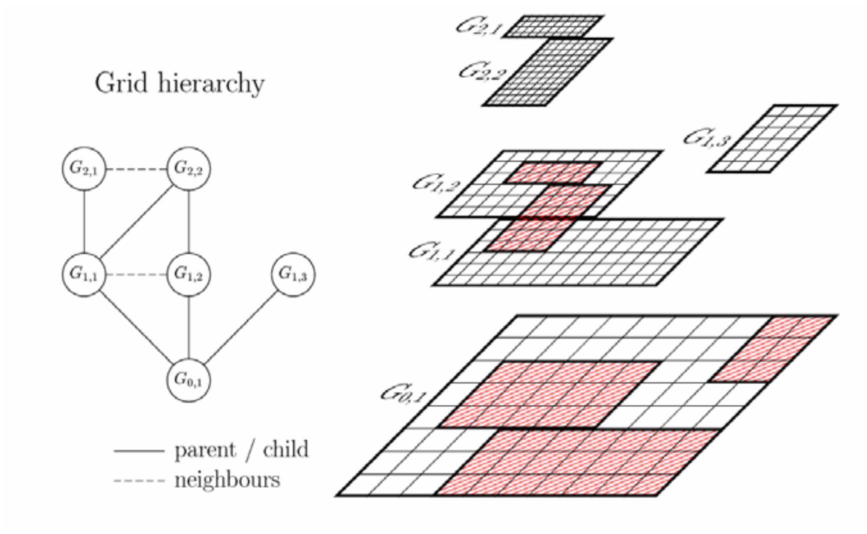
to top

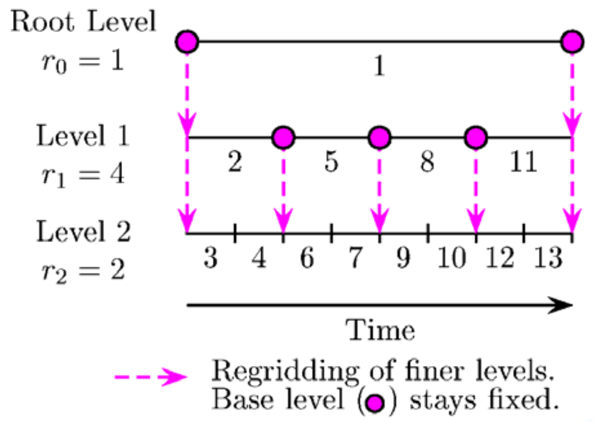
Structured AMR (Berger-Colella-type) for hyperbolic problems

- For simplicity

$$\partial_t \mathbf{q} + \nabla \cdot \mathbf{f}(\mathbf{q}) = 0$$

- Refined subgrids overlay coarser ones
- Computational decoupling of subgrids by using ghost cells
- Refinement in space *and* time
- Block-based data structures
- Cells without mark are refined
- Cluster-algorithm necessary
- Efficient cache-reuse / vectorization possible
- Explicit finite volume scheme



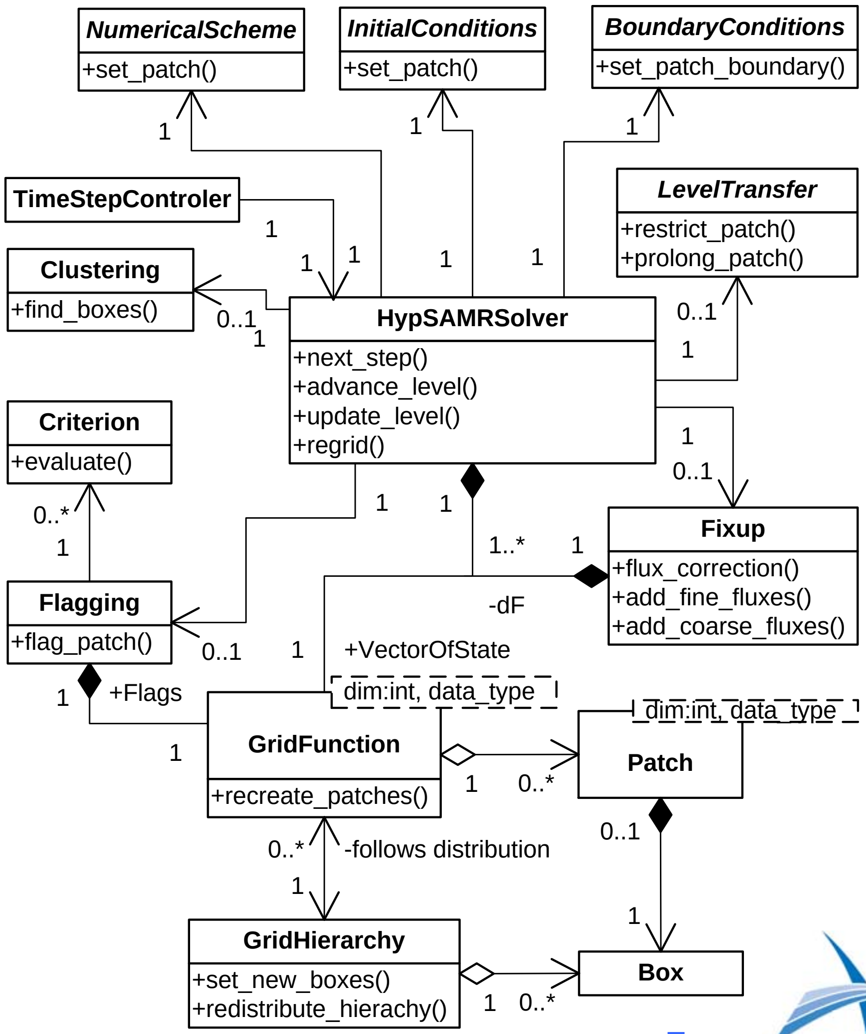


$$\mathbf{Q}_{jk}^{n+1} = \mathbf{Q}_{jk}^{n} - \frac{\Delta t}{\Delta x_1}\left[\mathbf{F}^1_{j+\frac{1}{2},k} - \mathbf{F}^1_{j-\frac{1}{2},k}\right] - \frac{\Delta t}{\Delta x_2}\left[\mathbf{F}^2_{j,k+\frac{1}{2}} - \mathbf{F}^2_{j,k-\frac{1}{2}}\right]$$

only for single rectangular grid necessary

UML design of Amroc

- Classical framework approach with generic main program in C++
- Customization / modification in Problem.h include file by derivation from base classes and redefining virtual interface functions
- Predefined, scheme-specific classes (with F77 interfaces) provided for standard simulations
- Standard simulations require only linking to F77 functions for initial and boundary conditions, source terms. No C++ knowledge required.
- User interface mimics Clawpack by R.J. LeVeque
- Expert usage (algorithm modification, advanced output, etc.) in C++

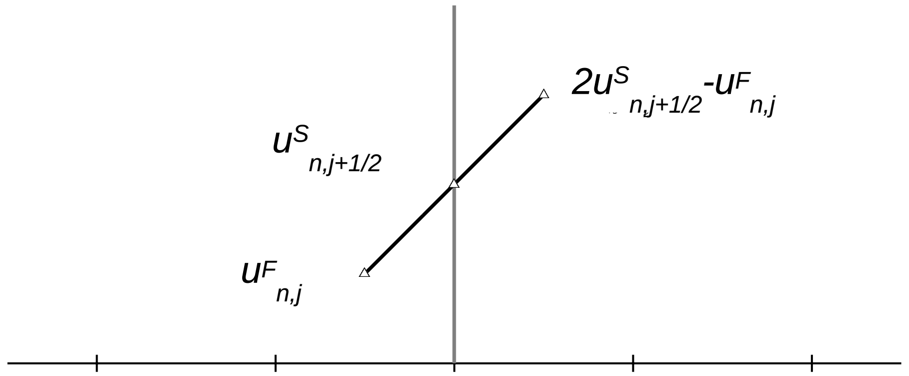


Available finite volume fluid solvers for Amroc

- Extended Clawpack with for full and one-step chemistry in Fortran 77 (R.Deiterding)
  - *Euler equations with various equations of state, mixtures of thermally perfect gases, stiffened gas equation of state for shocked liquids*
  - *Riemann solvers and flux vector splitting schemes with positivity preservation*
  - *Reference simulations and coupled simulations, especially with detonation modeling*
- WENO-TCD scheme with optional LES and chemical reaction capability in Fortran 90 (D.Hill, C.Pantano)
  - *Favre-averaged Navier-Stokes equations*
  - *Compressible turbulence LES model by D. Pullin*
  - *Used for turbulence simulations*
- Riemann Invariant Manifold Method (Euler equations for polytropic gas) by T. Lappas
- Riemann solver for gas-dynamics with chemistry in C++ (P.Hung)
- Ideal MHD solver by M. Torrilhon (only uniform for now)

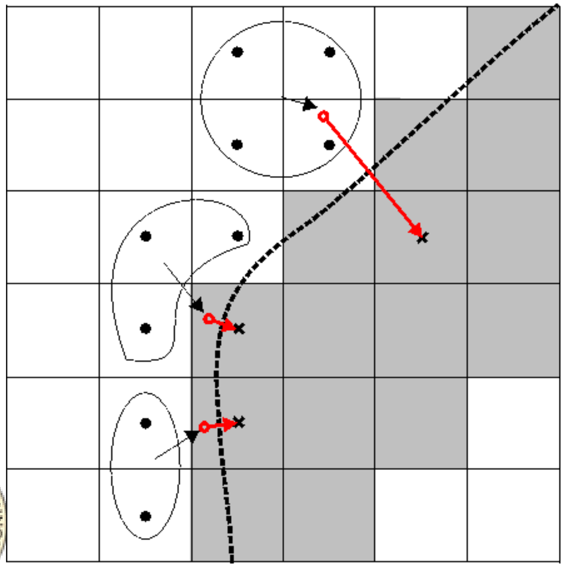
Ghost fluid method

- Incorporate complex moving boundary/ interfaces into a Cartesian solver (extension of work by R.Fedkiw and T.Aslam)
- Implicit boundary representation via distance function φ, normal $n=\nabla\varphi / |\nabla\varphi|$
- Treat an interface as a moving rigid wall
- Method diffuses boundary and is therefore not conservative
- Construction of values in embedded boundary cells by interpolation / extrapolation



$\rho^F_{n,j-1}$	$\rho^F_{n,j}$	$\rho^F_{n,j}$	$\rho^F_{n,j-1}$
$u^F_{n,j-1}$	$u^F_{n,j}$	$2u^S_{n,j+1/2}-u^F_{n,j}$	$2u^S_{n,j+1/2}-u^F_{n,j-1}$
$u^F_{t,j-1}$	$u^F_{t,j}$	$u^F_{t,j}$	$u^F_{t,j-1}$
$p^F_{n,j-1}$	$p^F_{n,j}$	$p^F_{n,j}$	$p^F_{n,j-1}$

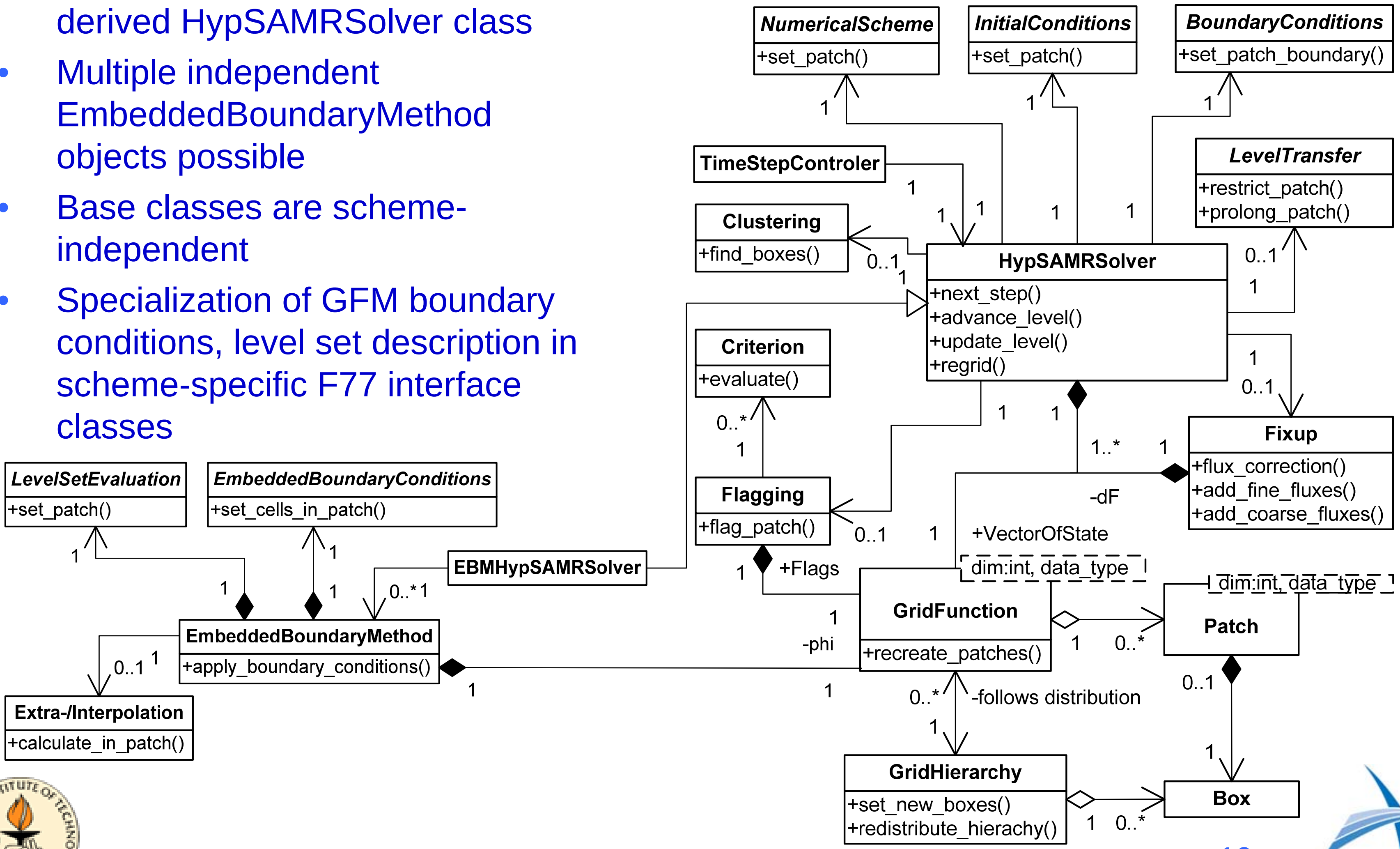
Velocity: $\mathbf{u}^F_{Gh}=2((\mathbf{u}^S-\mathbf{u}^F_M)\cdot\mathbf{n})\,\mathbf{n}+\mathbf{u}^F_M$



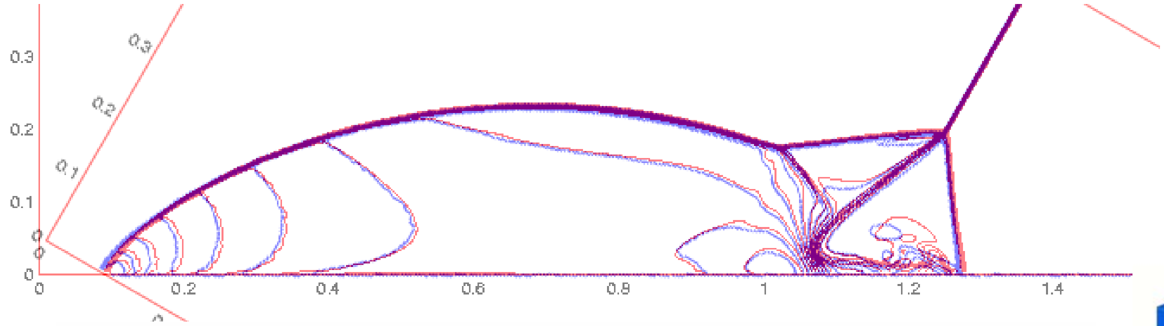
- Higher resolution at embedded boundary required than with first-order unstructured scheme
- Problems sensitive to boundary interaction require thorough convergence studies
- Appropriate level-set-based refinement criteria are available to cure deficiencies

Ghost fluid method in Amroc

- Core algorithm implemented in derived HypSAMRSolver class
- Multiple independent EmbeddedBoundaryMethod objects possible
- Base classes are scheme-independent
- Specialization of GFM boundary conditions, level set description in scheme-specific F77 interface classes

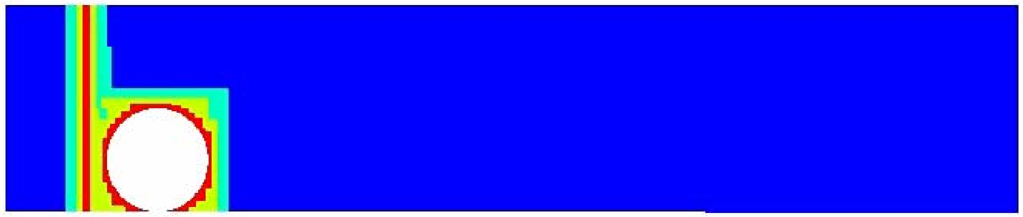


Verification of GFM



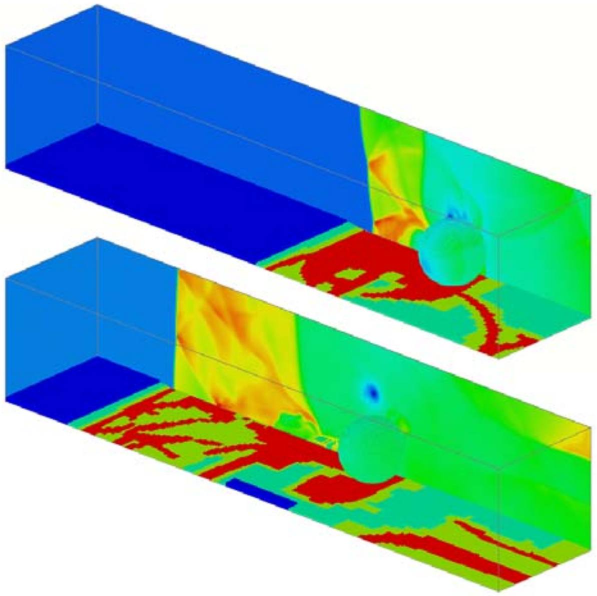
Overlay of two simulation of a Mach reflection on 800x400 grids with GFM (shown rotated) and 2nd order accurate scheme (initial conditions rotated)

Schlieren plot of density



3 additional refinement levels

Lift-up of a solid body in 2D and 3D when being hit by Mach 3 shock wave, Falcovitz et al. (1997)

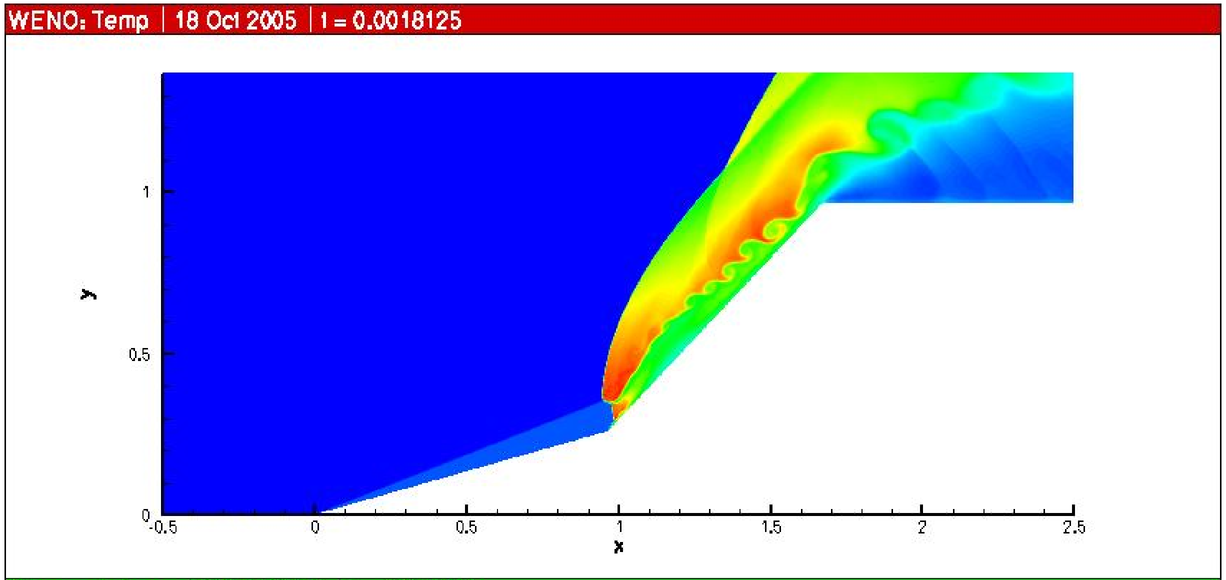


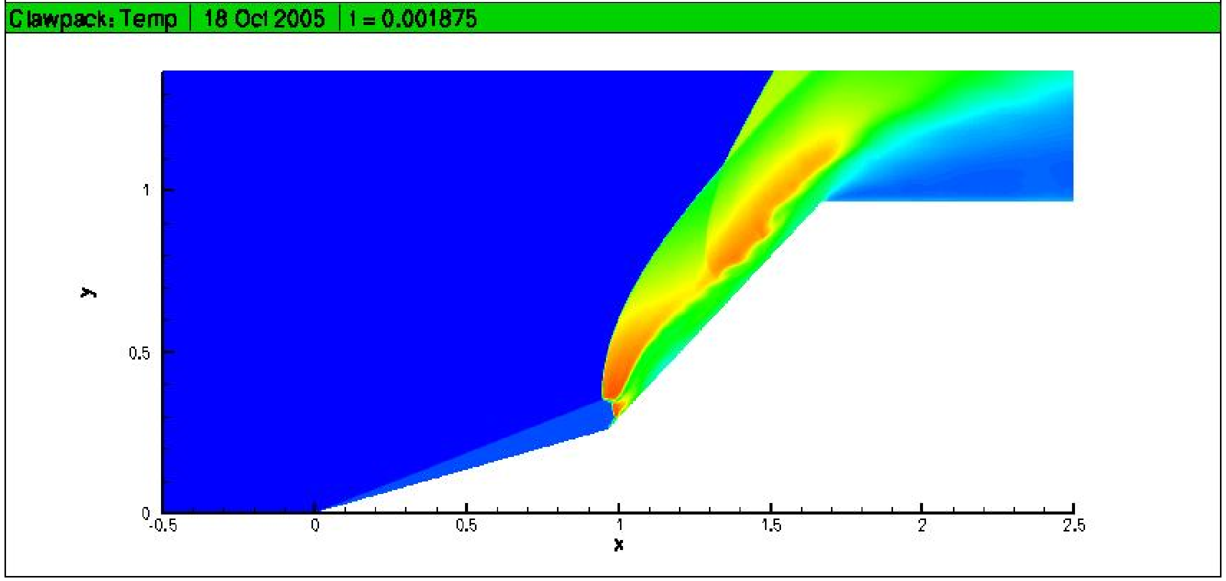
Extension to 3D, color plot of density

- 640h CPU on Pentium-4 2.2GHz
- AMR base grid 150x30x30, 3 additional levels all with factor 2

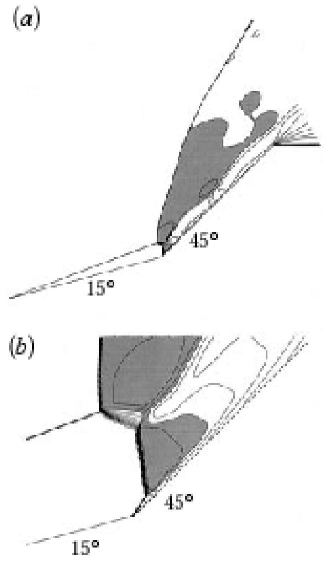
GFM verification: shock interaction at double-wedge geometry

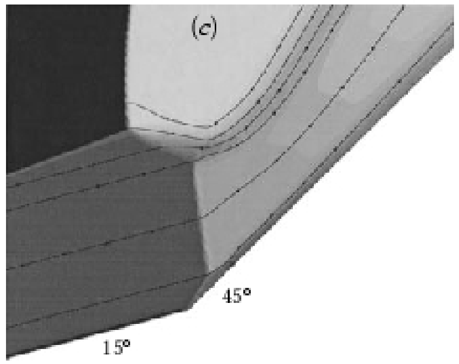
- Simulation by D. Hill
- Mach 9 flow in air hitting a double-wedge (15º and 45º)
- Example from Olejniczak, Wright and Candler (JFM 1997)
- AMR base mesh 300x100, 3 additional levels with factor 2
- 3rd order WENO computation vs. 2nd order MUSCL with van Leer flux vector splitting

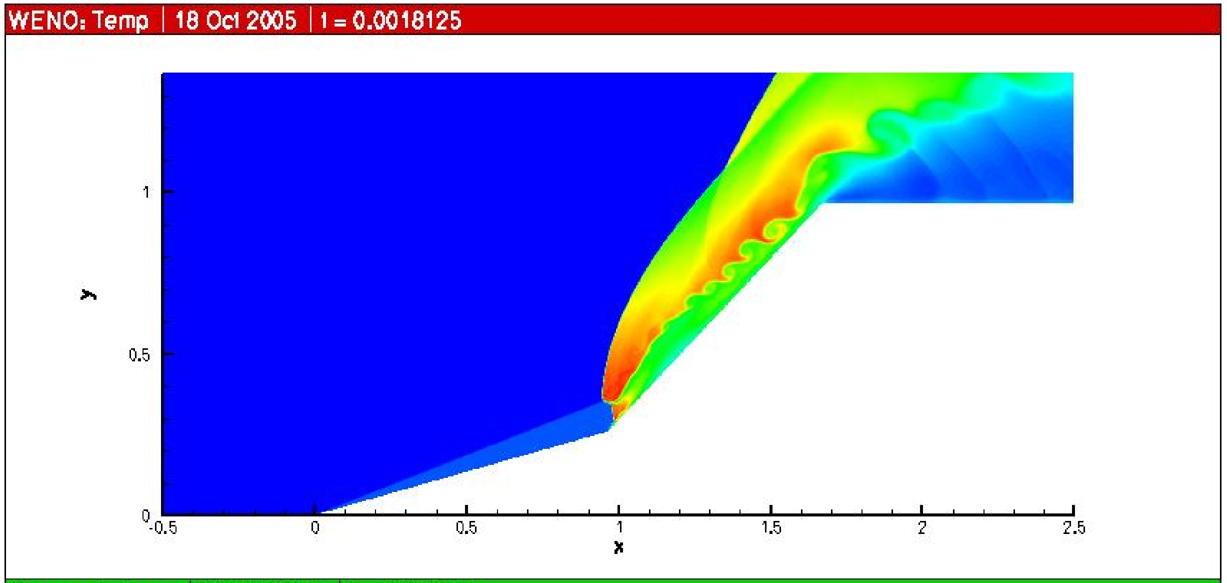


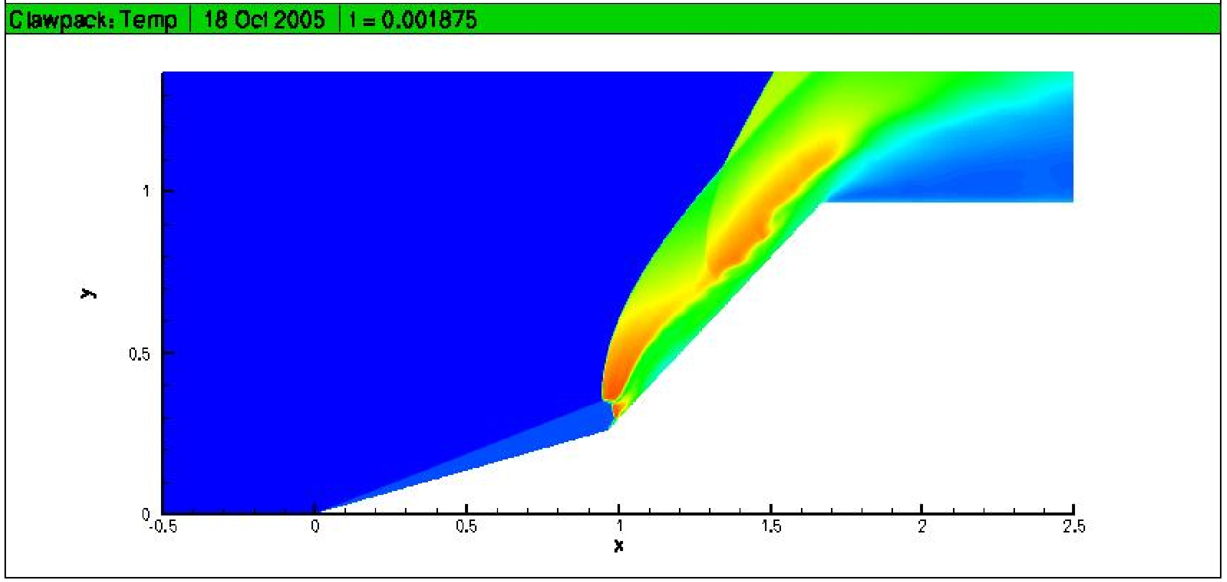


GFM verification: shock interaction at double-wedge geometry









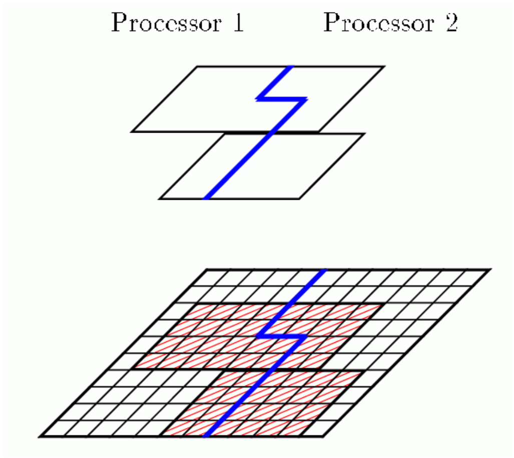
Parallelization strategy

Domain decomposition: $G_0 = \bigcup_{p=1}^{P} G_0^p$ with $G_0^p \cap G_0^q = \emptyset$ for $p \neq q$

$$G_0^p := \bigcup_{m=1}^{M_0^p} G_{0,m}^p \qquad \longrightarrow \qquad G_l^p := G_l \cap G_0^p$$

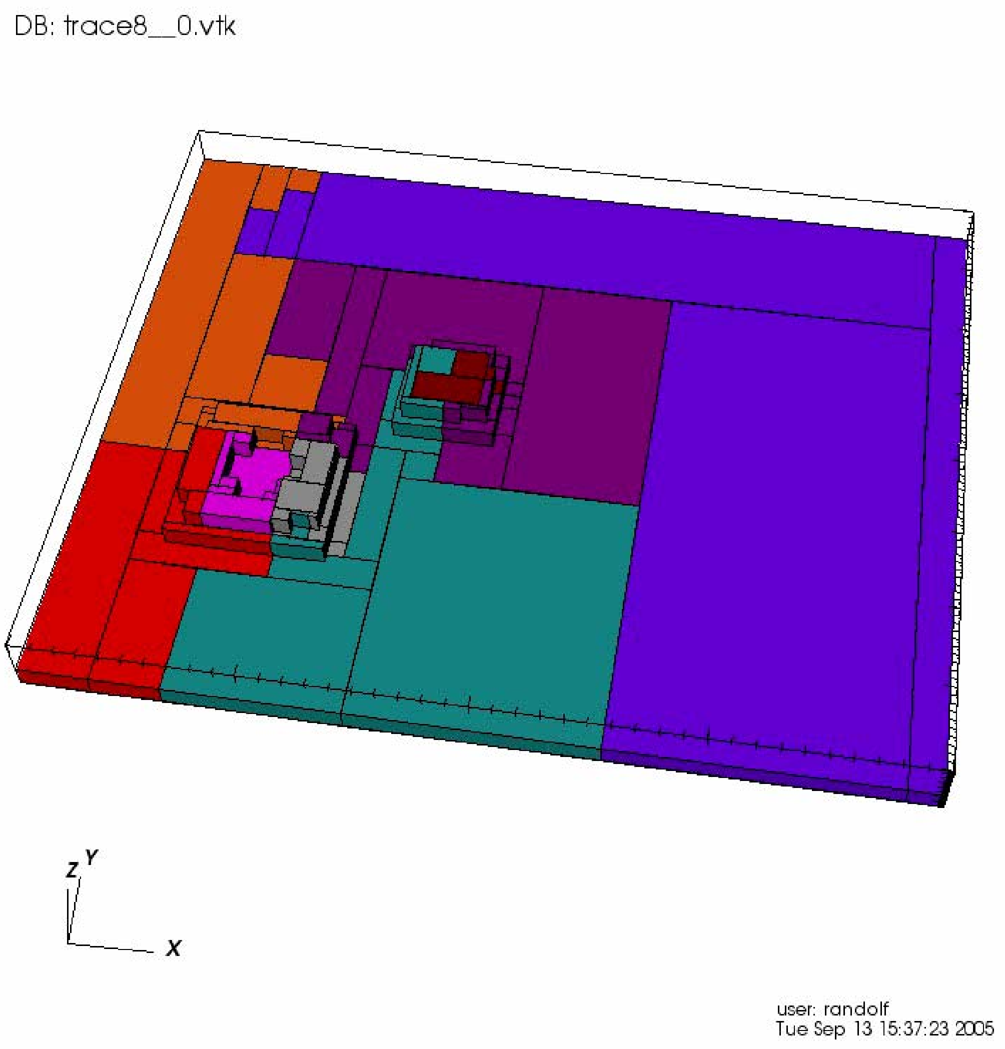
Workload: $\mathcal{W}(\Omega) = \sum_{l=0}^{l_{\max}} \left[\mathcal{N}_l(G_l \cap \Omega) \prod_{\kappa=0}^{l} r_\kappa \right]$, $\mathcal{N}_l(G)$ No. of cells on l

Load-balacing: $\mathcal{L}^p := \dfrac{P \cdot \mathcal{W}(G_0^p)}{\mathcal{W}(G_0)} \approx 1$ for all $p = 1, \dots, P$



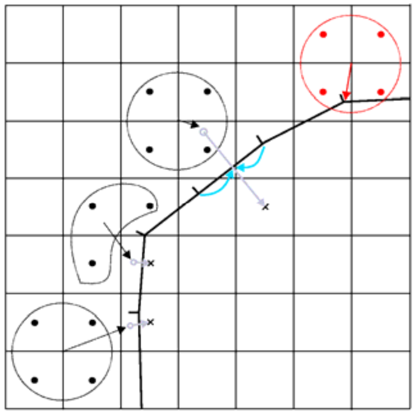
- Data of all levels resides on same node $\rightarrow$ Interpolation and averaging remain strictly local
- Only parallel operations to be considered:
  - *Parallel synchronization as part of ghost cell setting*
  - *Load-balanced repartitioning of data blocks as part of* **`Regrid(l)`**
  - *Application of flux correction terms on coarse-grid cells*
- Partitioning at root level with generalized Hilbert space-filling curve by M. Parashar

Parallelization strategy



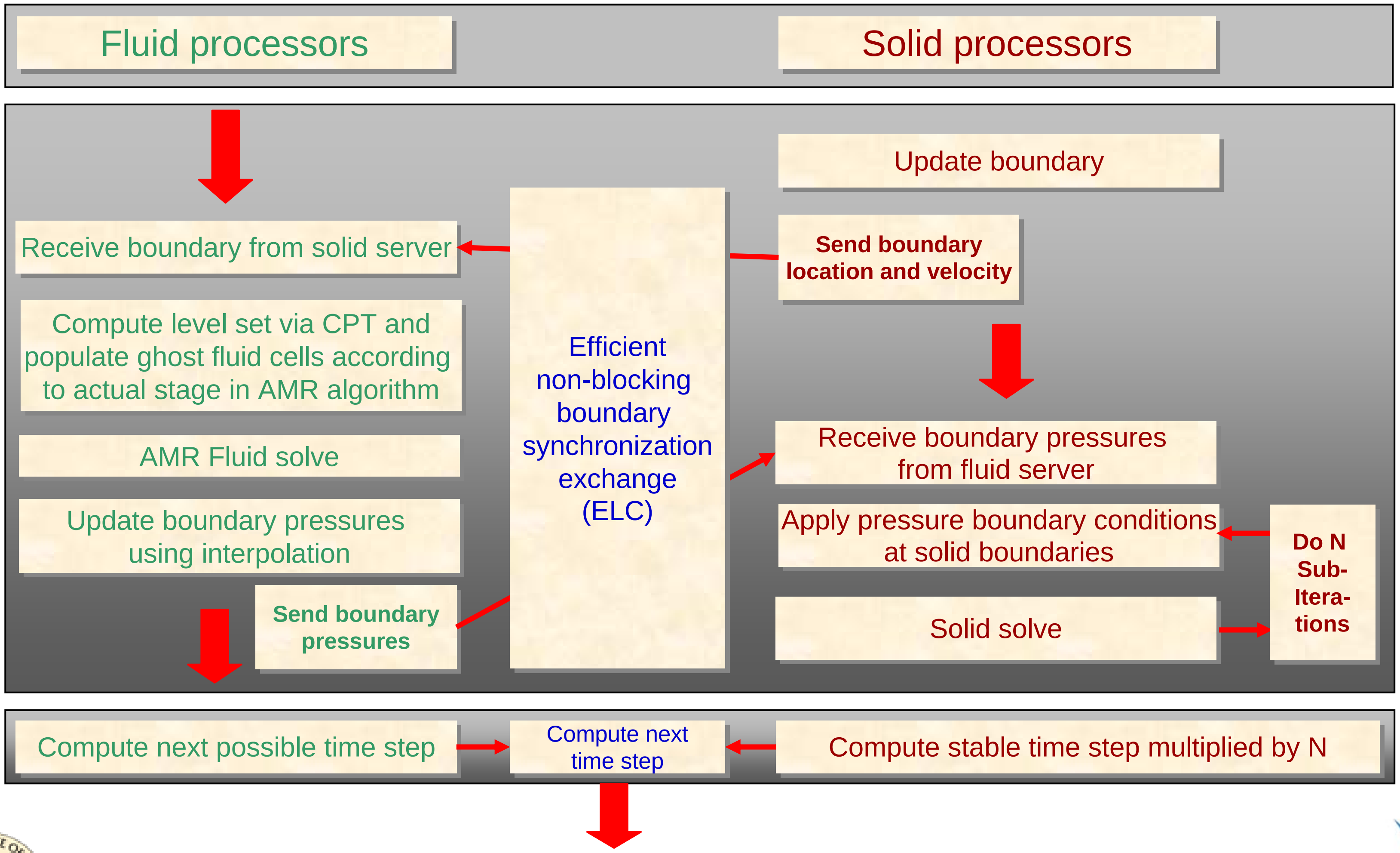
Fluid-structure coupling

- Couple compressible Euler equations to Lagrangian structure mechanics
- Compatibility conditions between inviscid fluid and solid at a slip interface
  - *Continuity of normal velocity: $u^S_n = u^F_n$*
  - *Continuity of normal stresses: $\sigma^S_{nn} = -p^F$*
  - *No shear stresses: $\sigma^S_{n\tau} = \sigma^S_{n\omega} = 0$*



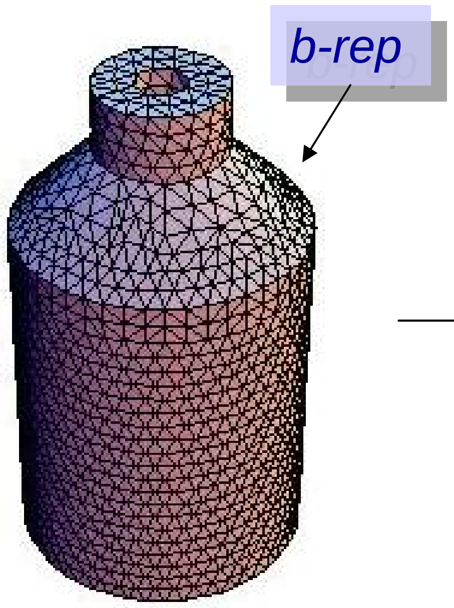
- Time-splitting approach for coupling
  - *Fluid:*
    - *Treat evolving solid surface with moving wall boundary conditions in fluid*
    - *Use solid surface mesh to calculate fluid level set*
    - *Use nearest velocity values $\mathbf{u}^S$ on surface facets to impose u^F_n in fluid*
  - *Solid:*
    - *Use interpolated hydro-pressure p^F to prescribe σ^S_{nn} on boundary facets*
- Ad-hoc separation in dedicated fluid and solid processors

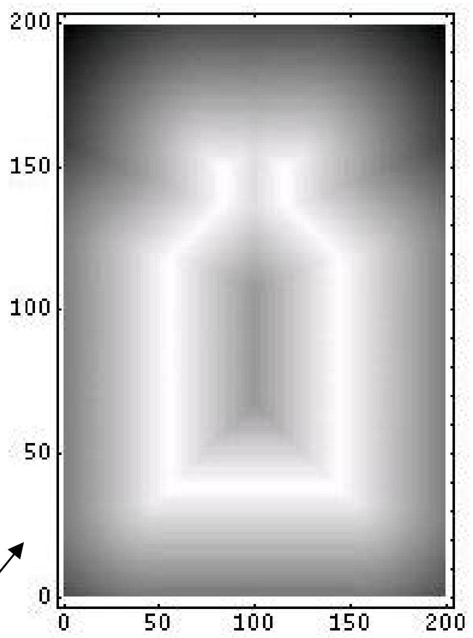
Algorithmic approach for coupling

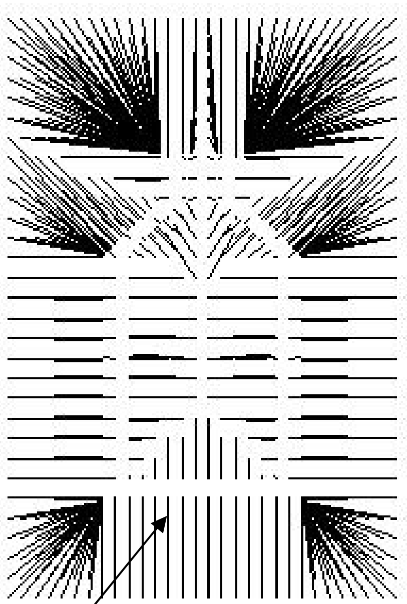

Fluid processors
Solid processors
Update boundary
Receive boundary from solid server
Send boundary location and velocity
Compute level set via CPT and populate ghost fluid cells according to actual stage in AMR algorithm
Efficient non-blocking boundary synchronization exchange (ELC)
AMR Fluid solve
Receive boundary pressures from fluid server
Update boundary pressures using interpolation
Apply pressure boundary conditions at solid boundaries
Do N Sub-Itera-tions
Send boundary pressures
Solid solve
Compute next possible time step
Compute next time step
Compute stable time step multiplied by N

Implicit representations of complex surfaces

- FEM Solid Solver
  - *Explicit representation of the solid boundary, b-rep*
  - *Triangular faceted surface.*
- Cartesian FV Solver
  - *Implicit level set representation.*
  - *need closest point on the surface at each grid point..*







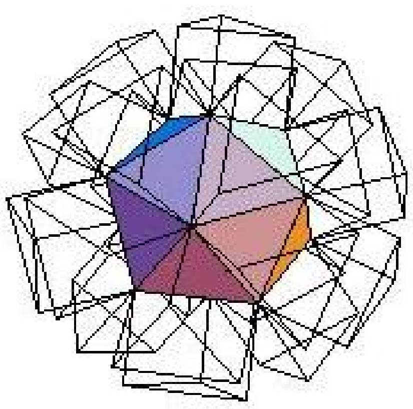
slice of distance

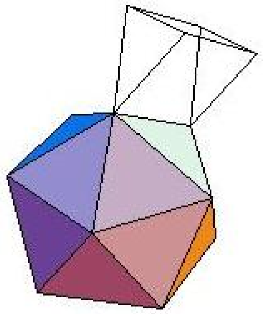
slice of closest point

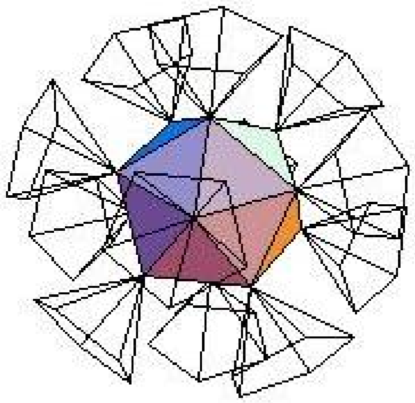
→ Closest point transform algorithm (CPT) by S. Mauch

CPT in linear time

- Problem reduction by evaluation only within specified max. distance
- The characteristic / scan conversion algorithm.
  - *For each face/edge/vertex.*
    - *Scan convert the polyhedron.*
    - *Find distance, closest point to that primitive for the scan converted points.*
- Computational complexity.
  - *O(m) to build the b-rep and the polyhedra.*
  - *O(n) to scan convert the polyhedra and compute the distance, etc.*

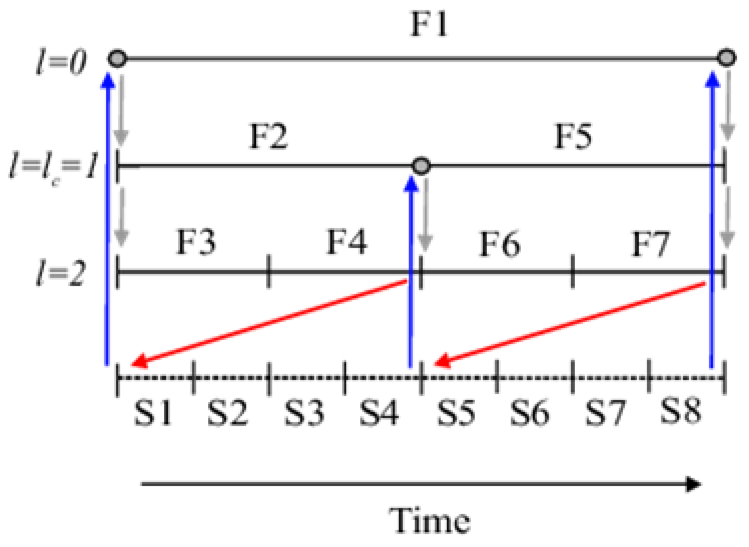

Face Polyhedra


Edge Polyhedra


Vertex Polyhedra

Incorporation into hierarchical SAMR

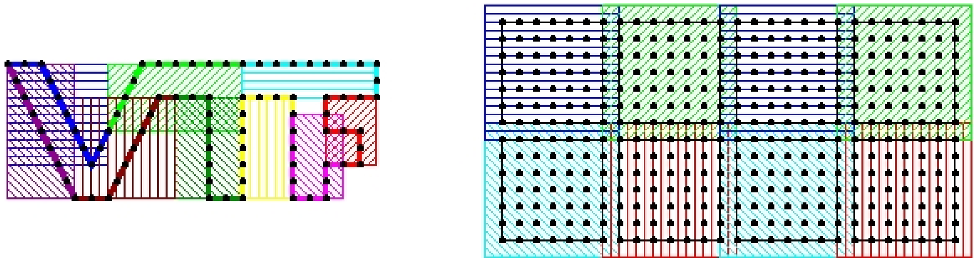
- Eulerian SAMR + non-adaptive Lagrangian FEM scheme
- Exploit SAMR time step refinement for effective coupling to solid solver
  - *Lagrangian simulation is called only at level $l_c < l_{max}$*
  - *SAMR refines solid boundary at least at level l_c*
  - *One additional level reserved to resolve ambiguities in GFM (e.g. thin structures)*
- Nevertheless: Inserting sub-steps accommodates for time step reduction from the solid solver within an SAMR cycle
- Communication strategy
  - *Updated boundary info from solid solver must be received (blue arrow) before regridding operation (gray dots and arrows)*
  - *Boundary data is sent to solid (red arrow) when highest level available*
- Inter-solver communication (point-to-point or globally) managed on the fly by current SAMR partition bounding box information by Eulerian-Lagragian-Coupling module (ELC)



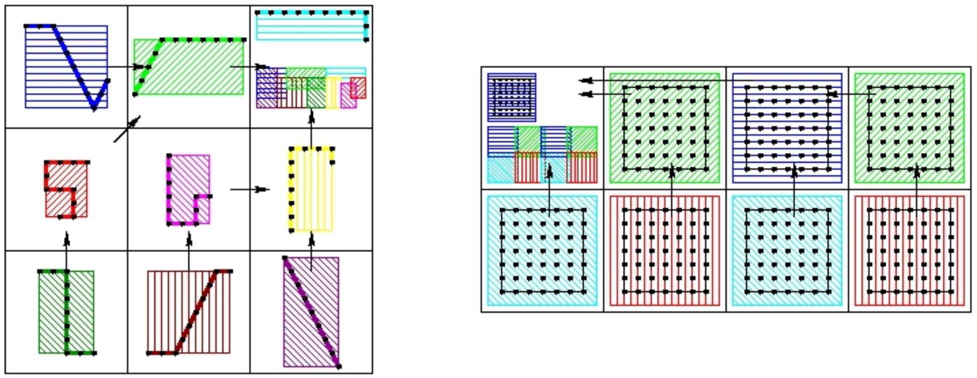
- *When SAMR mesh partitioning is done at runtime, the entire solid mesh must have been received (SAMR partitions must be allowed to change arbitrary)*
- *During strictly local regridding operations only the local portion of the solid mesh has to be received*

ELC communication module

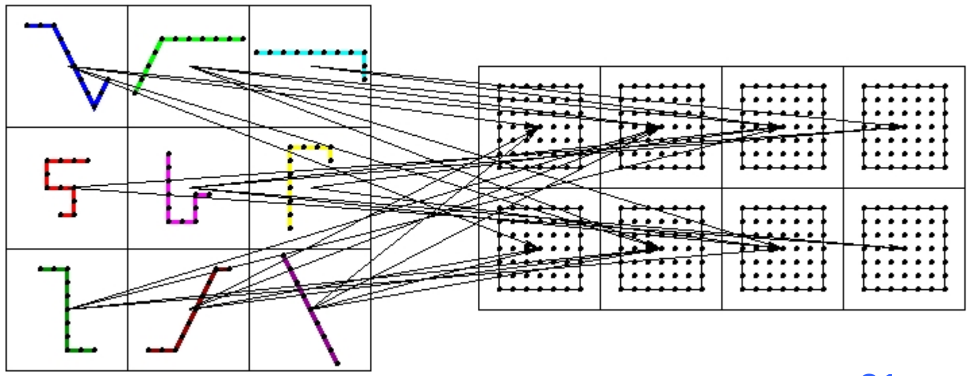
1. Put bounding boxes around each solid processor's piece of the boundary and around each fluid processor's grid.



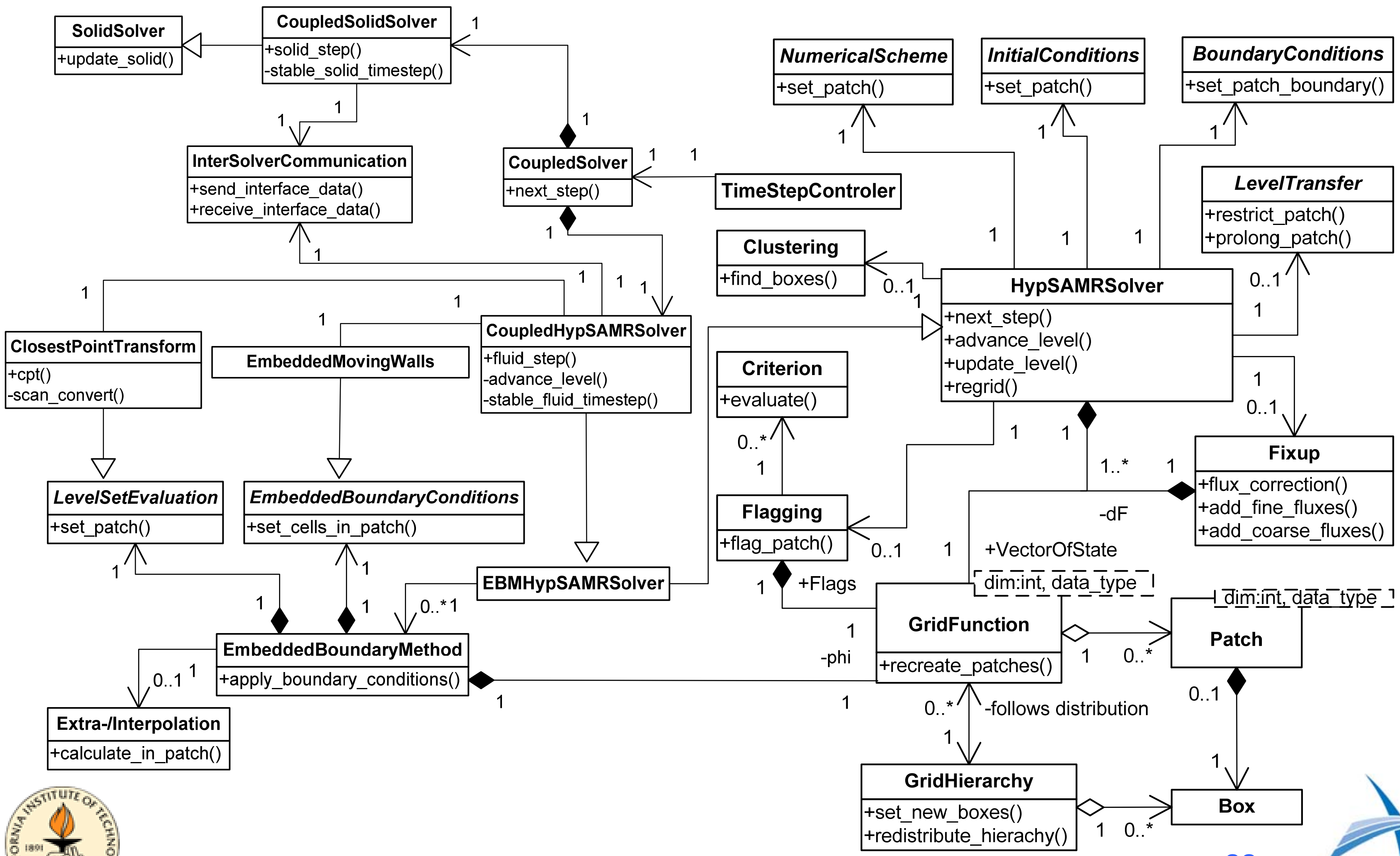
2. Gather, exchange and broadcast of bounding box information



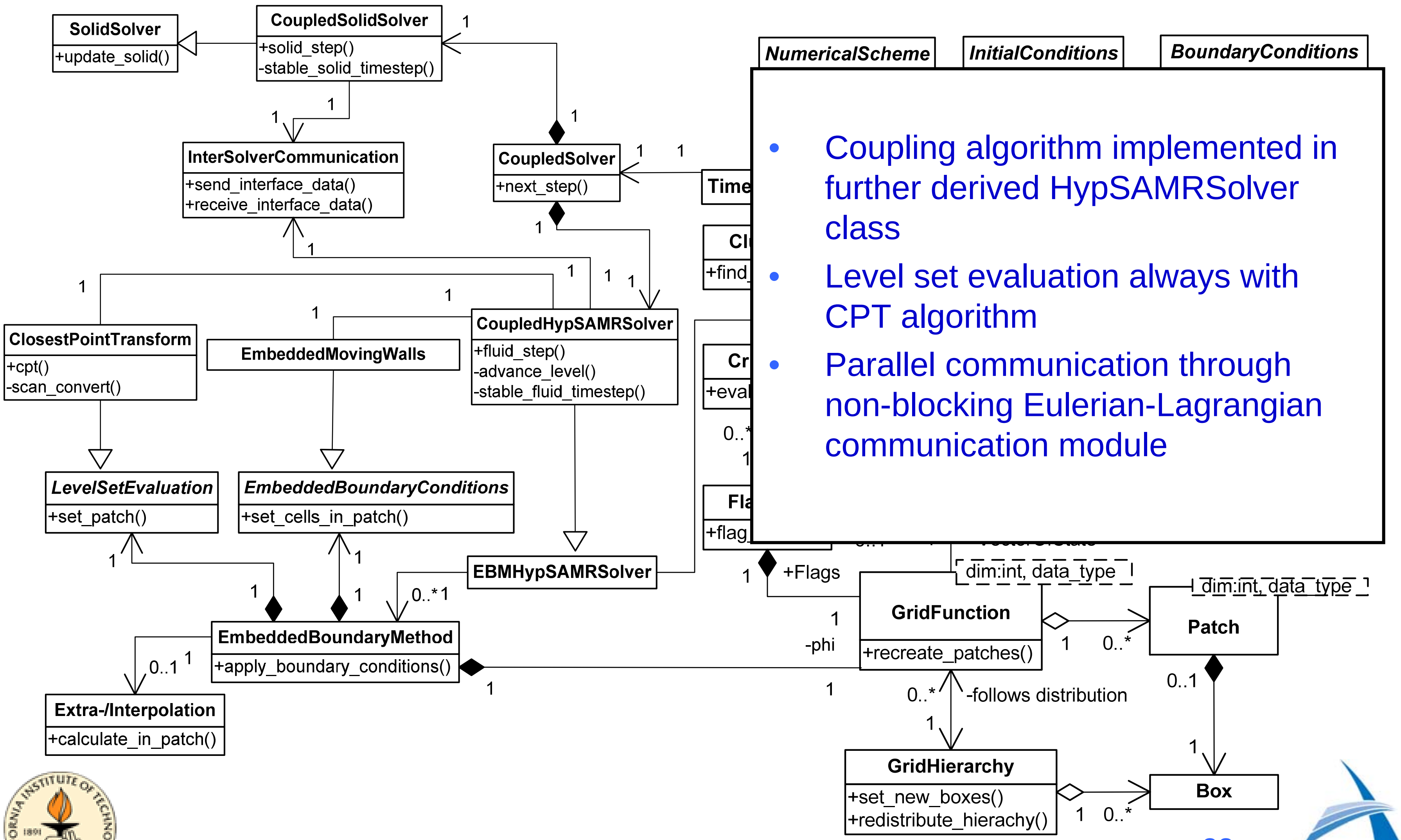
3. Optimal point-to-point communication pattern, non-blocking



Amroc coupled to CSD solver (the VTF)



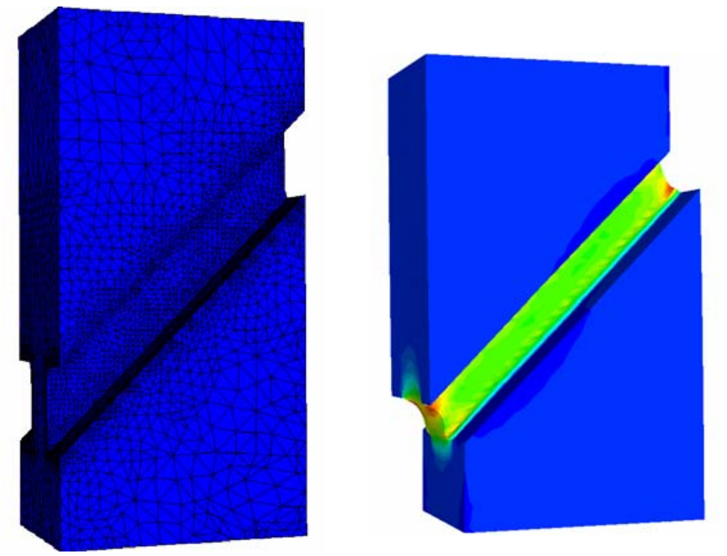
Amroc coupled to CSD solver (the VTF)



- Coupling algorithm implemented in further derived HypSAMRSolver class
- Level set evaluation always with CPT algorithm
- Parallel communication through non-blocking Eulerian-Lagrangian communication module

Solid mechanics: adlib

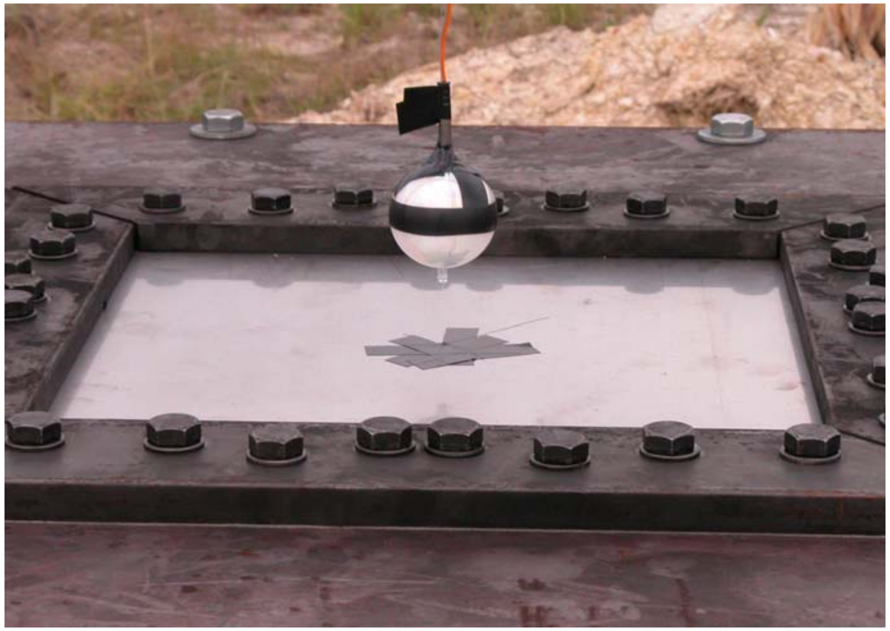
- Parallel explicit dynamics
- Fully scalable communications
- Solid modeling
- Fully scalable unstructured parallel meshing
- Thermomechanical coupling and multiphysics models
  - *Extensive constitutive library*
    - *single and polycrystal plasticity*
    - *ab initio EOS*
    - *shock physics, artificial viscocity*
- Contact
- Fracture and fragmentation
- Coupling to other solvers



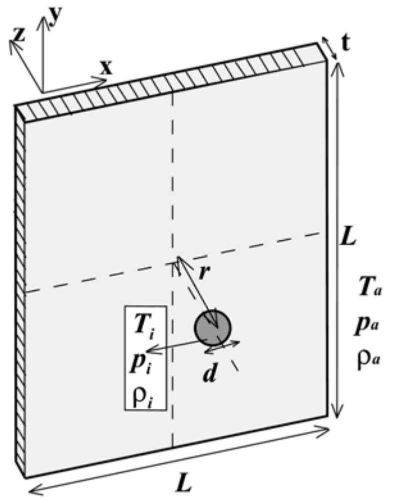
Explosively loaded steel plates

- Perfect clamping of the plate, constant C4 charge (150 g) at different stand-off distances *r*:

Experimental tests
(K. Dharmasena, H. Wadley, University of Virginia)



Numerical simulations
(L. Noels, R. Radovitzky, MIT)

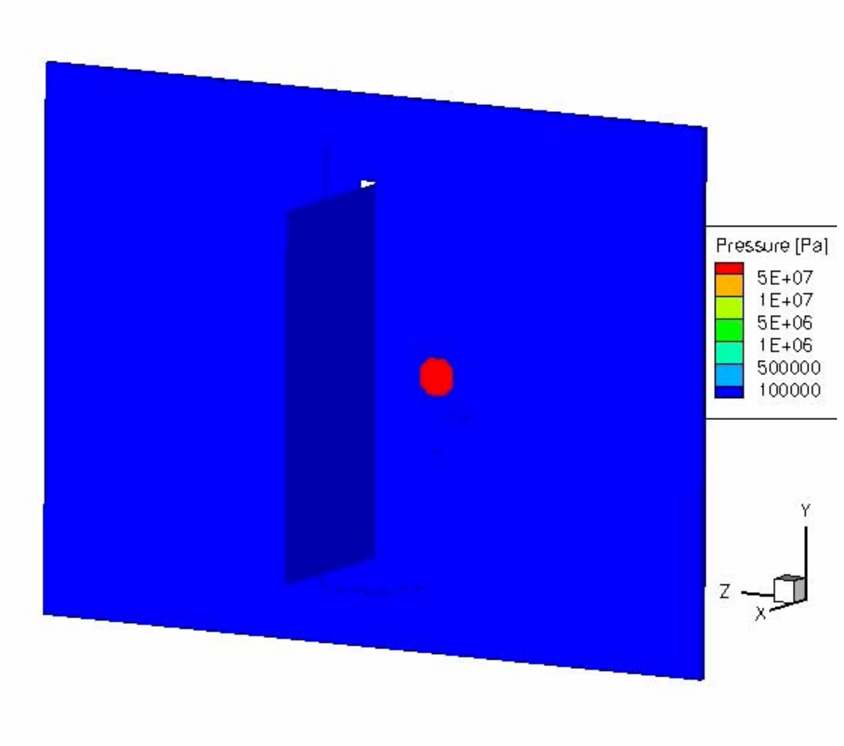


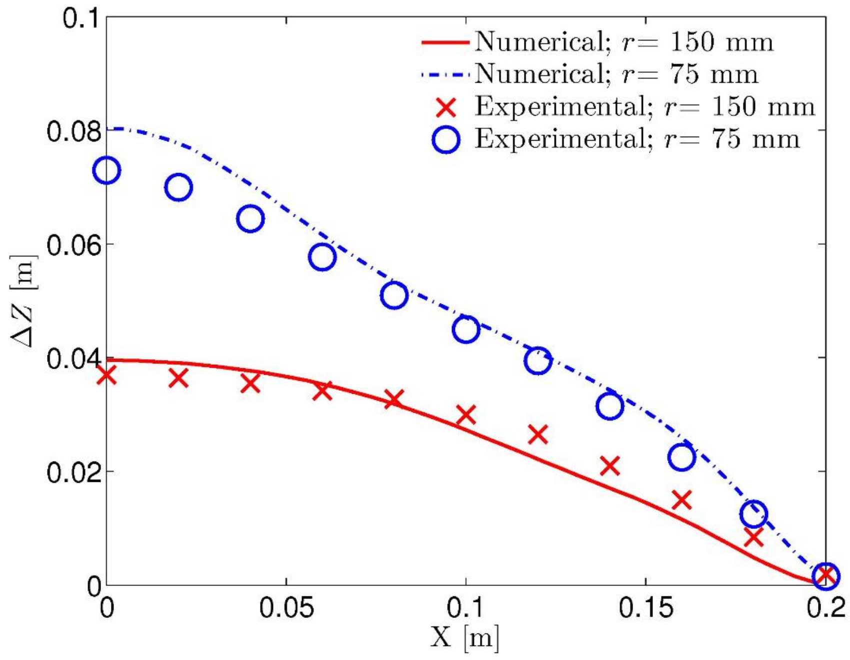
TNT [kg]	*L* [mm]	*t* [mm]	*r* [m]	*d* [m]	T_i [K]	p_i [GPa]	ρ_i [kg/m^3]
0.192	406.4	1.9	**0.15**	0.04	5860	10.4	6220
0.192	406.4	1.9	**0.075**	0.03	5860	24.7	14750

Explosively loaded steel plates - Validation

- Numerical simulations (5 nodes 2.2 GHz AMD Opteron quad processor; 480 h CPU):

Deformed profiles

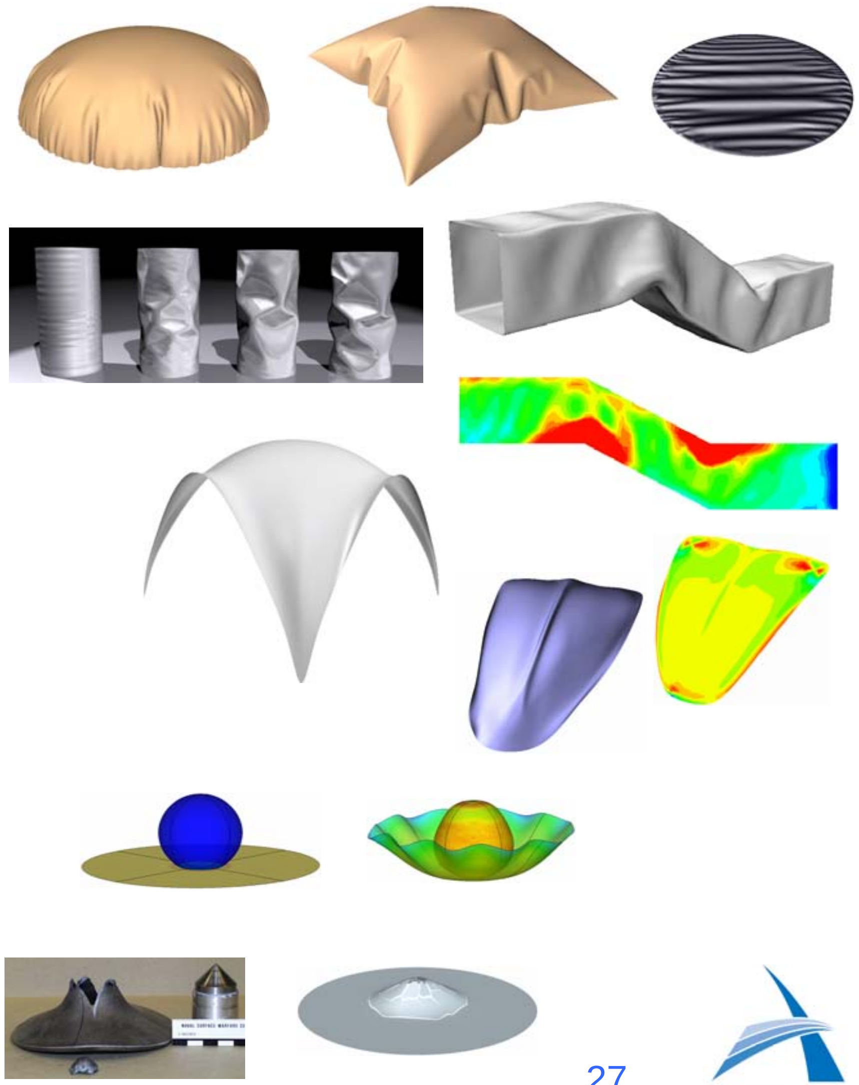




Time evolution

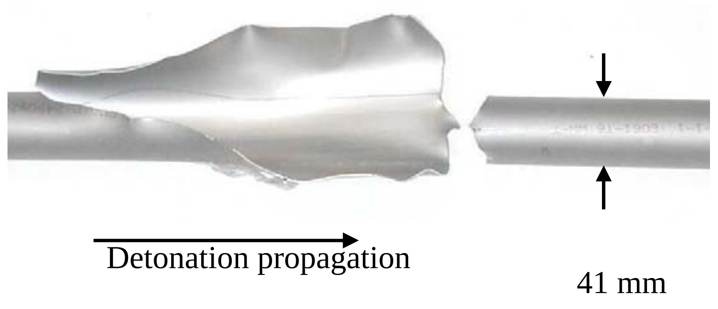
Solid mechanics: sfc

- Subdivision shell finite elements
  - *Stretching and bending resistance*
  - *Large deformations*
- Parallel explicit shell dynamics
  - *Fully scalable communications*
- Geometric modeling capabilities
- Access to a number of constitutive models
  - *Adlib models as well as own implementations*
- Parallel contact
- Fracture and fragmentation



Validation example: detonation-driven fracture

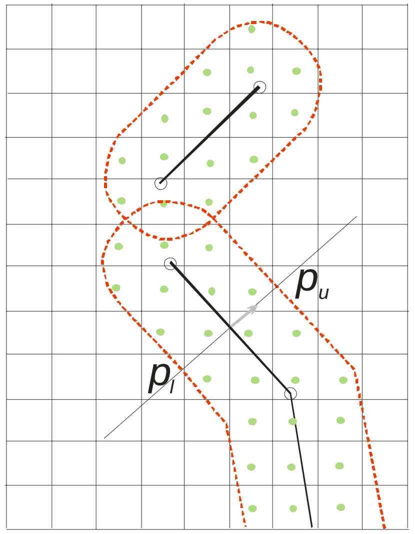
- Experiments by T. Chao, J. C. Krok, J. Karnesky, F. Pintgen, J.E. Shepherd (GalCIT)
- Motivation: Validate VTF for complex fluid-structure interaction problem
- Interaction of detonation, ductile deformation, fracture
- Modeling of ethylene-oxygen detonation with constant volume burn detonation model



Treatment of shells/thin structures

- Thin boundary structures or lower-dimensional shells require "thickening" to apply ghost fluid method
  - *Unsigned distance level set function φ*
  - *Treat cells with $0<\varphi<d$ as ghost fluid cells (indicated by green dots)*
  - *Leaving φ unmodified ensures correctness of $\nabla\varphi$*
  - *Refinement criterion based on φ ensures reliable mesh adaptation*
  - *Use face normal in shell element to evaluate in*

  $\Delta p = p_u - p_l$



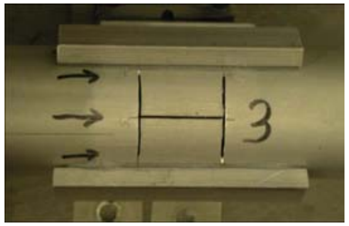
Elastic-plastic validation – Tube with flaps

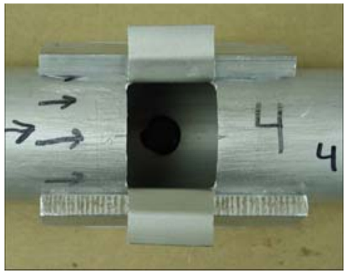
Fluid

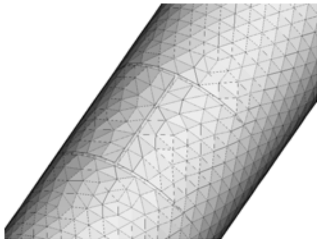
- Constant volume burn model with γ=1.24, P_{CJ} =3.3 MPa, D_{CJ}=2376 m/s
- AMR base level: 104x80x242, 3 additional levels, factors 2,2,4
- Approx. $4 \cdot 10^7$ cells instead of $7.9 \cdot 10^9$ cells (uniform)
- Tube and detonation fully refined
- Thickening of 2d mesh: 0.81mm on both sides (real thickness on both sides 0.445mm)
- 16 nodes 2.2 GHz AMD Opteron quad processor, PCI-X 4x Infiniband network

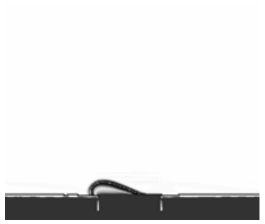
Solid – Thin-shell solver by F. Cirak

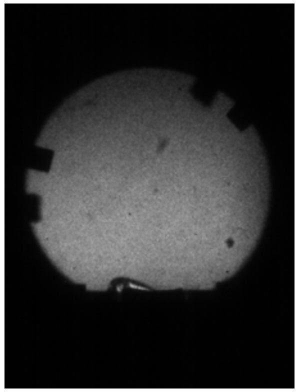
- Aluminum, J2 plasticity with hardening, rate sensitivity, and thermal softening
- Mesh: 8577 nodes, 17056 elements
- 16+2 nodes 2.2 GHz AMD Opteron quad processor, PCI-X 4x Infiniband network
- Ca. 4320h CPU to t=450 μs

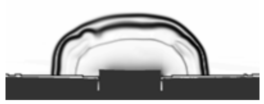


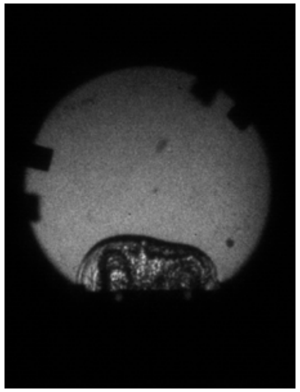





2 μs


0 μs


32 μs


30 μs

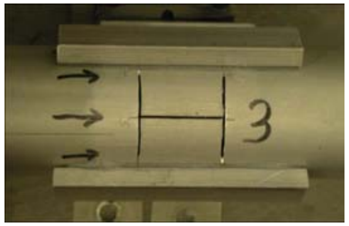
Elastic-plastic validation – Tube with flaps

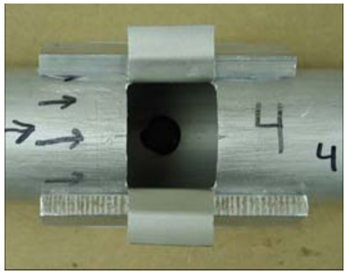
Fluid

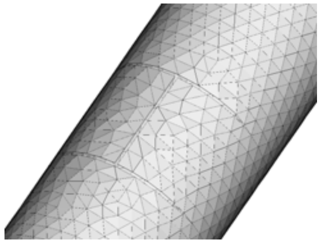
- Constant volume burn model with γ=1.24, P_{CJ} =3.3 MPa, D_{CJ}=2376 m/s
- AMR base level: 104x80x242, 3 additional levels, factors 2,2,4
- Approx. $4 \cdot 10^7$ cells instead of $7.9 \cdot 10^9$ cells (uniform)
- Tube and detonation fully refined
- Thickening of 2d mesh: 0.81mm on both sides (real thickness on both sides 0.445mm)
- 16 nodes 2.2 GHz AMD Opteron quad processor, PCI-X 4x Infiniband network

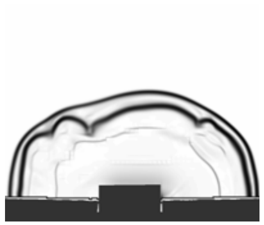
Solid – Thin-shell solver by F. Cirak

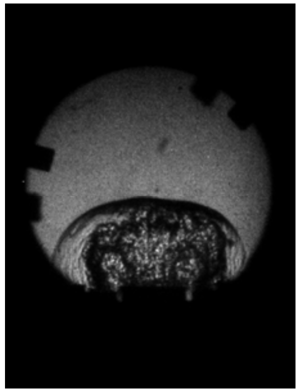
- Aluminum, J2 plasticity with hardening, rate sensitivity, and thermal softening
- Mesh: 8577 nodes, 17056 elements
- 16+2 nodes 2.2 GHz AMD Opteron quad processor, PCI-X 4x Infiniband network
- Ca. 4320h CPU to t=450 μs

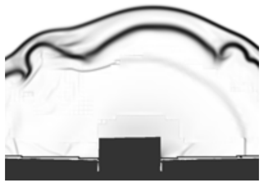


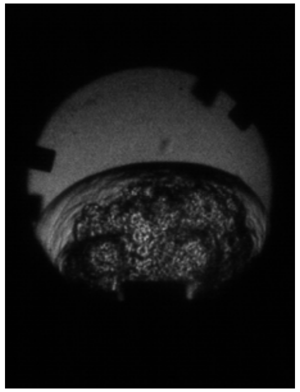





62 μs


60 μs


92 μs


90 μs

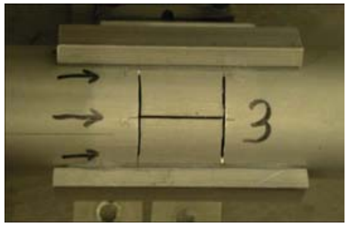
Elastic-plastic validation – Tube with flaps

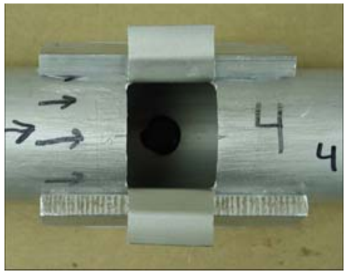
Fluid

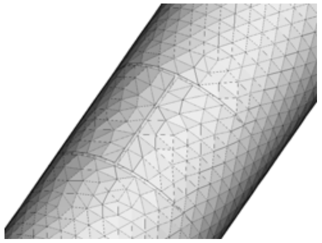
- Constant volume burn model with γ=1.24, P_{CJ} =3.3 MPa, D_{CJ}=2376 m/s
- AMR base level: 104x80x242, 3 additional levels, factors 2,2,4
- Approx. $4 \cdot 10^7$ cells instead of $7.9 \cdot 10^9$ cells (uniform)
- Tube and detonation fully refined
- Thickening of 2d mesh: 0.81mm on both sides (real thickness on both sides 0.445mm)
- 16 nodes 2.2 GHz AMD Opteron quad processor, PCI-X 4x Infiniband network

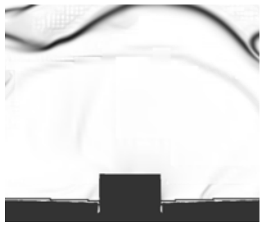
Solid – Thin-shell solver by F. Cirak

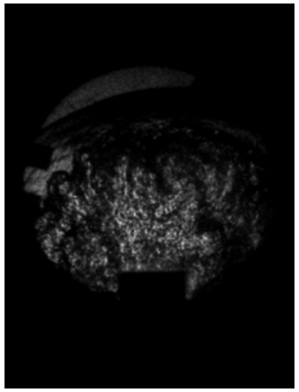
- Aluminum, J2 plasticity with hardening, rate sensitivity, and thermal softening
- Mesh: 8577 nodes, 17056 elements
- 16+2 nodes 2.2 GHz AMD Opteron quad processor, PCI-X 4x Infiniband network
- Ca. 4320h CPU to t=450 μs

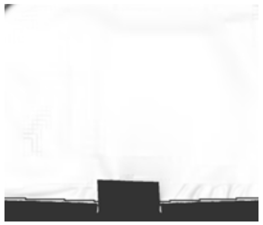


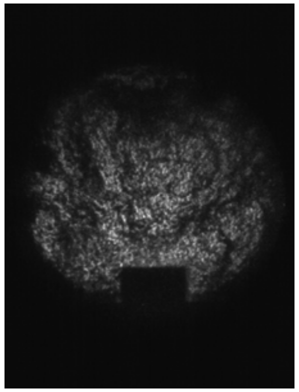




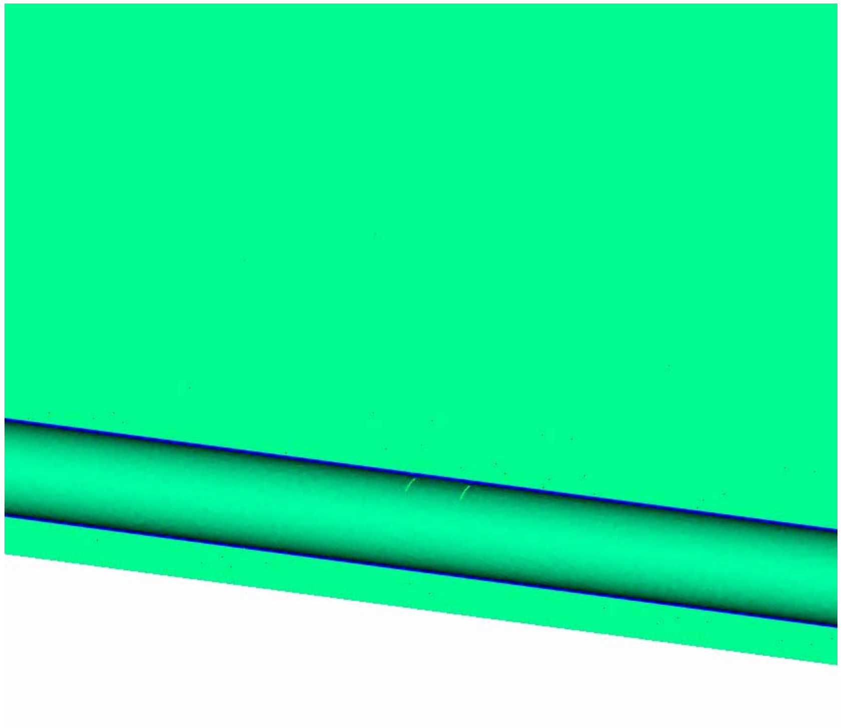

152 μs

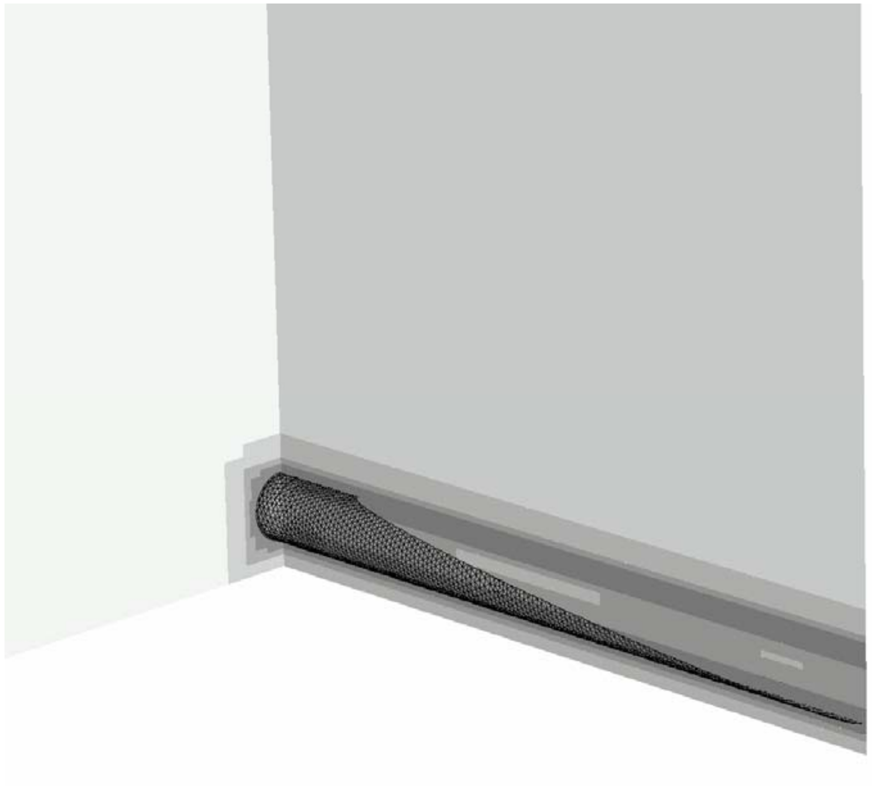

150 μs


212 μs

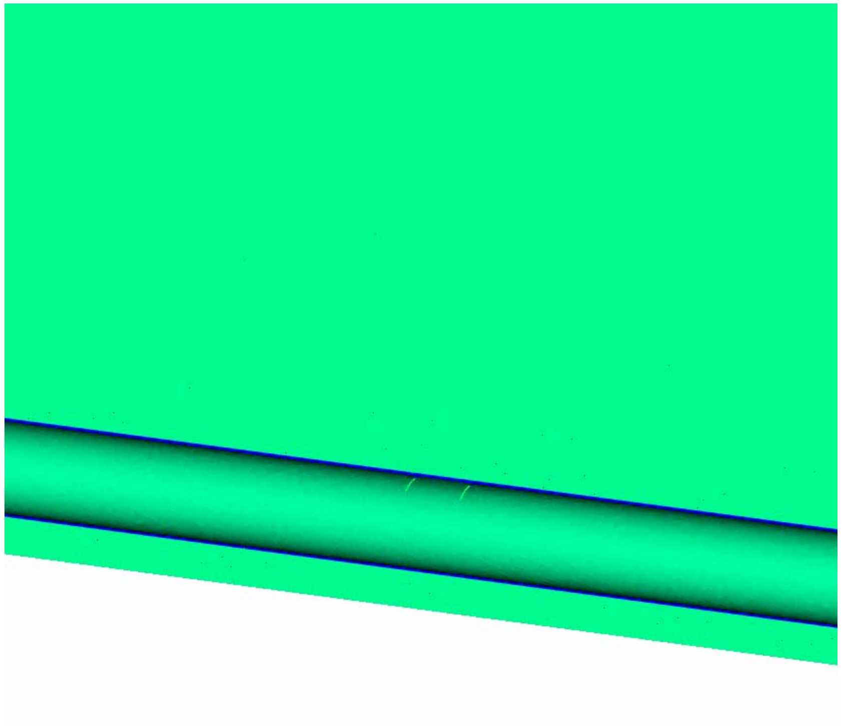

210 μs

Tube with flaps - Results

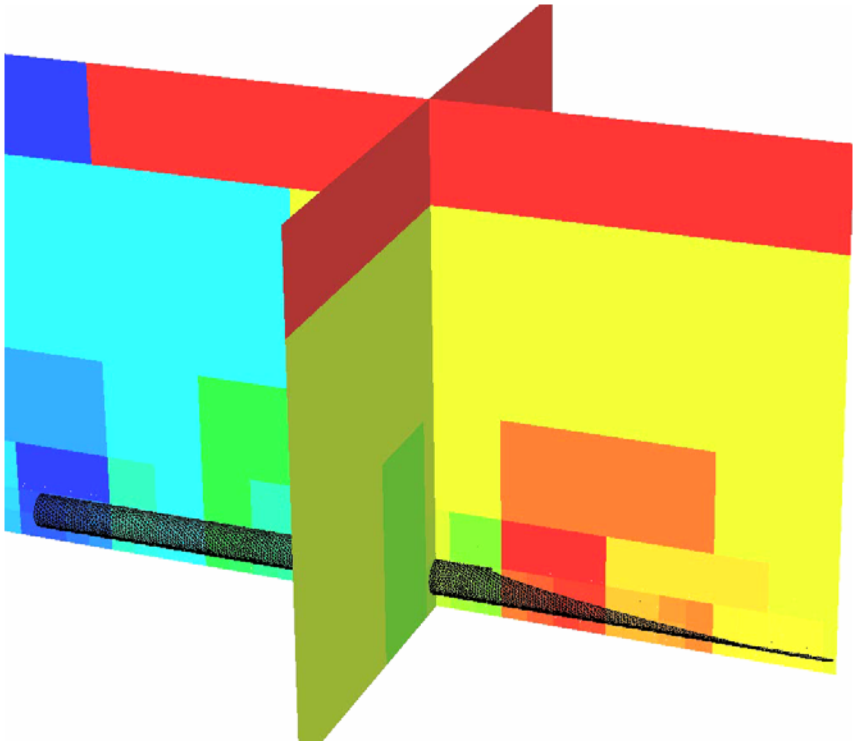

Fluid density and diplacement in y-direction in solid


Schlieren plot of fluid density on refinement levels

Tube with flaps - Results



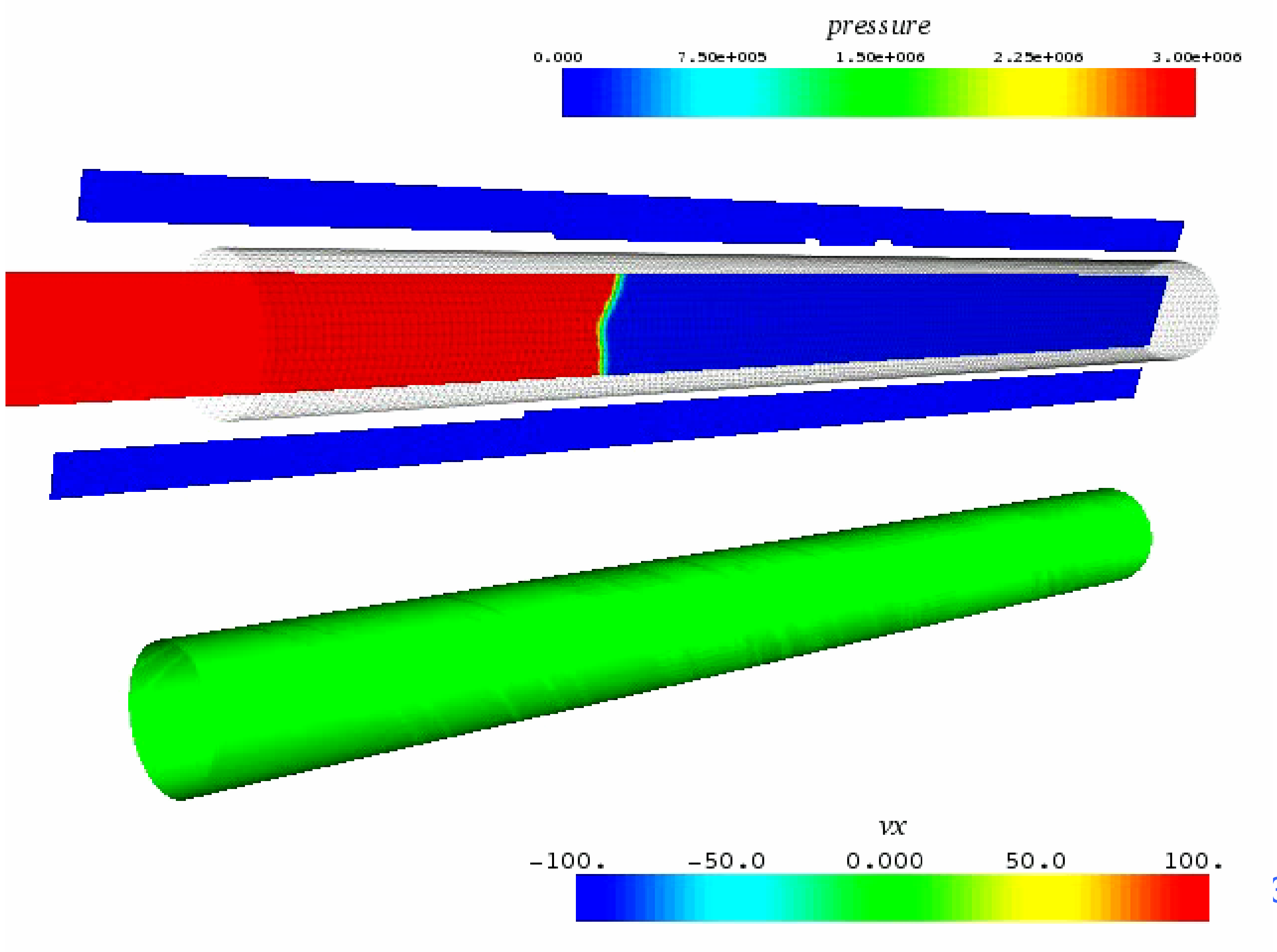
Fluid density and diplacement in
y-direction in solid



Schlieren plot of fluid density on
refinement levels

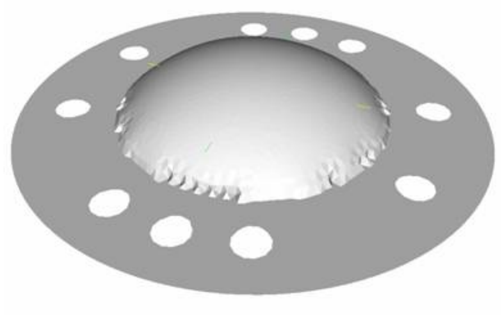
Coupled simulations with fracture

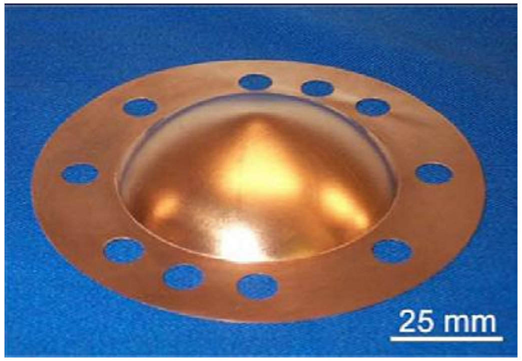
- Coupled simulation with one-step detonation model
- J2-plasticity model and cohesive elements to model fracture
- Solid mesh: ~ 10,000 elements
- ~ 2000h CPU on 64 processors Compaq QSC



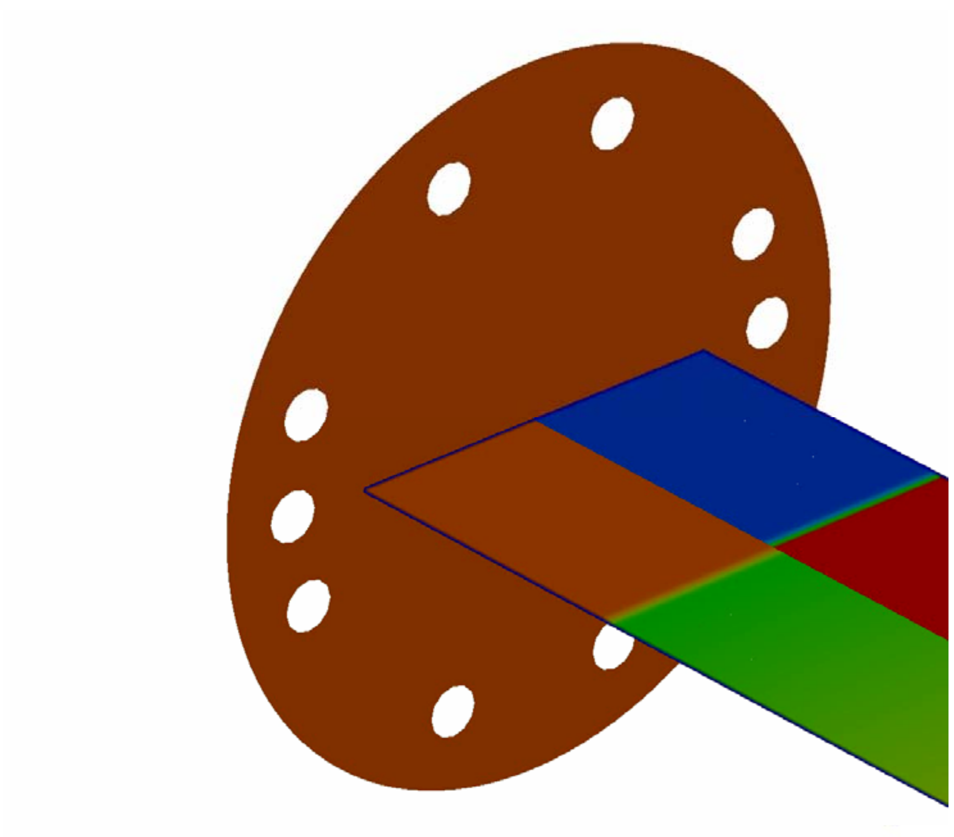
Deformation of plates by a water hammer

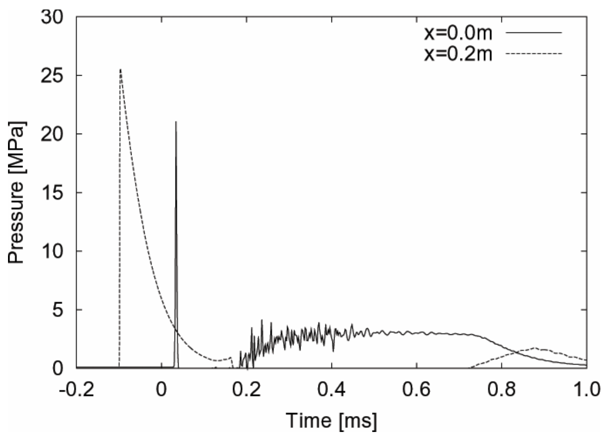
- Experimental results provided by V.S. Deshpande, University Cambridge.
- Strong pressure wave in water created by piston.
- Wave impinges onto a thin copper plate
- Depending on the initial projectile velocity the pressure induces plastic deformation of different rupture patterns.
- Fluid
  - *Pressure wave generated by solving equation of motion for piston during entire fluid-structure simulation.*
  - *Modeling of water with stiffened gas equation of state with*
  - *AMR base level: 350x20x20, 2 additional levels, refinement factor 2,2.*
  - *Approx. 1,2M cells used in fluid on average instead of 9M (uniform)*
- Solid
  - *SFC solver*
  - *Copper plate of 0.25mm, J2 plasticity model with hardening, rate sensitivity, and thermal softening*
  - *Solid mesh: 4675 nodes, 8896 elements*

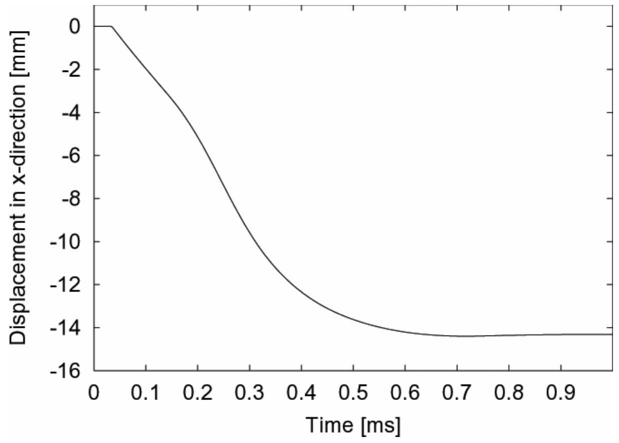




Plastic deformation



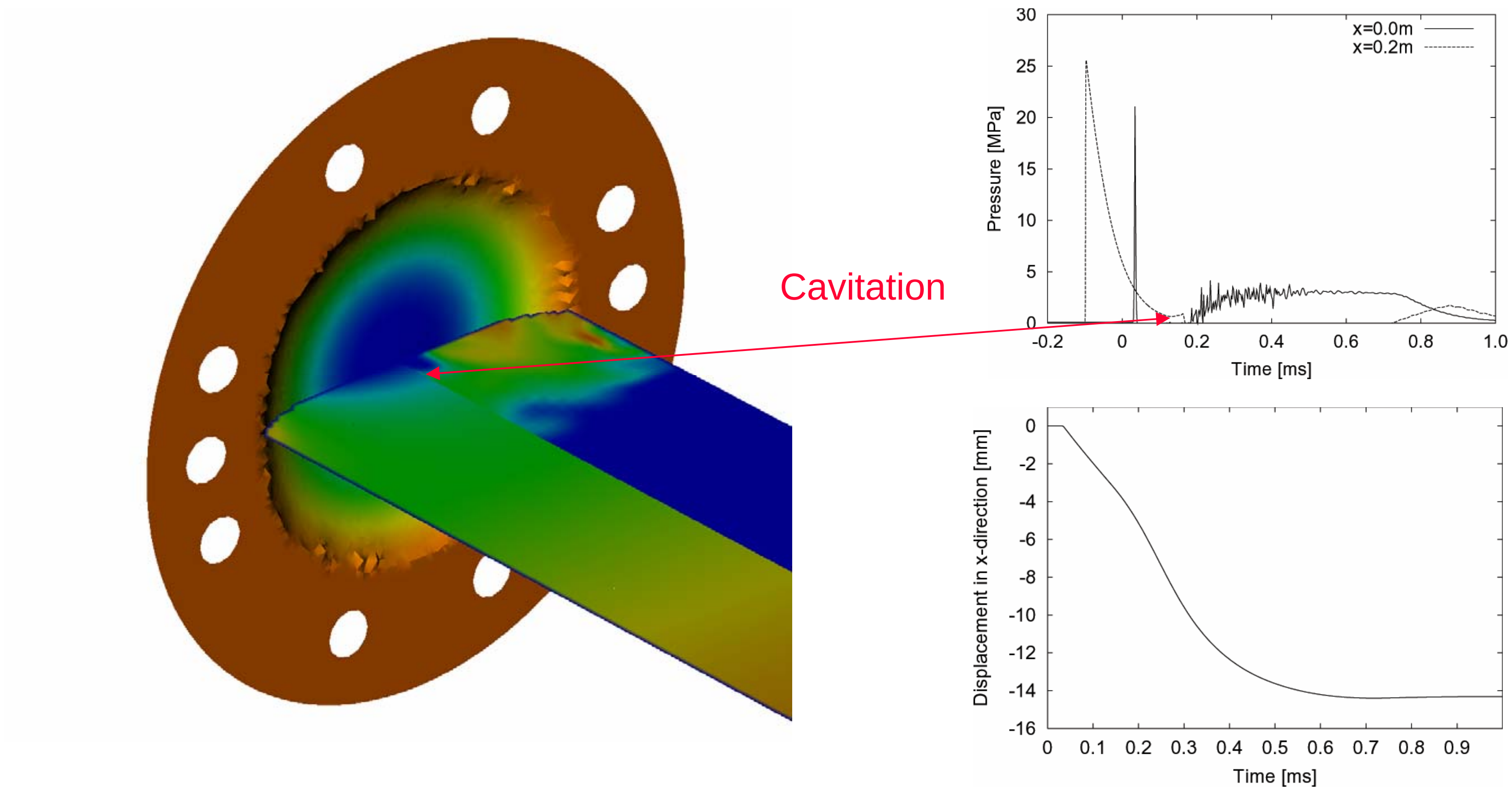




Color of plate and lower half of plane shows the normal velocity

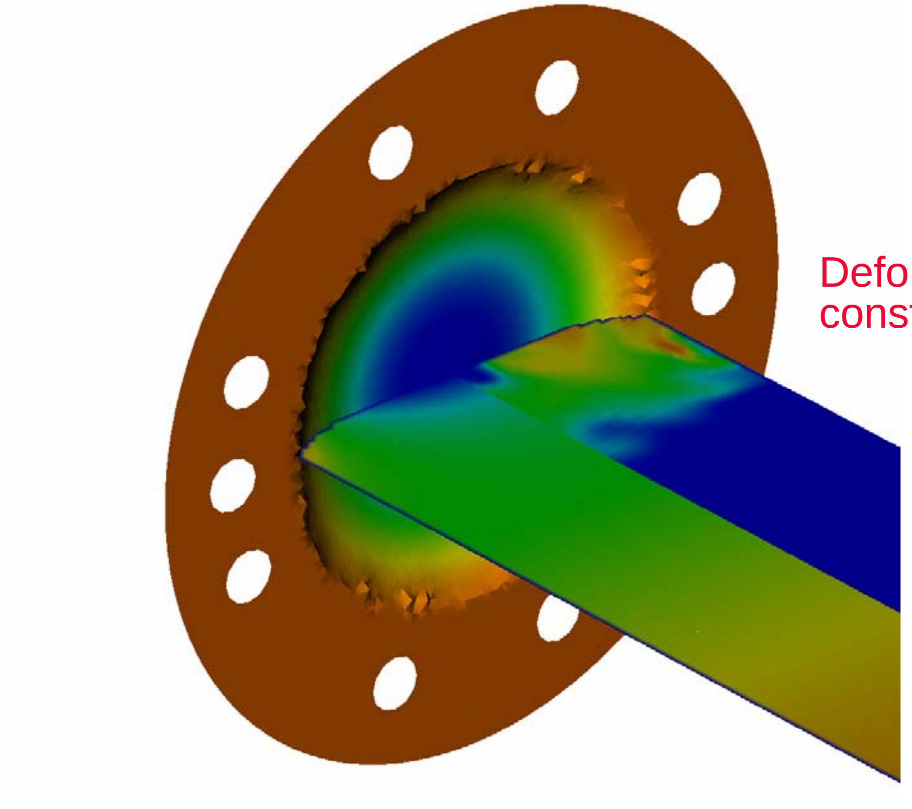
- *p_0=34 MPa*
- *8 nodes 3.4 GHz Intel Xeon dual processor, Gigabit ethernet network, ca. 130h CPU*

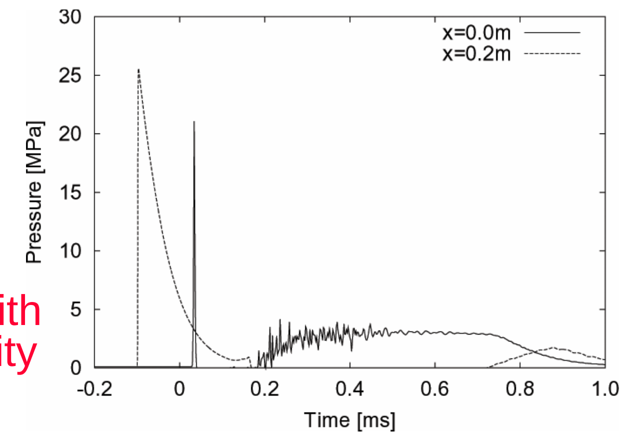
Plastic deformation



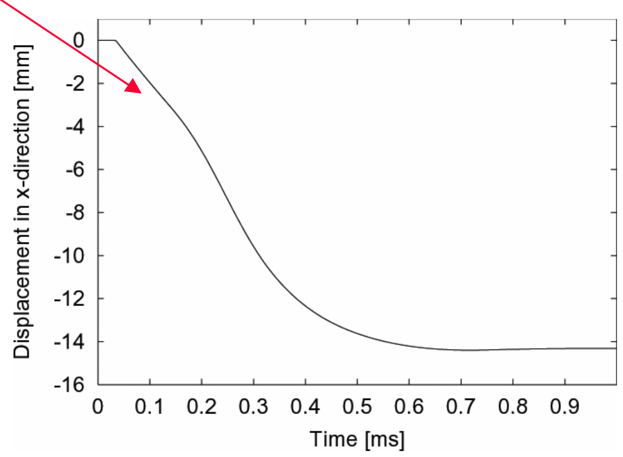
- p_0=34 MPa
- 8 nodes 3.4 GHz Intel Xeon dual processor, Gigabit ethernet network, ca. 130h CPU

Plastic deformation



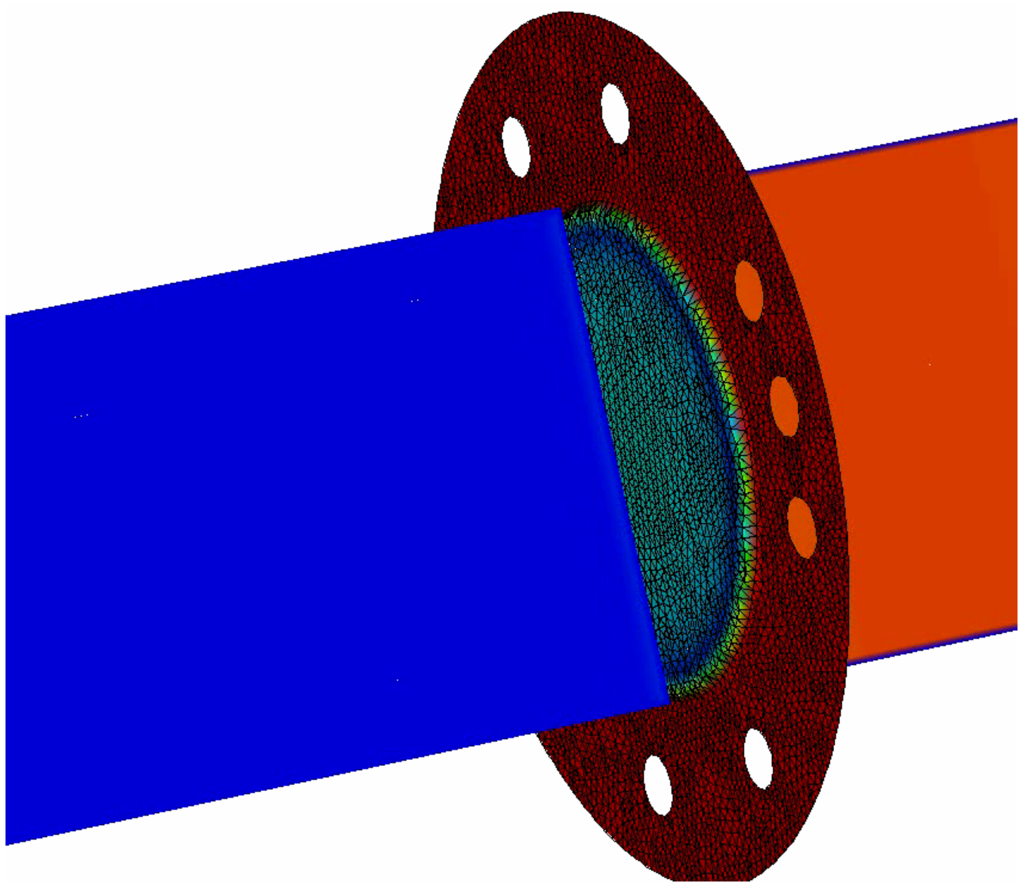


Deformation with constant velocity



- p_0=34 MPa
- 8 nodes 3.4 GHz Intel Xeon dual processor, Gigabit ethernet network, ca. 130h CPU

Plate fracture



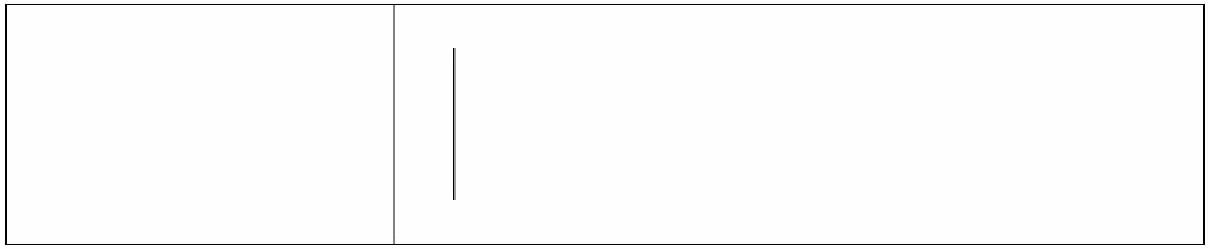
- *Two-component solver with stiffened gas EOS for water and ideal gas EOS for air*
- *Material model for cohesive interface: linear decreasing envelope, cohesive stress σ_c=525 MPa*
- *4+4 nodes 3.4 GHz Intel Xeon dual processor, ca. 550h CPU*

FSI example: shock-induced panel motion

- Elastic motion of a thin steel plate (thickness h=1mm, length 50mm) hit by a Mach 1.21 shock wave in air, Giordano et al. Shock Waves (2005)
- Steel plate modeled with finite difference solver using the beam equation

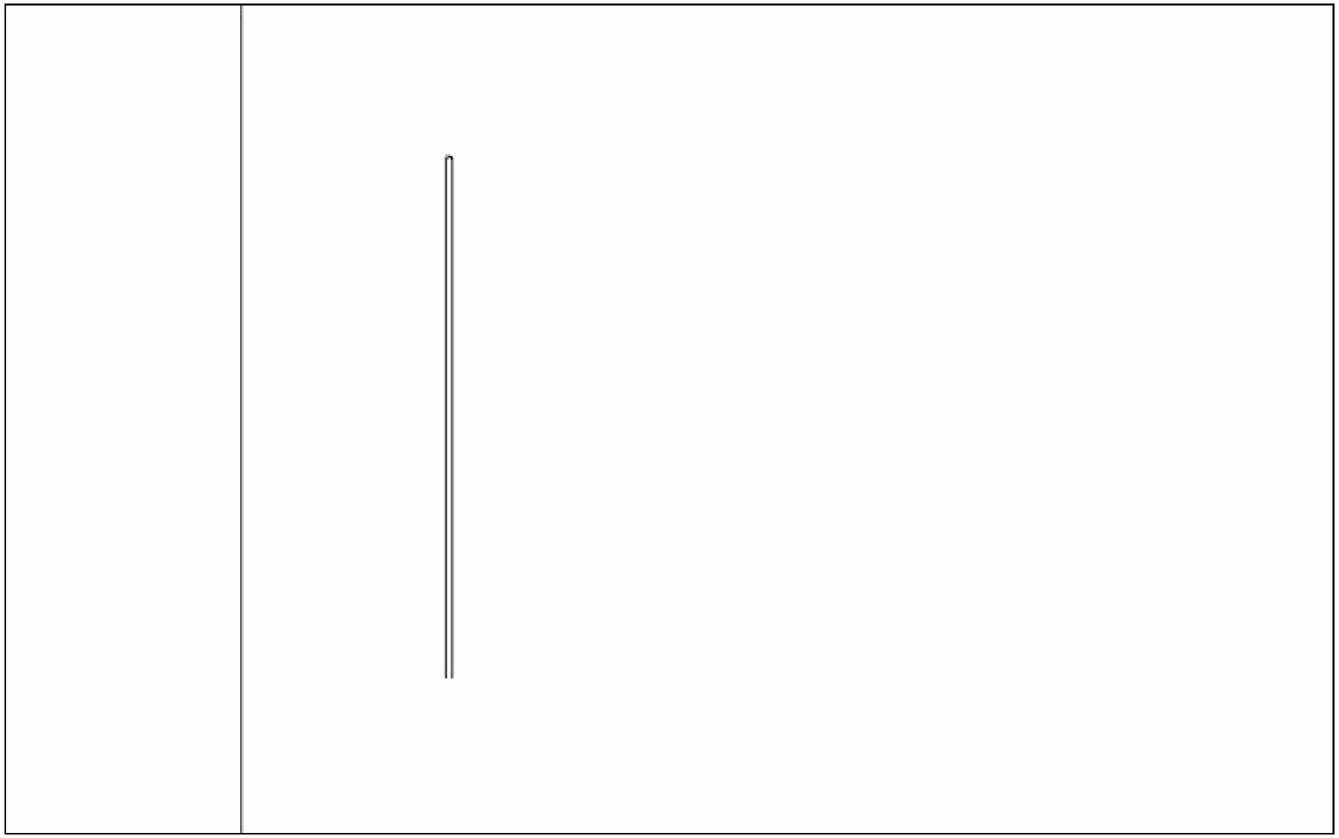
$$\rho h \frac{\partial^2 w}{\partial t^2} + EI \frac{\partial^4 w}{\partial x^4} = p(x,t)$$

- Forward facing step geometry, reflective boundaries everywhere except inflow at left side, panel 1.5cm behind start of step
- SAMR base mesh 320x64, 2 additional level with factors 2, 4
- 54h CPU on 4 nodes with Intel 3.4GHz Xeon dual processors connected with Gigabit Ethernet



Schlieren plot of density

Shock-induced panel motion



Schlieren plot of density enlarged to
show panel motion

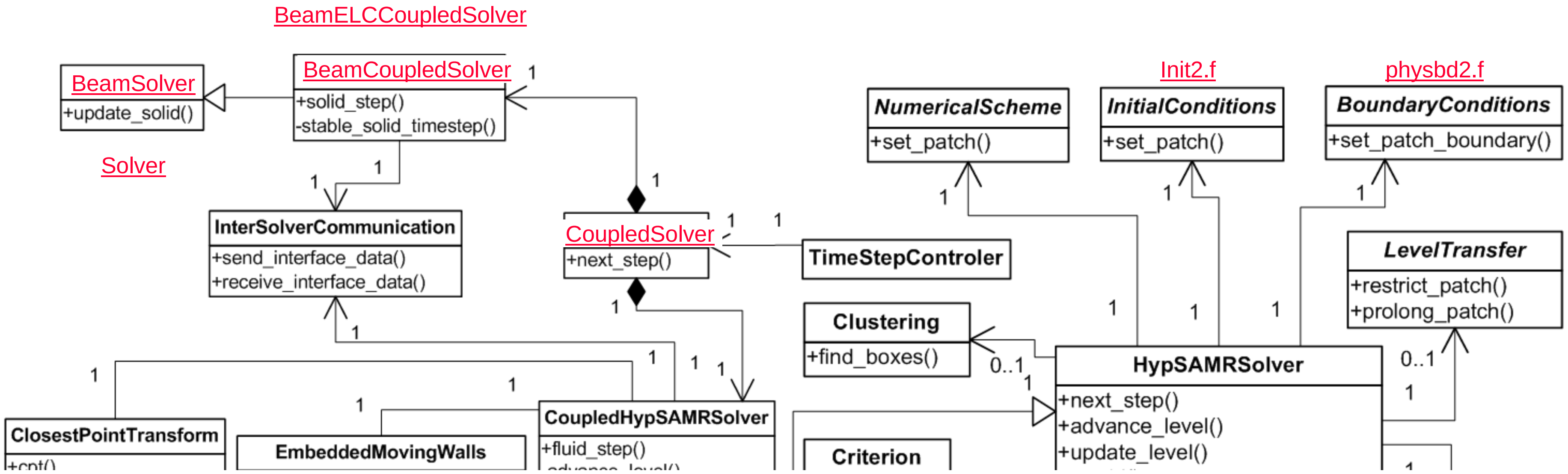
vtf/fsi/beam-solver/VibratingBeam

vtf/fsi/beam-amroc/VibratingBeam

- solver.in
- display_file.in
- run.py

vtf/fsi/beam-amroc/VibratingBeam/src

- FluidProblem.h
- SolidProblem.h
- Makefile.am


BeamELCCoupledSolver
BeamSolver
+update_solid()
Solver
BeamCoupledSolver
+solid_step()
-stable_solid_timestep()
InterSolverCommunication
+send_interface_data()
+receive_interface_data()
CoupledSolver
+next_step()
TimeStepControler
NumericalScheme
+set_patch()
Init2.f
InitialConditions
+set_patch()
physbd2.f
BoundaryConditions
+set_patch_boundary()
LevelTransfer
+restrict_patch()
+prolong_patch()
Clustering
+find_boxes()
HypSAMRSolver
+next_step()
+advance_level()
+update_level()
CoupledHypSAMRSolver
+fluid_step()
ClosestPointTransform
EmbeddedMovingWalls
Criterion
1
0..1