Third generation computational fluid dynamics: examples of adaptive Cartesian simulations with the AMROC framework

Ralf Deiterding

Aerodynamics and Flight Mechanics Research Group
University of Southampton
Boldrewood Campus, Southampton SO16 7QF, UK
E-mail: r.deiterding@soton.ac.uk

October 28, 2019

Computational fluid dynamics – CFD

Solve the equations of fluid dynamics computationally, usually on a grid

Computational fluid dynamics – CFD

Solve the equations of fluid dynamics computationally, usually on a grid

- 1st generation: structured non-Cartesian grid
- 2nd generation: unstructured grids
- 3rd generation: adaptive Cartesian (with geometry embedding)

Computational fluid dynamics – CFD

Solve the equations of fluid dynamics computationally, usually on a grid

- 1st generation: structured non-Cartesian grid
- 2nd generation: unstructured grids
- 3rd generation: adaptive Cartesian (with geometry embedding)

Approach with commercial CFD software separated into the steps

1. Pre-processing
2. Solution
3. Post-processing

Computational fluid dynamics – CFD

Solve the equations of fluid dynamics computationally, usually on a grid

- 1st generation: structured non-Cartesian grid
- 2nd generation: unstructured grids
- 3rd generation: adaptive Cartesian (with geometry embedding)

Approach with commercial CFD software separated into the steps

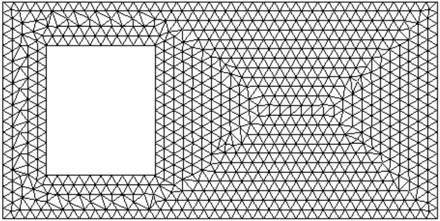
1. Pre-processing
2. Solution
3. Post-processing

Idea of Cartesian methods is to automate the grid generation and incorporate it into the solution.

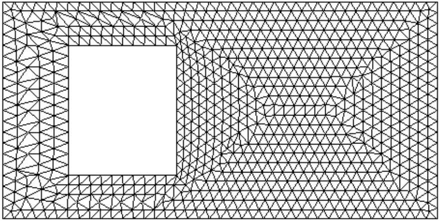
Especially with Cartesian methods post-processing is also increacingly incorporated into the solution process.

Outline

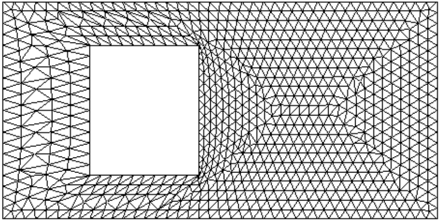
FSI simulation with unstructured grids



FSI simulation with unstructured grids

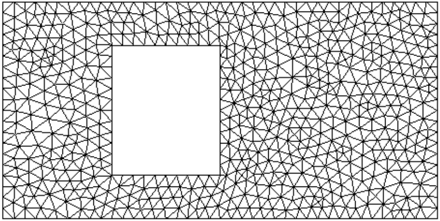


FSI simulation with unstructured grids



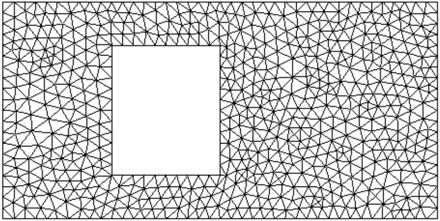
FSI simulation with unstructured grids

- Global grid (re-)generation is part of the simulation (major parallelization and scalability obstacle)
- Sophisticated data remapping required when grid topology changes



FSI simulation with unstructured grids

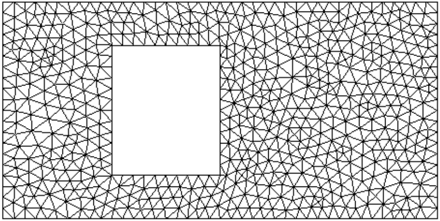
- Global grid (re-)generation is part of the simulation (major parallelization and scalability obstacle)
- Sophisticated data remapping required when grid topology changes
- Some generalities about unstructured grids:
- + Hanging nodes can be avoided
- Higher order difficult to achieve
- High computational performance challenging



FSI simulation with unstructured grids

- Global grid (re-)generation is part of the simulation (major parallelization and scalability obstacle)
- Sophisticated data remapping required when grid topology changes
- ▶ Some generalities about unstructured grids:
- \+ Hanging nodes can be avoided
- \- Higher order difficult to achieve
- \- High computational performance challenging

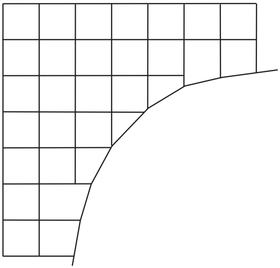
$\longrightarrow$ Alternative: Adaptive Cartesian CFD methods with embedded boundaries



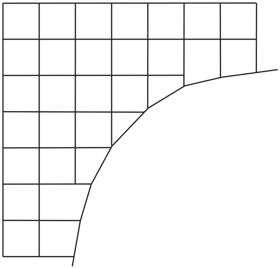
Geometry handling in Cartesian methods

Methods that represent the boundary sharply

- Cut-cell approach constructs appropriate finite volumes
- Conservative by construction. Correct boundary flux
- Key question: How to avoid small-cell time step restriction in explicit methods?



Geometry handling in Cartesian methods



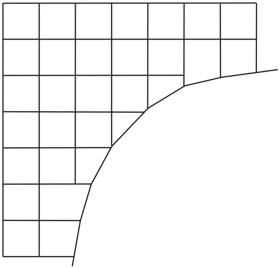
Methods that represent the boundary sharply

- Cut-cell approach constructs appropriate finite volumes
- Conservative by construction. Correct boundary flux
- Key question: How to avoid small-cell time step restriction in explicit methods?

Methods that diffuse the boundary in one cell [Mittal and Iaccarino, 2005]

- Related to the immersed boundary method by Peskin, cf. [Roma et al., 1999]
- Boundary prescription often by internal ghost cell values
- Not conservative by construction but conservative correction possible
- Usually combined with implicit geometry representation

Geometry handling in Cartesian methods



Methods that represent the boundary sharply

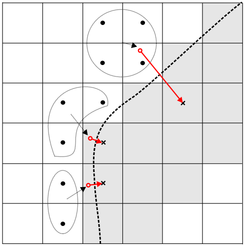
- Cut-cell approach constructs appropriate finite volumes
- Conservative by construction. Correct boundary flux
- Key question: How to avoid small-cell time step restriction in explicit methods?

Methods that diffuse the boundary in one cell [Mittal and Iaccarino, 2005]

- Related to the immersed boundary method by Peskin, cf. [Roma et al., 1999]
- Boundary prescription often by internal ghost cell values
- Not conservative by construction but conservative correction possible
- Usually combined with implicit geometry representation

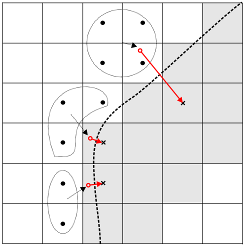
Volume of fluid methods that resemble a cut-cell technique, e.g. [Berger and Helzel, 2002]

Level-set method for boundary embedding



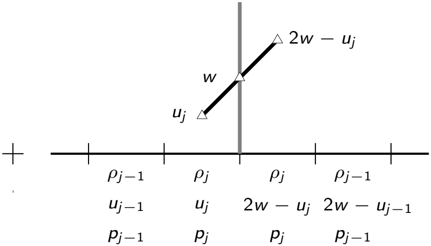
- Implicit boundary representation via distance function φ, normal $\mathbf{n} = \nabla\varphi / |\nabla\varphi|$
- Complex boundary moving with local velocity $\mathbf{w}$, treat interface as moving rigid wall
- Construction of values in embedded boundary cells by interpolation / extrapolation

Level-set method for boundary embedding

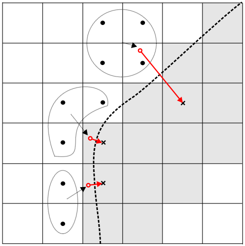


- Implicit boundary representation via distance function φ, normal $\mathbf{n} = \nabla\varphi / |\nabla\varphi|$
- Complex boundary moving with local velocity $\mathbf{w}$, treat interface as moving rigid wall
- Construction of values in embedded boundary cells by interpolation / extrapolation

Interpolate / constant value extrapolate values at $\tilde{\mathbf{x}} = \mathbf{x} + 2\varphi\mathbf{n}$



Level-set method for boundary embedding

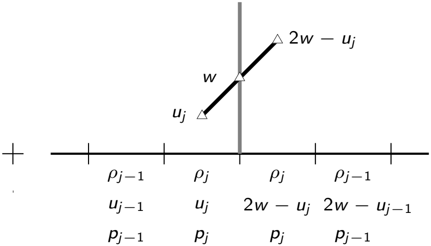


- Implicit boundary representation via distance function φ, normal $\mathbf{n} = \nabla\varphi/|\nabla\varphi|$
- Complex boundary moving with local velocity $\mathbf{w}$, treat interface as moving rigid wall
- Construction of values in embedded boundary cells by interpolation / extrapolation

Interpolate / constant value extrapolate values at $\tilde{\mathbf{x}} = \mathbf{x} + 2\varphi\mathbf{n}$

Velocity in ghost cells
No-slip: $\mathbf{u}' = 2\mathbf{w} - \mathbf{u}$
Slip:

$$\mathbf{u}' = (2\mathbf{w}\cdot\mathbf{n} - \mathbf{u}\cdot\mathbf{n})\mathbf{n} + (\mathbf{u}\cdot\mathbf{t})\mathbf{t}$$
$$= 2\left((\mathbf{w}-\mathbf{u})\cdot\mathbf{n}\right)\mathbf{n} + \mathbf{u}$$



Closest point transform algorithm

The signed distance φ to a surface $\mathcal{I}$ satisfies the eikonal equation

$$|\nabla\varphi| = 1 \quad \text{with} \quad \varphi\big|_{\mathcal{I}} = 0$$

Closest point transform algorithm

The signed distance φ to a surface $\mathcal{I}$ satisfies the eikonal equation

$$|\nabla\varphi| = 1 \quad \text{with} \quad \varphi\big|_{\mathcal{I}} = 0$$

Distance computation trivial for non-overlapping elementary shapes but difficult to do efficiently for triangulated surface meshes

- Geometric solution approach with plosest-point-transform (CPT) algorithm [Mauch, 2003]

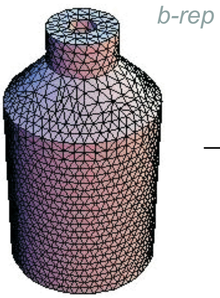
Closest point transform algorithm

The signed distance φ to a surface $\mathcal{I}$ satisfies the eikonal equation

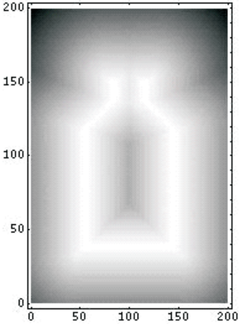
$$|\nabla\varphi| = 1 \quad \text{with} \quad \varphi\big|_{\mathcal{I}} = 0$$

Distance computation trivial for non-overlapping elementary shapes but difficult to do efficiently for triangulated surface meshes

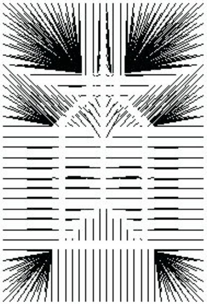
- Geometric solution approach with plosest-point-transform (CPT) algorithm [Mauch, 2003]



Surface mesh $\mathcal{I}$



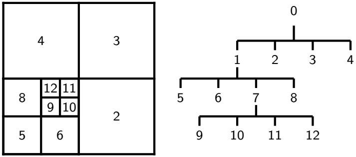
Distance φ



Normal to closest point

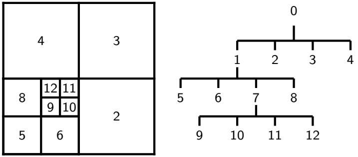
Structured mesh refinement techniques

- Block-based data of equal size
- Block stored in a quad-tree



Structured mesh refinement techniques

- Block-based data of equal size
- Block stored in a quad-tree
- Time-step refinement
- Global index coordinate system
- Neighborhoods need not be stored



Structured mesh refinement techniques

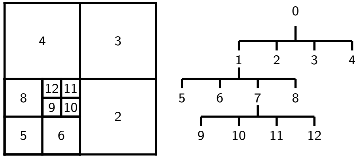
- Block-based data of equal size
- Block stored in a quad-tree
- Time-step refinement
- Global index coordinate system
- Neighborhoods need not be stored

\+ Easy to implement

\+ Simple load-balancing

\+ Parent/Child relations according to tree

\- Larger stencil for higher-order schemes are major problem for cell-based AMR



Structured mesh refinement techniques

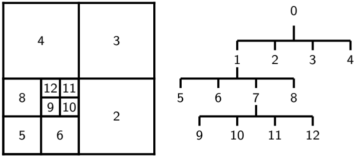
- Block-based data of equal size
- Block stored in a quad-tree
- Time-step refinement
- Global index coordinate system
- Neighborhoods need not be stored

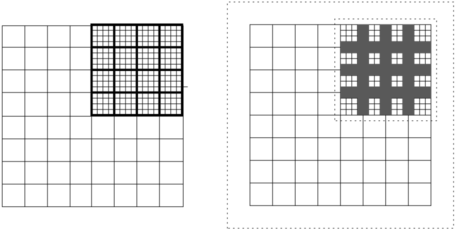
\+ Easy to implement

\+ Simple load-balancing

\+ Parent/Child relations according to tree

\- Larger stencil for higher-order schemes are major problem for cell-based AMR

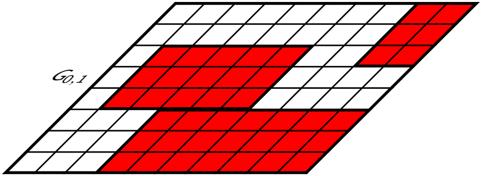




Wasted boundary space in a quad-tree

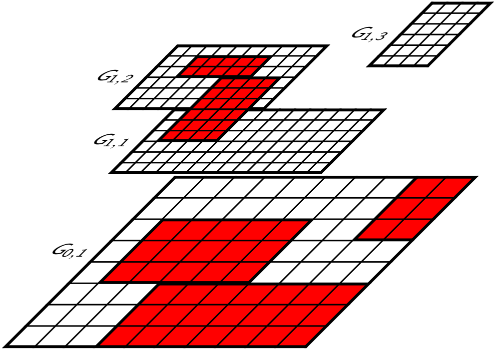
Block-structured adaptive mesh refinement (SAMR)

- Refined blocks overlay coarser ones



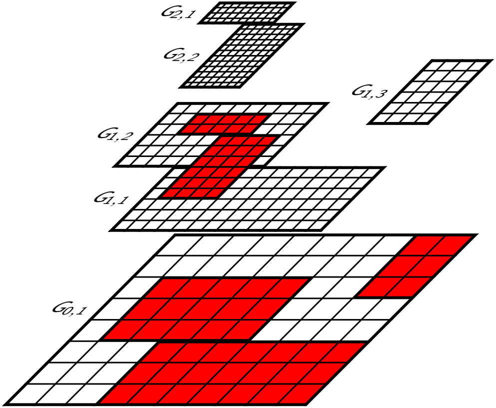
Block-structured adaptive mesh refinement (SAMR)

- Refined blocks overlay coarser ones



Block-structured adaptive mesh refinement (SAMR)

- Refined blocks overlay coarser ones

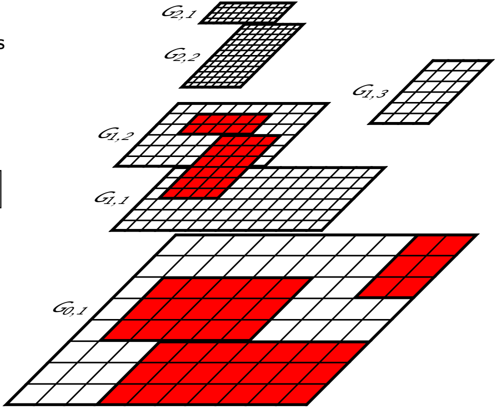


Block-structured adaptive mesh refinement (SAMR)

- Refined blocks overlay coarser ones
- Refinement in space *and time* by factor r_l [Berger and Colella, 1988]
- Block (aka patch) based data structures
+ Numerical scheme (here finite volume)

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

only for single patch necessary



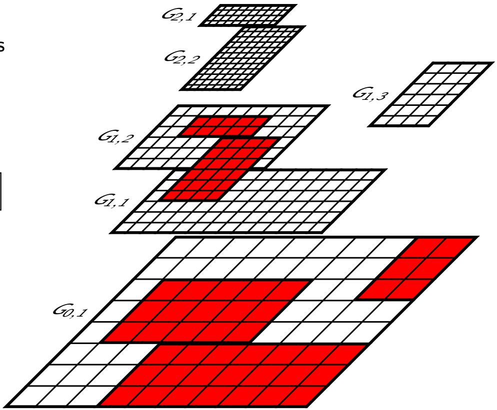
Block-structured adaptive mesh refinement (SAMR)

- Refined blocks overlay coarser ones
- Refinement in space *and time* by factor r_l [Berger and Colella, 1988]
- Block (aka patch) based data structures
- \+ Numerical scheme (here finite volume)

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

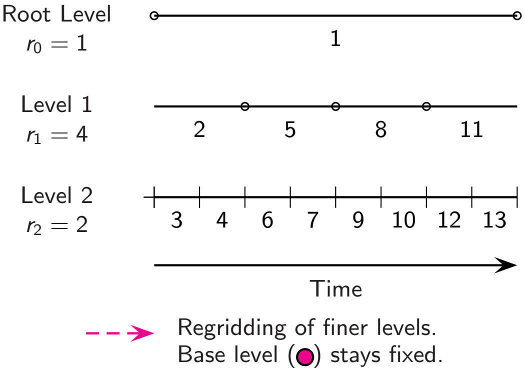
only for single patch necessary

- \+ Efficient cache-reuse / vectorization possible
- \- Cells without mark are refined
- \- Cluster-algorithm necessary
- \- Difficult to implement



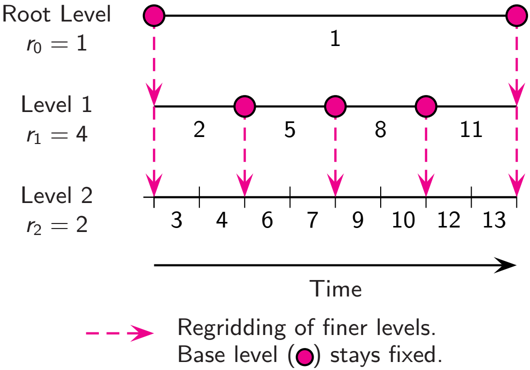
Recursive integration order

- Space-time interpolation to create data at refinement boundaries



Recursive integration order

- Space-time interpolation to create data at refinement boundaries
- Regridding:
  - Creation of new grids, copy existing cells on level $l > 0$
  - Spatial interpolation to initialize new cells on level $l > 0$



Parallelization strategies

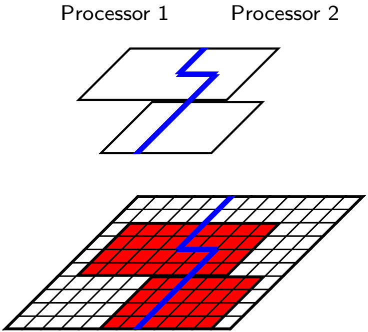
Decomposition of the hierarchical data

- Distribution of each grid
- Separate distribution of each level, cf. [Rendleman et al., 2000]
- Rigorous domain decomposition

Parallelization strategies

Decomposition of the hierarchical data

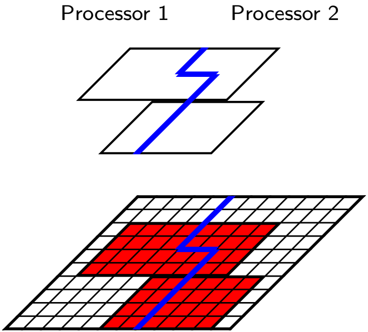
- Distribution of each grid
- Separate distribution of each level, cf. [Rendleman et al., 2000]
- Rigorous domain decomposition
  - Data of all levels resides on same node
  - Grid hierarchy defines unique "floor-plan"



Parallelization strategies

Decomposition of the hierarchical data

- Distribution of each grid
- Separate distribution of each level, cf. [Rendleman et al., 2000]
- Rigorous domain decomposition
  - Data of all levels resides on same node
  - Grid hierarchy defines unique "floor-plan"
  - Redistribution of data blocks during reorganization of hierarchical data
  - Synchronization when setting ghost cells

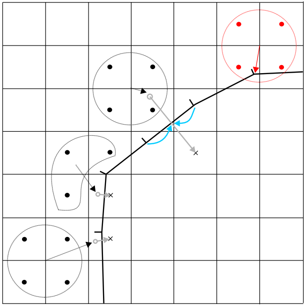


Construction of coupling data

- Moving boundary/interface is treated as a moving contact discontinuity and represented by level set
- Construction of level set from triangulated surface data with CPT algorithm

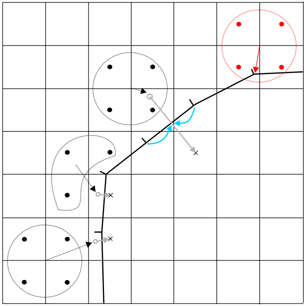
Construction of coupling data

- Moving boundary/interface is treated as a moving contact discontinuity and represented by level set
- Construction of level set from triangulated surface data with CPT algorithm
- One-sided construction of mirrored ghost cell and new FEM nodal point values



Construction of coupling data

- Moving boundary/interface is treated as a moving contact discontinuity and represented by level set
- Construction of level set from triangulated surface data with CPT algorithm
- One-sided construction of mirrored ghost cell and new FEM nodal point values
- Explicit coupling possible if geometry and velocities are prescribed for the more compressible medium



$u^F := u^S(t)|_{\mathcal{I}}$
UpdateFluid(Δt)
$\sigma^S_{nm} := \sigma^F_{nm}(t+\Delta t)|_{\mathcal{I}}$
UpdateSolid(Δt)
$t := t + \Delta t$

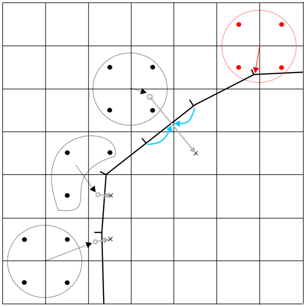
Coupling conditions on interface
Viscous fluid:

$$\left.\begin{aligned} u^S &= u^F \\ \sigma^S_{nm} &= \sigma^F_{nm} \end{aligned}\right|_{\mathcal{I}}$$

with $\sigma^F_{nm} = -p^F\delta_{nm} + \Sigma^F_{nm}$

Construction of coupling data

- Moving boundary/interface is treated as a moving contact discontinuity and represented by level set
- Construction of level set from triangulated surface data with CPT algorithm
- One-sided construction of mirrored ghost cell and new FEM nodal point values
- Explicit coupling possible if geometry and velocities are prescribed for the more compressible medium



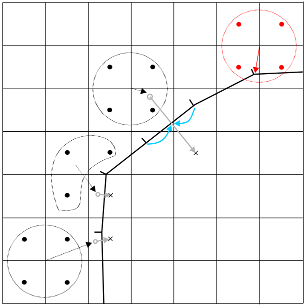
$u_n^F := u_n^S(t)|_{\mathcal{I}}$
`UpdateFluid(` Δt `)`
$\sigma_{nm}^S := -p^F(t+\Delta t)\delta_{nm}|_{\mathcal{I}}$
`UpdateSolid(` Δt `)`
$t := t + \Delta t$

Coupling conditions on interface
Inviscid fluid:

$$\left.\begin{array}{rcl} u_n^S & = & u_n^F \\ \sigma_{nm}^S & = & -p^F\delta_{nm} \end{array}\right|_{\mathcal{I}}$$

Construction of coupling data

- Moving boundary/interface is treated as a moving contact discontinuity and represented by level set
- Construction of level set from triangulated surface data with CPT algorithm
- One-sided construction of mirrored ghost cell and new FEM nodal point values
- Explicit coupling possible if geometry and velocities are prescribed for the more compressible medium



$$u_n^F := u_n^S(t)|_{\mathcal{I}} \qquad \sigma_{nm}^S := -p^F(t)\delta_{nm}|_{\mathcal{I}}$$

`UpdateFluid(` Δt `)` `UpdateSolid(` Δt `)`

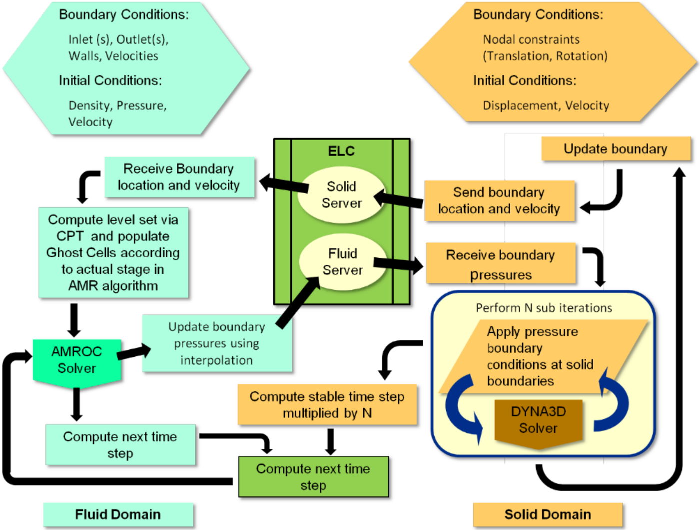
$$t := t + \Delta t$$

[Deiterding and Wood, 2013]

Coupling conditions on interface
Inviscid fluid:

$$\left.\begin{array}{rcl} u_n^S & = & u_n^F \\ \sigma_{nm}^S & = & -p^F\delta_{nm} \end{array}\right|_{\mathcal{I}}$$

Coupling elements



AMROC framework and most important solvers

- Implements described algorithms and facilitates easy exchange of the block-based numerical scheme
- Altogether $\sim 500,000$ LOC in C++, C, Fortran-77, Fortran-90
- Templatized SAMR kernel $\sim 50,000$ lines of code (LOC)
- V2.0 plus FSI coupling routines as open source at http://www.vtf.website
- Used here V3.0 with enhanced parallelization (V2.1 not released)

AMROC framework and most important solvers

- Implements described algorithms and facilitates easy exchange of the block-based numerical scheme
- Altogether $\sim 500,000$ LOC in C++, C, Fortran-77, Fortran-90
- Templatized SAMR kernel $\sim 50,000$ lines of code (LOC)
- V2.0 plus FSI coupling routines as open source at http://www.vtf.website
- Used here V3.0 with enhanced parallelization (V2.1 not released)
- Scientifically most relevant patch solvers:
  - Ideal gas dynamics: various 2nd order methods
  - Hybrid WENO methods for LES and DNS

AMROC framework and most important solvers

- Implements described algorithms and facilitates easy exchange of the block-based numerical scheme
- Altogether $\sim 500,000$ LOC in C++, C, Fortran-77, Fortran-90
- Templatized SAMR kernel $\sim 50,000$ lines of code (LOC)
- V2.0 plus FSI coupling routines as open source at http://www.vtf.website
- Used here V3.0 with enhanced parallelization (V2.1 not released)
- Scientifically most relevant patch solvers:
  - Ideal gas dynamics: various 2nd order methods
  - Hybrid WENO methods for LES and DNS
  - Shock-induced combustion with detailed chemistry (uses Chemkin 2)
  - Two-temperature model for non-equilibrium hypersonics
  - Compressible multi-phase flows

AMROC framework and most important solvers

- Implements described algorithms and facilitates easy exchange of the block-based numerical scheme
- Altogether $\sim 500,000$ LOC in C++, C, Fortran-77, Fortran-90
- Templatized SAMR kernel $\sim 50,000$ lines of code (LOC)
- V2.0 plus FSI coupling routines as open source at http://www.vtf.website
- Used here V3.0 with enhanced parallelization (V2.1 not released)
- Scientifically most relevant patch solvers:
  - Ideal gas dynamics: various 2nd order methods
  - Hybrid WENO methods for LES and DNS
  - Shock-induced combustion with detailed chemistry (uses Chemkin 2)
  - Two-temperature model for non-equilibrium hypersonics
  - Compressible multi-phase flows
  - Ideal magneto-hydrodynamics
  - Level-set method for Eulerian solid mechanics
  - Lattice Boltzmann method for LES

AMROC framework and most important solvers

- Implements described algorithms and facilitates easy exchange of the block-based numerical scheme
- Altogether $\sim 500,000$ LOC in C++, C, Fortran-77, Fortran-90
- Templatized SAMR kernel $\sim 50,000$ lines of code (LOC)
- V2.0 plus FSI coupling routines as open source at http://www.vtf.website
- Used here V3.0 with enhanced parallelization (V2.1 not released)
- Scientifically most relevant patch solvers:
  - Ideal gas dynamics: various 2nd order methods
  - Hybrid WENO methods for LES and DNS
  - Shock-induced combustion with detailed chemistry (uses Chemkin 2)
  - Two-temperature model for non-equilibrium hypersonics
  - Compressible multi-phase flows
  - Ideal magneto-hydrodynamics
  - Level-set method for Eulerian solid mechanics
  - Lattice Boltzmann method for LES
- Structural solvers:
  - SFC thin shell solver
  - Adlib volumetric FEM solver (not released)
  - Dyna3d (requires DOE license)

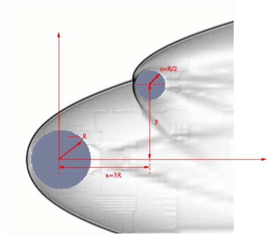
Proximal bodies in hypersonic flow

Flow modeled by Euler equations for a single ideal gas with $p = (\gamma - 1)\,\rho e$

$$\partial_t \rho + \partial_{x_n}(\rho u_n) = 0\ , \quad \partial_t(\rho u_k) + \partial_{x_n}(\rho u_k u_n + \delta_{kn} p) = 0\ , \quad \partial_t(\rho E) + \partial_{x_n}(u_n(\rho E + p)) = 0$$

Numerical approximation with

- Finite volume flux-vector splitting scheme with MUSCL reconstruction, dimensional splitting



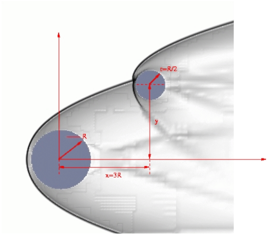
Proximal bodies in hypersonic flow

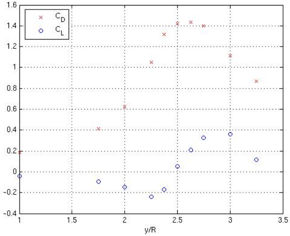
Flow modeled by Euler equations for a single ideal gas with $p = (\gamma - 1)\,\rho e$

$$\partial_t \rho + \partial_{x_n}(\rho u_n) = 0 \ , \quad \partial_t(\rho u_k) + \partial_{x_n}(\rho u_k u_n + \delta_{kn} p) = 0 \ , \quad \partial_t(\rho E) + \partial_{x_n}(u_n(\rho E + p)) = 0$$

Numerical approximation with

- Finite volume flux-vector splitting scheme with MUSCL reconstruction, dimensional splitting
- Spherical bodies, force computation with overlaid lattitude-longitude mesh to obtain drag and lift coefficients $C_{D,L} = \dfrac{2F_{D,L}}{\rho v^2 \pi r^2}$
- inflow $M = 10$, C_D and C_L on secondary sphere, lateral position varied, no motion





Verification and validation

Static force measurements, $M = 10$:
[Laurence et al., 2007]

- Refinement study: $40 \times 40 \times 32$ base grid , up to without AMR up to $\sim 209.7 \cdot 10^6$ cells, largest run $\sim 35,000\,\mathrm{h}$ CPU

$l_{\max}$	C_D	ΔC_D	C_L	ΔC_L
1	1.264		-0.176	
2	1.442	0.178	-0.019	0.157
3	1.423	-0.019	0.052	0.071
4	1.408	-0.015	0.087	0.035

Verification and validation

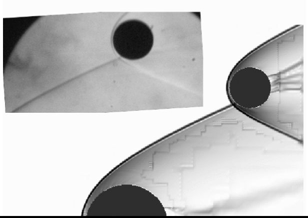
Static force measurements, $M = 10$:
[Laurence et al., 2007]

- Refinement study: $40 \times 40 \times 32$ base grid , up to without AMR up to $\sim 209.7 \cdot 10^6$ cells, largest run $\sim 35,000\,\mathrm{h}$ CPU

$l_{\max}$	C_D	ΔC_D	C_L	ΔC_L
1	1.264		-0.176	
2	1.442	0.178	-0.019	0.157
3	1.423	-0.019	0.052	0.071
4	1.408	-0.015	0.087	0.035

- Comparison with experimental results: 3 additional levels, $\sim 2000\,\mathrm{h}$ CPU

	Experimental	Computational
C_D	1.11 ± 0.08	1.01
C_L	0.29 ± 0.05	0.28



Verification and validation

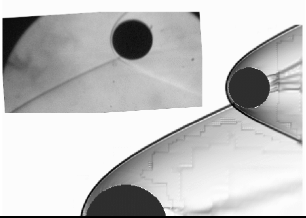
Static force measurements, $M = 10$:
[Laurence et al., 2007]

- Refinement study: $40 \times 40 \times 32$ base grid , up to without AMR up to $\sim 209.7 \cdot 10^6$ cells, largest run $\sim 35,000\,\mathrm{h}$ CPU

$l_{\max}$	C_D	ΔC_D	C_L	ΔC_L
1	1.264		-0.176	
2	1.442	0.178	-0.019	0.157
3	1.423	-0.019	0.052	0.071
4	1.408	-0.015	0.087	0.035

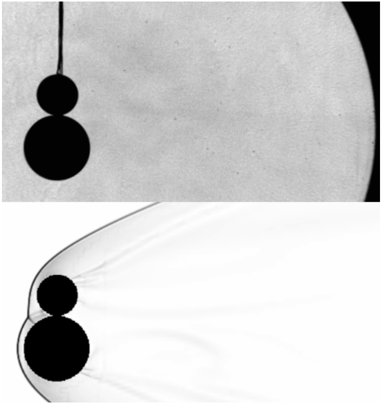
- Comparison with experimental results: 3 additional levels, $\sim 2000\,\mathrm{h}$ CPU

	Experimental	Computational
C_D	1.11 ± 0.08	1.01
C_L	0.29 ± 0.05	0.28



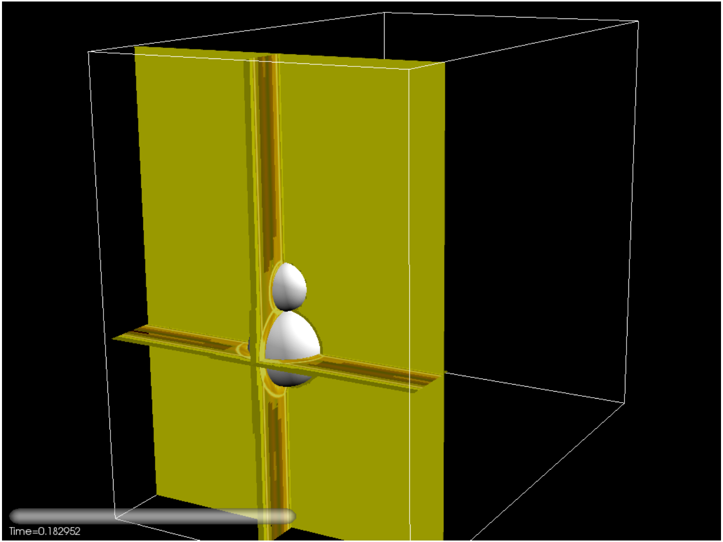
Dynamic motion, $M = 4$:

- Base grid $150 \times 125 \times 90$, two additional levels with factor 2
- 24,704 time steps, $36,808\,\mathrm{h}$ CPU on 256 cores IBM BG/P



[Laurence and Deiterding, 2011]

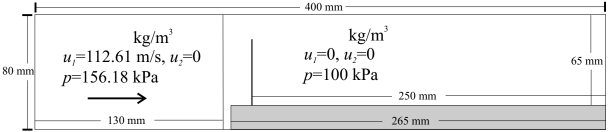
Schlieren graphics on refinement regions



Shock-driven elastic panel motion

Test case with shock at $M = 1.21$ suggested by [Giordano et al., 2005]

- Forward facing step geometry, fixed walls everywhere except at inflow

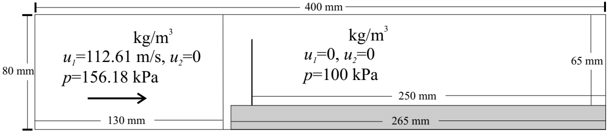


- SAMR base mesh $320 \times 64(\times 2)$, 2 additional levels with factor 2

Shock-driven elastic panel motion

Test case with shock at $M = 1.21$ suggested by [Giordano et al., 2005]

- Forward facing step geometry, fixed walls everywhere except at inflow

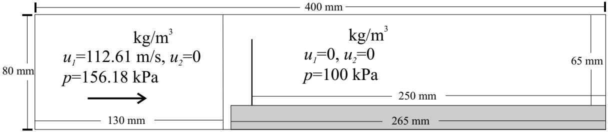


- SAMR base mesh $320 \times 64(\times 2)$, 2 additional levels with factor 2
- Intel 3.4GHz Xeon dual processors, GB Ethernet interconnect
  - $\sim 450\,\mathrm{h}$ CPU on 15 fluid CPU + 1 solid CPU for DYNA3D [Hallquist and Lin, 2005]

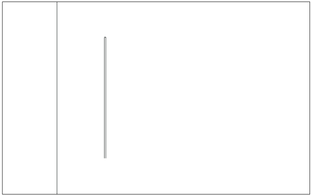
Shock-driven elastic panel motion

Test case with shock at $M = 1.21$ suggested by [Giordano et al., 2005]

- Forward facing step geometry, fixed walls everywhere except at inflow



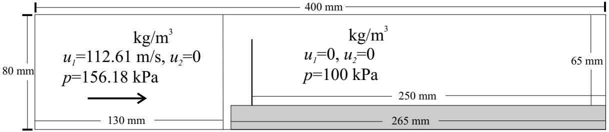
- SAMR base mesh $320 \times 64(\times 2)$, 2 additional levels with factor 2
- Intel 3.4GHz Xeon dual processors, GB Ethernet interconnect
  - $\sim$ 450 h CPU on 15 fluid CPU + 1 solid CPU for DYNA3D [Hallquist and Lin, 2005]



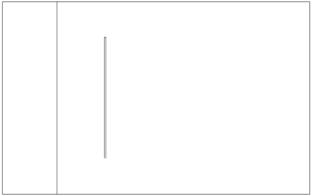
Shock-driven elastic panel motion

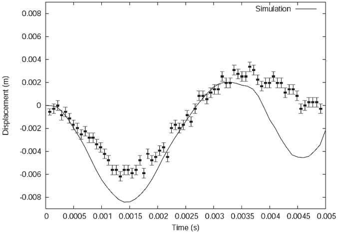
Test case with shock at $M = 1.21$ suggested by [Giordano et al., 2005]

- Forward facing step geometry, fixed walls everywhere except at inflow



- SAMR base mesh $320 \times 64(\times 2)$, 2 additional levels with factor 2
- Intel 3.4GHz Xeon dual processors, GB Ethernet interconnect
  - $\sim 450\,\mathrm{h}$ CPU on 15 fluid CPU + 1 solid CPU for DYNA3D [Hallquist and Lin, 2005]





Tip displacement in simulation and experiment

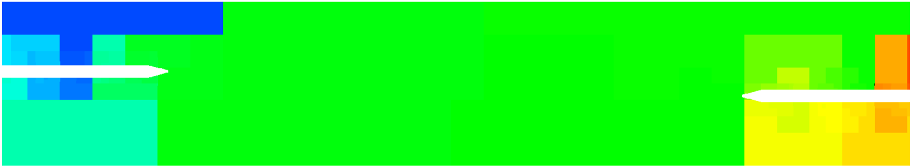
Train aerodynamics of NGT2 prototype

- Next Generation Train 2 (NGT2) geometry by the German Aerospace Centre (DLR) [Fragner and Deiterding, 2017]
- Mirrored train head of length $\sim 60\,\mathrm{m}$, no wheels or tracks, train models $0.17\,\mathrm{m}$ above ground above the ground level.
- Train velocities $100\,\mathrm{m/s}$ and $-100\,\mathrm{m/s}$, middle axis $6\,\mathrm{m}$ apart, initial distance between centers $200\,\mathrm{m}$
- Base mesh of $360 \times 40 \times 30$ for domain of $360\,\mathrm{m} \times 40\,\mathrm{m} \times 30\,\mathrm{m}$
- Two/three additional levels, refined by factor 2. Refinement based on pressure gradient and level set. Parallel redistribution at every level-0 time step.
- On 96 cores Intel Xeon E5-2670 $2.6\,\mathrm{GHz}$ a final $t_e = 3\,\mathrm{sec}$ was reached after $12,385\,\mathrm{sec}$ / $43,395\,\mathrm{sec}$ wall time, i.e., $330\,\mathrm{h}$ and $1157\,\mathrm{h}$ CPU

Passing in open space – AMR and dynamic distribution

Domains of three-level refinement

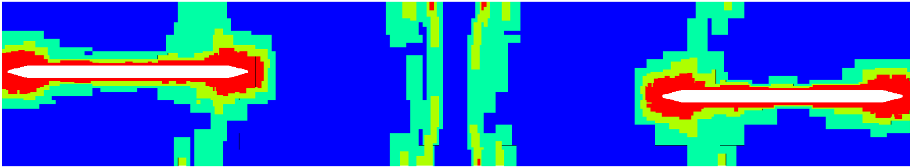
Distribution to 96 processors



Enlargement of domain center shown

Passing in open space – AMR and dynamic distribution

Domains of three-level refinement



Distribution to 96 processors

Enlargement of domain center shown

Passing in open space – AMR and dynamic distribution

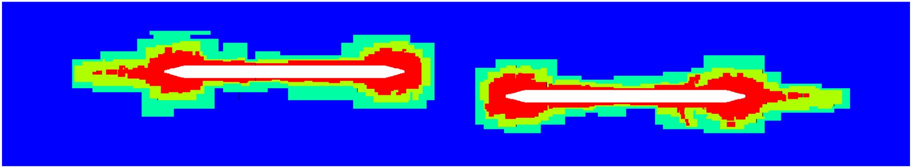
Domains of three-level refinement

Distribution to 96 processors

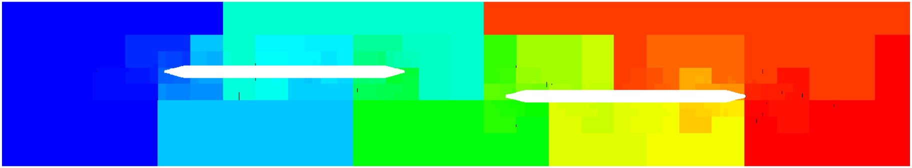
Enlargement of domain center shown

Passing in open space – AMR and dynamic distribution

Domains of three-level refinement



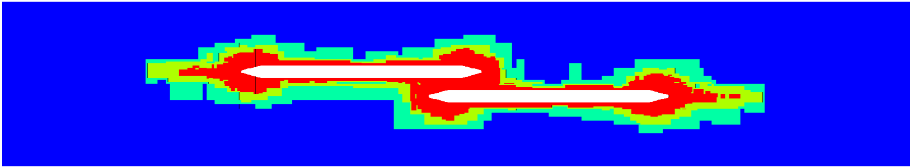
Distribution to 96 processors



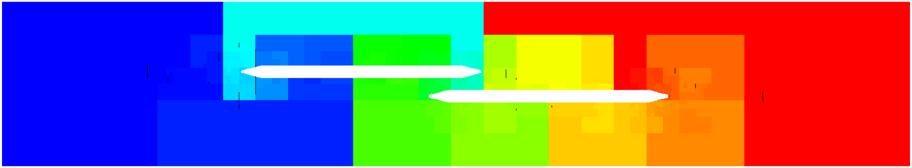
Enlargement of domain center shown

Passing in open space – AMR and dynamic distribution

Domains of three-level refinement



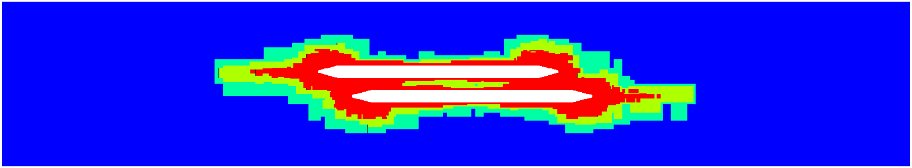
Distribution to 96 processors



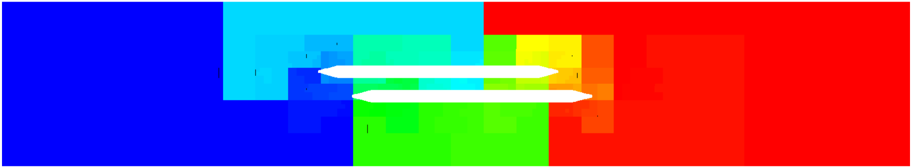
Enlargement of domain center shown

Passing in open space – AMR and dynamic distribution

Domains of three-level refinement



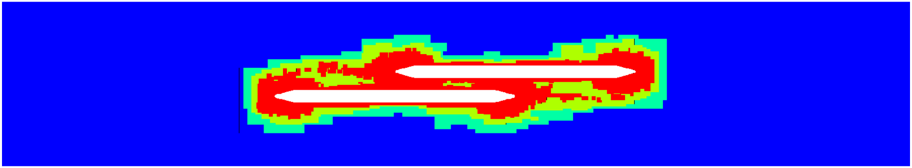
Distribution to 96 processors



Enlargement of domain center shown

Passing in open space – AMR and dynamic distribution

Domains of three-level refinement

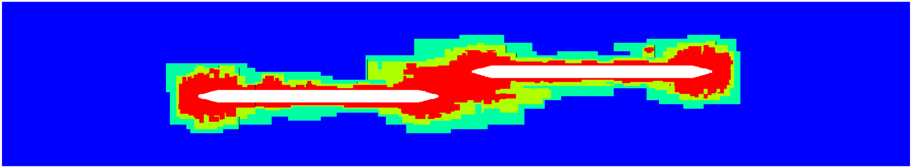


Distribution to 96 processors

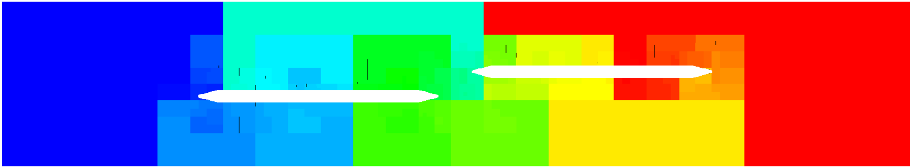
Enlargement of domain center shown

Passing in open space – AMR and dynamic distribution

Domains of three-level refinement



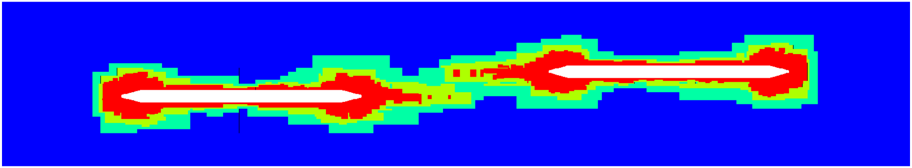
Distribution to 96 processors



Enlargement of domain center shown

Passing in open space – AMR and dynamic distribution

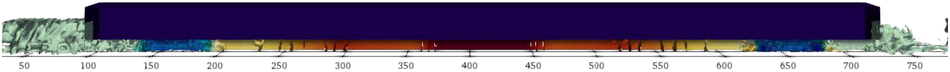
Domains of three-level refinement

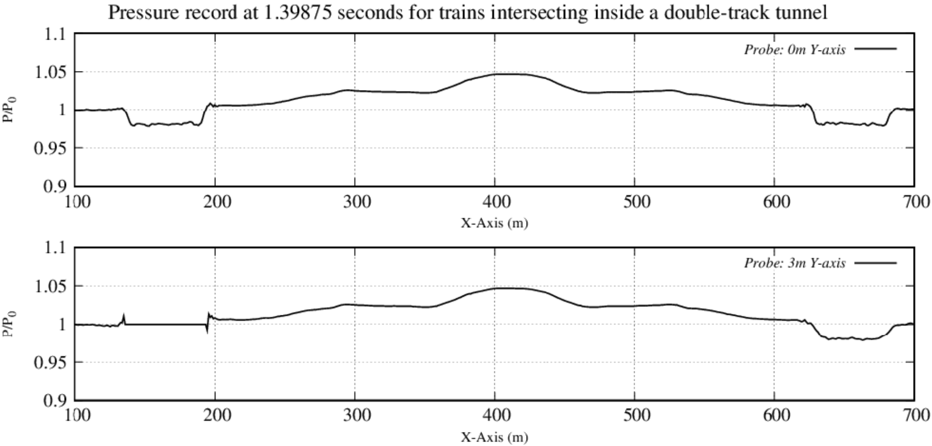


Distribution to 96 processors

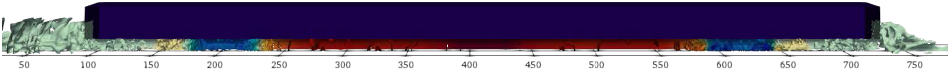
Enlargement of domain center shown

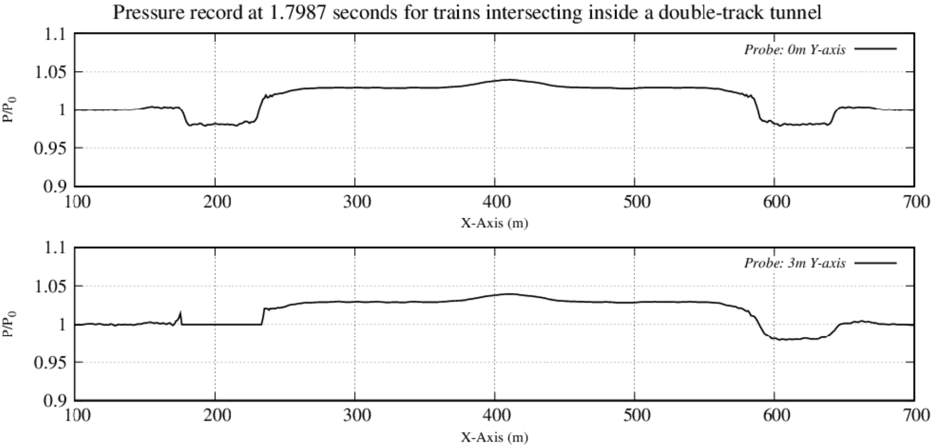
Pressure transects



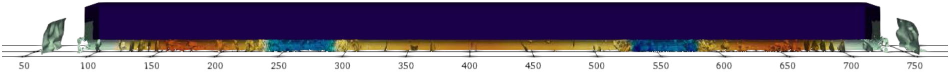


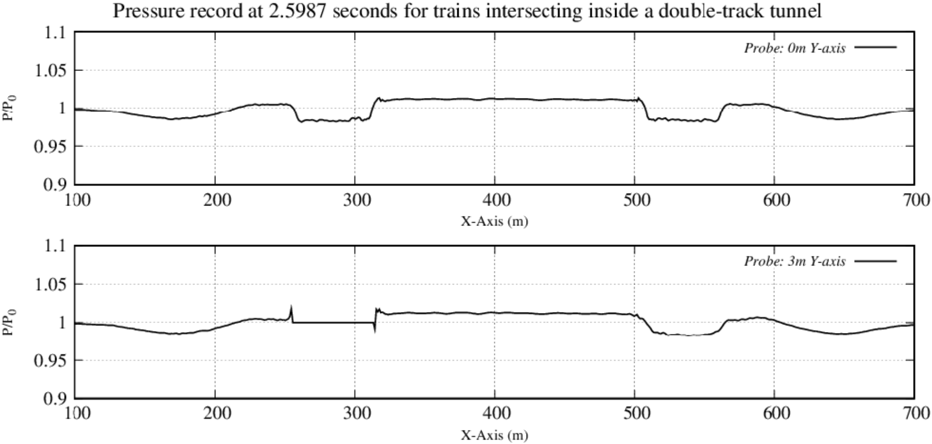
Pressure transects



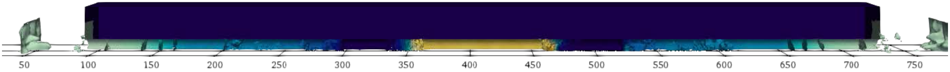


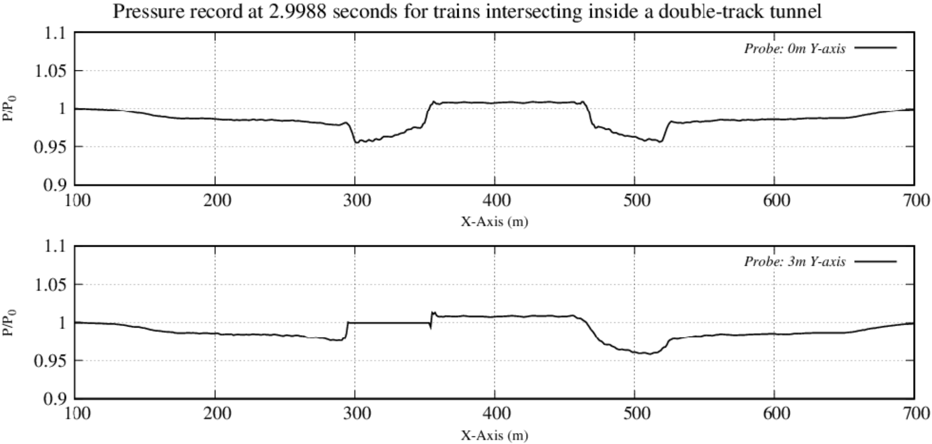
Pressure transects



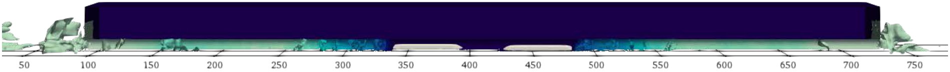


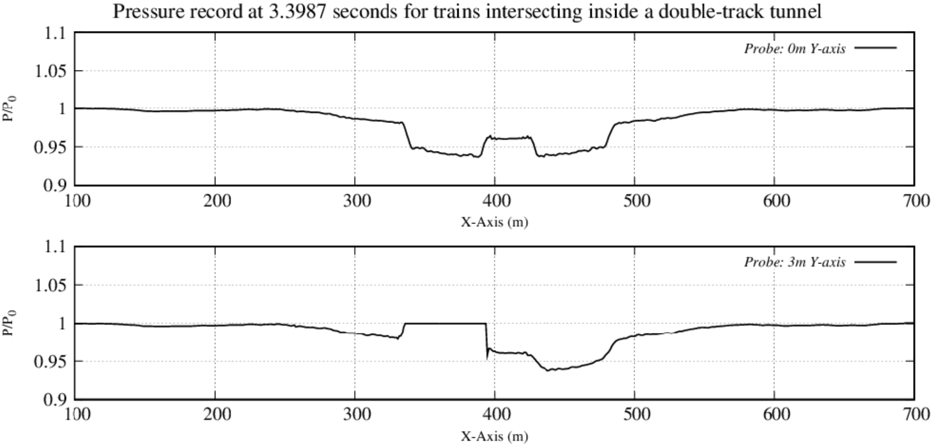
Pressure transects





Pressure transects



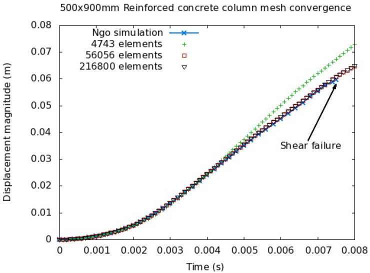


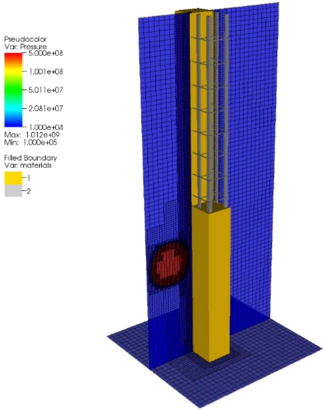
Plastic deformation of reinforced concrete column

- Column of 6.4 m and $500 \times 900\,\mathrm{mm}$ cross-section as in [Ngo et al., 2007]
- DYNA3D elastic-plastic concrete model: strength $\sigma_{\mathrm{max}} = 80\,\mathrm{MPa}$
  - $\rho_s = 2010\,\mathrm{kg/m}^3$, $E = 21.72\,\mathrm{GPa}$, $\nu = 0.2$, yield stress $\sigma_y = 910\,\mathrm{kPa}$, $E_T = 11.2\,\mathrm{GPa}$, $\beta = 0.03$
- Spherical energy deposition $\equiv 150\,\mathrm{kg}$ TNT, $0.5\,\mathrm{m}$ distance, $2\,\mathrm{m}$ above the ground
- $297\,\mathrm{h}$ CPU on 33+1 CPU 3.4 GHz Intel-Xeon

Plastic deformation of reinforced concrete column

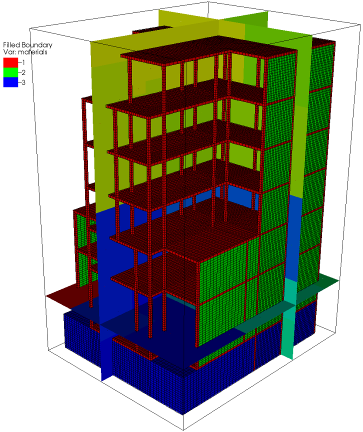
- Column of 6.4 m and $500 \times 900\,\mathrm{mm}$ cross-section as in [Ngo et al., 2007]
- DYNA3D elastic-plastic concrete model: strength $\sigma_{\text{max}} = 80\,\mathrm{MPa}$
  - $\rho_s = 2010\,\mathrm{kg/m^3}$, $E = 21.72\,\mathrm{GPa}$, $\nu = 0.2$, yield stress $\sigma_y = 910\,\mathrm{kPa}$, $E_T = 11.2\,\mathrm{GPa}$, $\beta = 0.03$
- Spherical energy deposition $\equiv 150\,\mathrm{kg}$ TNT, 0.5 m distance, 2 m above the ground
- 297 h CPU on 33+1 CPU 3.4 GHz Intel-Xeon





Blast explosion in a multistory building

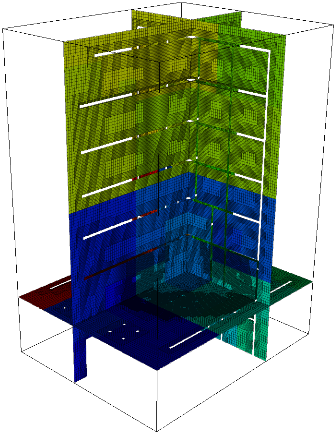
- $20\,\mathrm{m} \times 40\,\mathrm{m} \times 25\,\mathrm{m}$ seven-story building similar to [Luccioni et al., 2004]
- Spherical energy deposition $\equiv 400\,\mathrm{kg}$ TNT, $r = 0.5\,\mathrm{m}$ in lobby of building
- SAMR: $80 \times 120 \times 90$ base level, three additional levels with factor 2, FSI coupling at level 1
- Simulation with ground: $1,070$ coupled time steps, $830\,\mathrm{h}$ CPU ($\sim 25.9\,\mathrm{h}$ wall time) on 31+1 cores
- $\sim 8,000,000$ cells instead of $55,296,000$ (uniform)
- $69,709$ hexahedral elements and with material parameters



	ρ_s [kg/m^3]	σ_0 [MPa]	E_T [GPa]	β	K [GPa]	G [GPa]	$\bar{\epsilon}^p$	p_f [MPa]
Columns	2010	50	11.2	1.0	21.72	4.67	0.02	-30
Walls	2010	25	11.2	1.0	6.22	4.67	0.01	-15

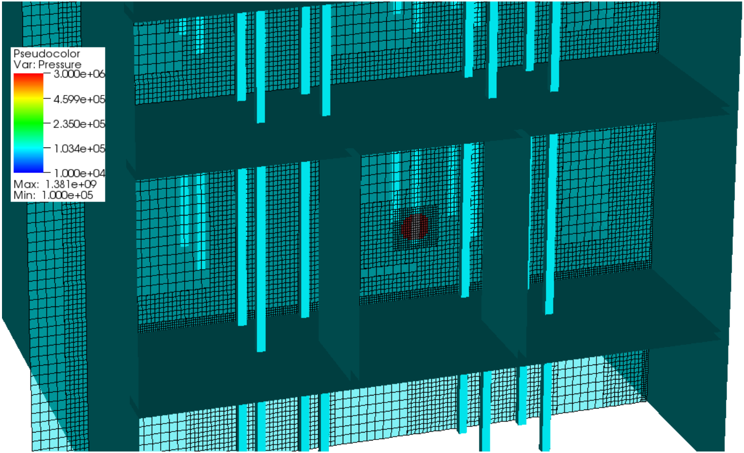
Blast explosion in a multistory building

- $20\,\mathrm{m} \times 40\,\mathrm{m} \times 25\,\mathrm{m}$ seven-story building similar to [Luccioni et al., 2004]
- Spherical energy deposition $\equiv 400\,\mathrm{kg}$ TNT, $r = 0.5\,\mathrm{m}$ in lobby of building
- SAMR: $80 \times 120 \times 90$ base level, three additional levels with factor 2, FSI coupling at level 1
- Simulation with ground: 1,070 coupled time steps, $830\,\mathrm{h}$ CPU ($\sim 25.9\,\mathrm{h}$ wall time) on 31+1 cores
- $\sim 8,000,000$ cells instead of $55,296,000$ (uniform)
- 69,709 hexahedral elements and with material parameters



	ρ_s [kg/m^3]	σ_0 [MPa]	E_T [GPa]	β	K [GPa]	G [GPa]	$\bar{\epsilon}^p$	p_f [MPa]
Columns	2010	50	11.2	1.0	21.72	4.67	0.02	-30
Walls	2010	25	11.2	1.0	6.22	4.67	0.01	-15

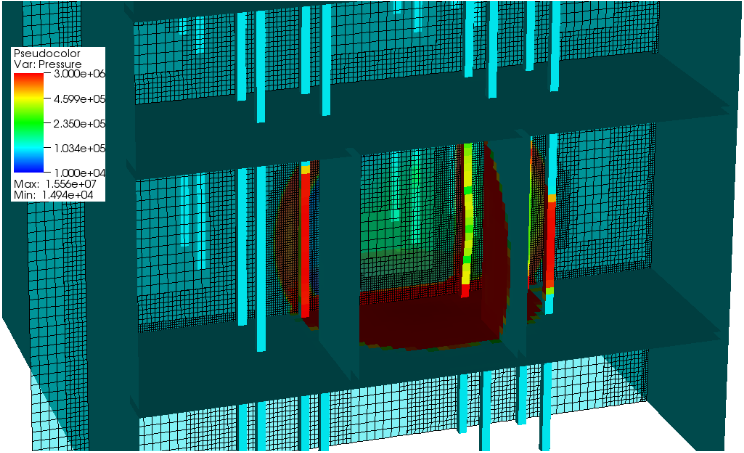
Blast explosion in a multistory building – II



$t = 0$

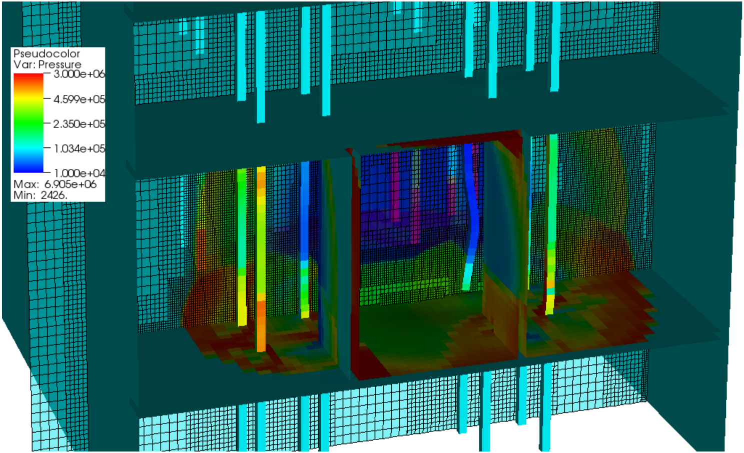
[Deiterding and Wood, 2013]

Blast explosion in a multistory building – II



[Deiterding and Wood, 2013]

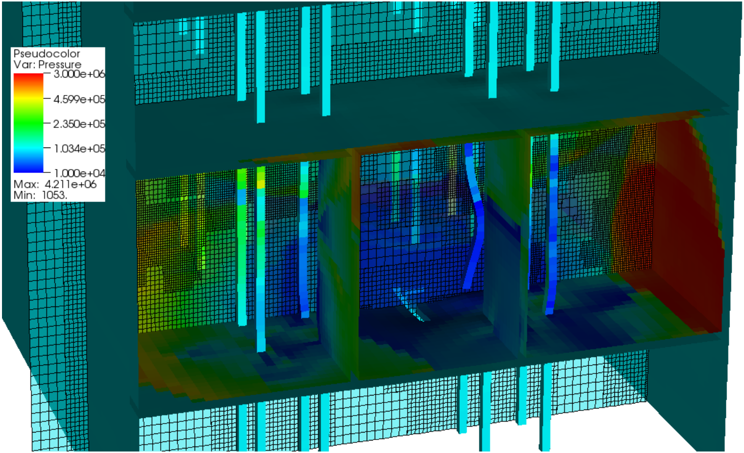
Blast explosion in a multistory building – II



$t = 6.1\,\mathrm{ms}$

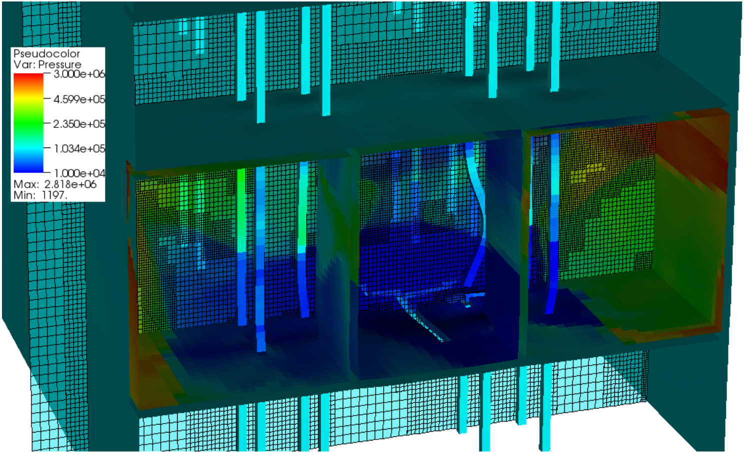
[Deiterding and Wood, 2013]

Blast explosion in a multistory building – II



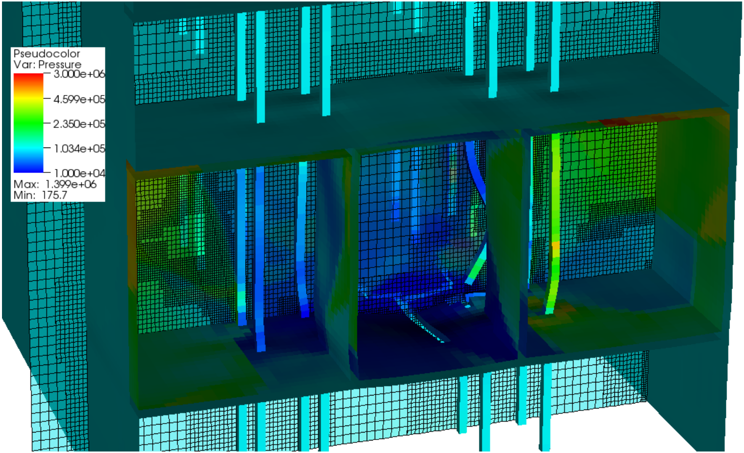
[Deiterding and Wood, 2013]

Blast explosion in a multistory building – II



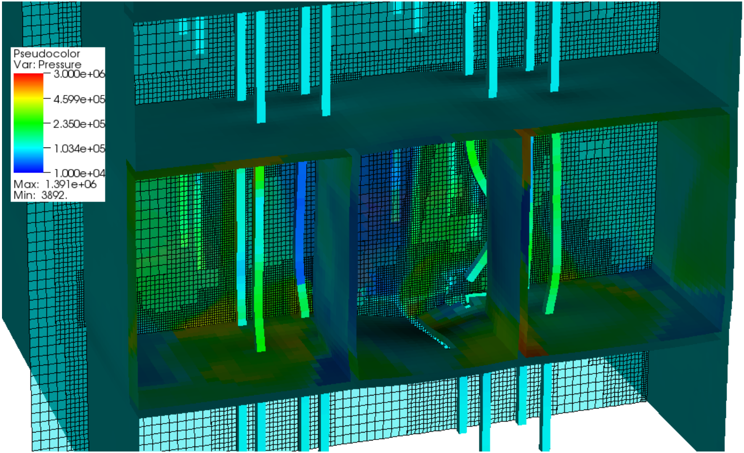
[Deiterding and Wood, 2013]

Blast explosion in a multistory building – II



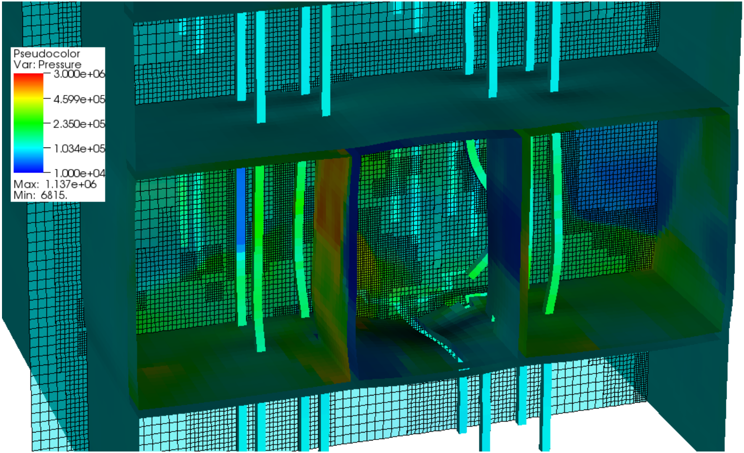
[Deiterding and Wood, 2013]

Blast explosion in a multistory building – II



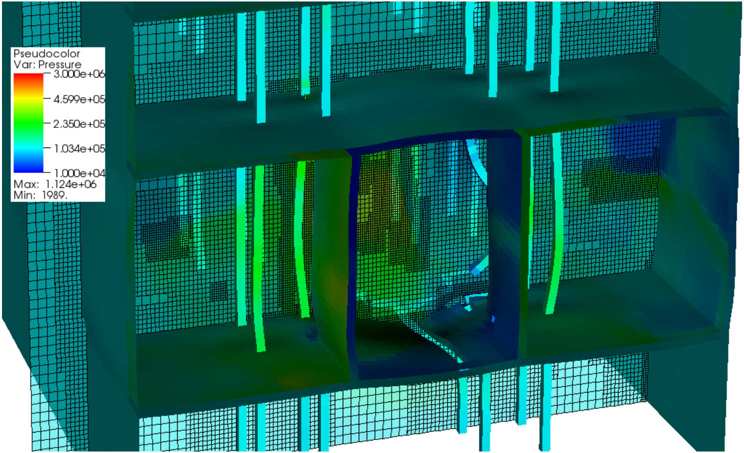
[Deiterding and Wood, 2013]

Blast explosion in a multistory building – II



[Deiterding and Wood, 2013]

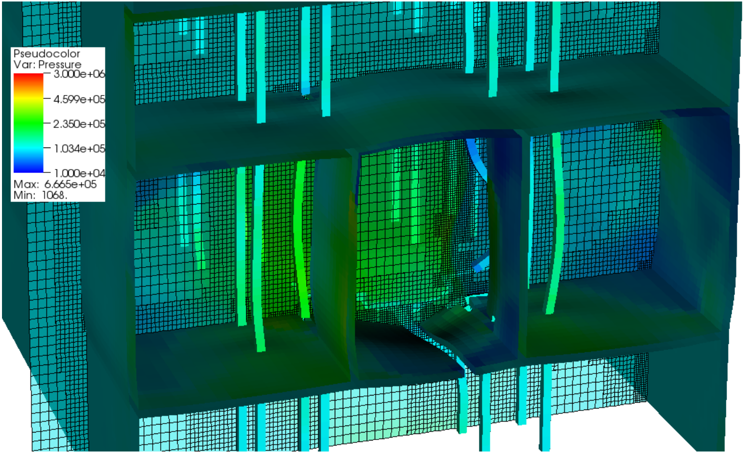
Blast explosion in a multistory building – II



$t = 29.2\,\mathrm{ms}$

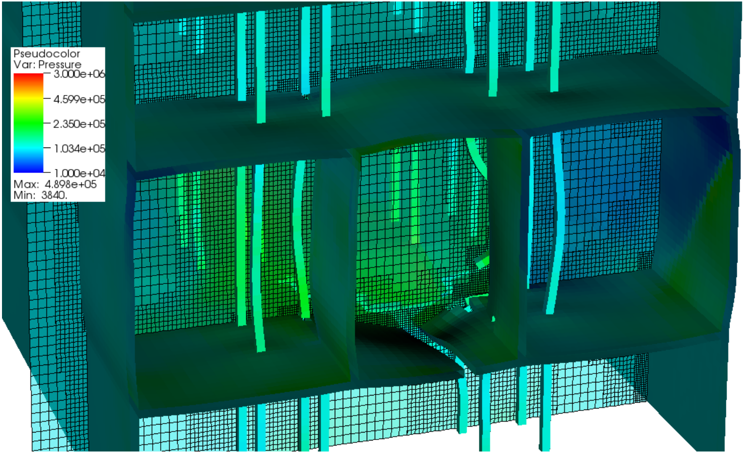
[Deiterding and Wood, 2013]

Blast explosion in a multistory building – II



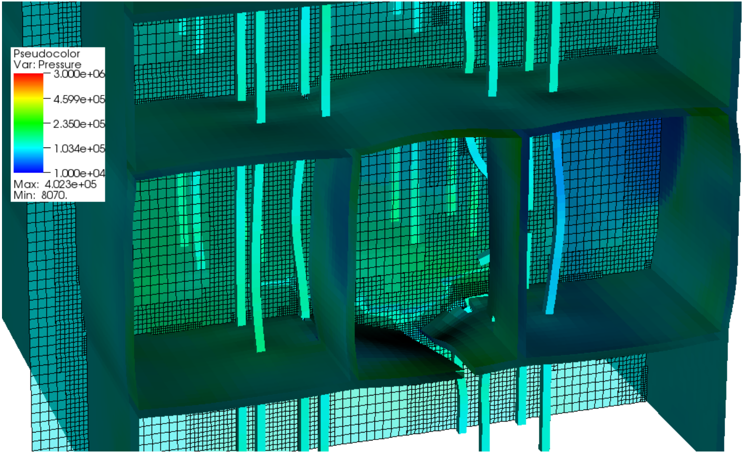
[Deiterding and Wood, 2013]

Blast explosion in a multistory building – II



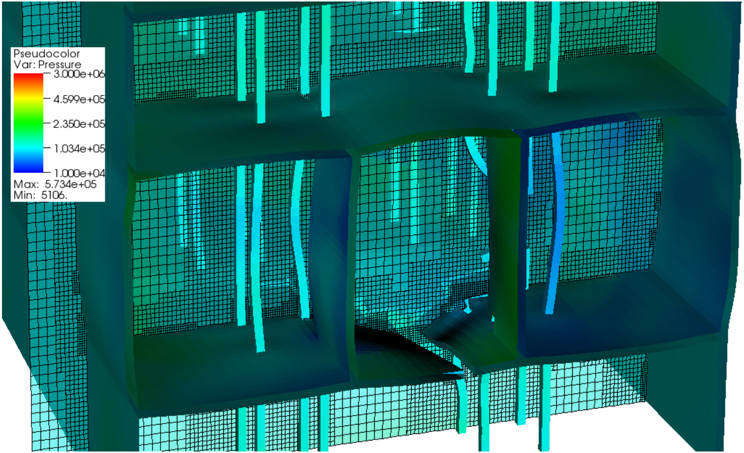
[Deiterding and Wood, 2013]

Blast explosion in a multistory building – II



[Deiterding and Wood, 2013]

Blast explosion in a multistory building – II



$t = 48.7\,\mathrm{ms}$

[Deiterding and Wood, 2013]

Prototypical hydrogen explosion in nuclear reactor

Chapman-Jouguet detonation in hydrogen-air mixture at atmospheric pressure. Euler equations with single exothermic reaction $A \longrightarrow B$

$$\partial_t \rho + \partial_{x_n}(\rho u_n) = 0 \,, \quad \partial_t(\rho u_k) + \partial_{x_n}(\rho u_k u_n + \delta_{kn} p) = 0 \,, k = 1, \ldots, d$$

$$\partial_t(\rho E) + \partial_{x_n}(u_n(\rho E + p)) = 0 \,, \quad \partial_t(Y\rho) + \partial_{x_n}(Y\rho u_n) = \psi$$

with

$$p = (\gamma - 1)(\rho E - \frac{1}{2}\rho u_n u_n - \rho Y q_0) \quad \text{and} \quad \psi = -kY\rho \exp\left(\frac{-E_{\mathrm{A}}\rho}{p}\right)$$

modeled with empirical detonation model by [Mader, 1979]

Prototypical hydrogen explosion in nuclear reactor

Chapman-Jouguet detonation in hydrogen-air mixture at atmospheric pressure. Euler equations with single exothermic reaction $A \longrightarrow B$

$$\partial_t \rho + \partial_{x_n}(\rho u_n) = 0 \;, \quad \partial_t(\rho u_k) + \partial_{x_n}(\rho u_k u_n + \delta_{kn} p) = 0 \;, k = 1, \dots, d$$

$$\partial_t(\rho E) + \partial_{x_n}(u_n(\rho E + p)) = 0 \;, \quad \partial_t(Y\rho) + \partial_{x_n}(Y\rho u_n) = \psi$$

with

$$p = (\gamma - 1)(\rho E - \frac{1}{2}\rho u_n u_n - \rho Y q_0) \quad \text{and} \quad \psi = -kY\rho \exp\left(\frac{-E_{\mathrm{A}}\rho}{p}\right)$$

modeled with empirical detonation model by [Mader, 1979]

$V := \rho^{-1},\; V_0 := \rho_0^{-1},\; V_{\mathrm{CJ}} := \rho_{\mathrm{CJ}}$
$Y' := 1 - (V - V_0)/(V_{\mathrm{CJ}} - V_0)$
If $0 \le Y' \le 1$ and $Y > 10^{-8}$ then
  If $Y < Y'$ and $Y' < 0.9$ then $Y' := 0$
  If $Y' < 0.99$ then $p' := (1 - Y')p_{\mathrm{CJ}}$
    else $p' := p$
  $\rho_{\mathrm{A}} := Y'\rho$
  $E := p'/(\rho(\gamma - 1)) + Y'q_0 + \frac{1}{2}u_n u_n$

Prototypical hydrogen explosion in nuclear reactor

Chapman-Jouguet detonation in hydrogen-air mixture at atmospheric pressure. Euler equations with single exothermic reaction $A \longrightarrow B$

$$\partial_t \rho + \partial_{x_n}(\rho u_n) = 0 \; , \quad \partial_t(\rho u_k) + \partial_{x_n}(\rho u_k u_n + \delta_{kn} p) = 0 \; , k = 1, \dots, d$$

$$\partial_t(\rho E) + \partial_{x_n}(u_n(\rho E + p)) = 0 \; , \quad \partial_t(Y\rho) + \partial_{x_n}(Y\rho u_n) = \psi$$

with

$$p = (\gamma - 1)(\rho E - \frac{1}{2}\rho u_n u_n - \rho Y q_0) \quad \text{and} \quad \psi = -kY\rho \exp\left(\frac{-E_\mathrm{A}\rho}{p}\right)$$

modeled with empirical detonation model by [Mader, 1979]

$V := \rho^{-1},\ V_0 := \rho_0^{-1},\ V_\mathrm{CJ} := \rho_\mathrm{CJ}$
$Y' := 1 - (V - V_0)/(V_\mathrm{CJ} - V_0)$
If $0 \le Y' \le 1$ and $Y > 10^{-8}$ then
  If $Y < Y'$ and $Y' < 0.9$ then $Y' := 0$
  If $Y' < 0.99$ then $p' := (1 - Y')p_\mathrm{CJ}$
    else $p' := p$
  $\rho_\mathrm{A} := Y'\rho$
  $E := p'/(\rho(\gamma - 1)) + Y' q_0 + \frac{1}{2} u_n u_n$

Used parameters for $\mathrm{H_2}$-Air, stoichiometry 0.5, induction length $3.2\,\mathrm{mm}$, $d_\mathrm{CJ} \approx 1620\,\mathrm{m/s}$

ρ_0	$0.985\,\mathrm{kg/m^3}$
p_0	$100\,\mathrm{kPa}$
ρ_CJ	$1.951\,\mathrm{kg/m^3}$
p_CJ	$1378\,\mathrm{kPa}$
γ	1.266

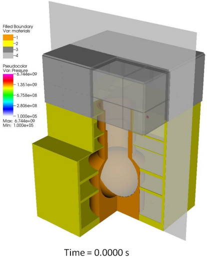
H_2-Air detonation in reactor building

Four materials used

- orange: high strength
- yellow: low strength
- dark gray: concrete, girders
- light gray: paneling

19502 solid hexahedron elements

Exemplary ignition in center plane



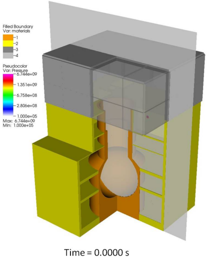
H_2-Air detonation in reactor building

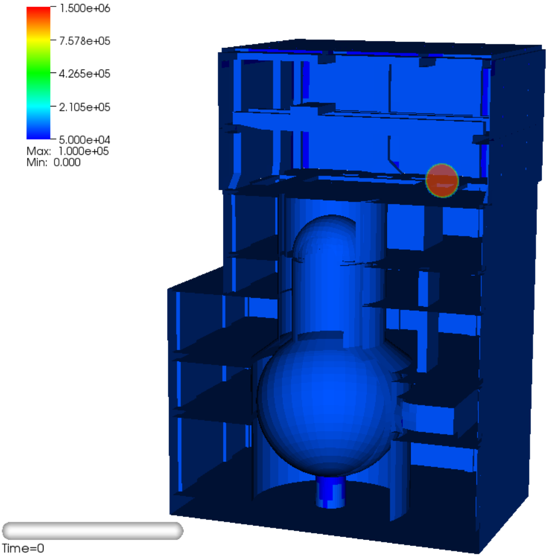
Four materials used

- orange: high strength
- yellow: low strength
- dark gray: concrete, girders
- light gray: paneling

19502 solid hexahedron elements

Exemplary ignition in center plane





Lattice Boltzmann method (LBM)

Instead of solving the Navier-Stokes equations, we use the lattice Boltzmann method.

The LBM is based on solving the Boltzmann equation with a simplified collision operator

$$\partial_t f + \mathbf{u} \cdot \nabla f = \omega(f^{eq} - f) + F$$

- $\mathrm{Kn} = l_f / L \ll 1$, where l_f is replaced with Δx
- Weak compressibilty and small Mach number assumed
- Genuine Cartesian embedded boundary approach

Lattice Boltzmann method (LBM)

Instead of solving the Navier-Stokes equations, we use the lattice Boltzmann method.

The LBM is based on solving the Boltzmann equation with a simplified collision operator

$$\partial_t f + \mathbf{u} \cdot \nabla f = \omega(f^{eq} - f) + F$$

- $\mathrm{Kn} = l_f / L \ll 1$, where l_f is replaced with Δx
- Weak compressibilty and small Mach number assumed
- Genuine Cartesian embedded boundary approach

Equation is approximated with a splitting approach:

1.) Transport step solves $\partial_t f_\alpha + \mathbf{e}_\alpha \cdot \nabla f_\alpha = 0$

2.) Collision step $\partial_t f_\alpha = \omega(f_\alpha^{eq} - f_\alpha) + F_\alpha$

Lattice Boltzmann method (LBM)

Instead of solving the Navier-Stokes equations, we use the lattice Boltzmann method.

The LBM is based on solving the Boltzmann equation with a simplified collision operator

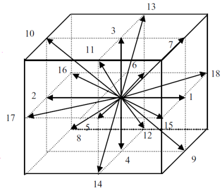
$$\partial_t f + \mathbf{u} \cdot \nabla f = \omega(f^{eq} - f) + F$$

- $\mathrm{Kn} = l_f / L \ll 1$, where l_f is replaced with Δx
- Weak compressibilty and small Mach number assumed
- Genuine Cartesian embedded boundary approach

Equation is approximated with a splitting approach:

1.) Transport step solves $\partial_t f_\alpha + \mathbf{e}_\alpha \cdot \nabla f_\alpha = 0$

2.) Collision step $\partial_t f_\alpha = \omega(f_\alpha^{eq} - f_\alpha) + F_\alpha$



Lattice Boltzmann method (LBM)

Instead of solving the Navier-Stokes equations, we use the lattice Boltzmann method.

The LBM is based on solving the Boltzmann equation with a simplified collision operator

$$\partial_t f + \mathbf{u} \cdot \nabla f = \omega(f^{eq} - f) + F$$

- $\mathrm{Kn} = l_f / L \ll 1$, where l_f is replaced with Δx
- Weak compressibilty and small Mach number assumed
- Genuine Cartesian embedded boundary approach

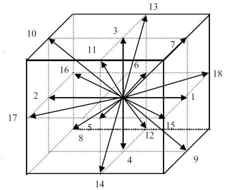
Equation is approximated with a splitting approach:

1.) Transport step solves $\partial_t f_\alpha + \mathbf{e}_\alpha \cdot \nabla f_\alpha = 0$

2.) Collision step $\partial_t f_\alpha = \omega(f_\alpha^{eq} - f_\alpha) + F_\alpha$

Macroscopic quantities from moments:

$$\rho(\mathbf{x}, t) = \sum_{\alpha=0}^{18} f_\alpha(\mathbf{x}, t), \quad \rho(\mathbf{x}, t) u_i(\mathbf{x}, t) = \sum_{\alpha=0}^{18} \mathbf{e}_{\alpha i} f_\alpha(\mathbf{x}, t)$$



Lattice Boltzmann method (LBM)

Instead of solving the Navier-Stokes equations, we use the lattice Boltzmann method.

The LBM is based on solving the Boltzmann equation with a simplified collision operator

$$\partial_t f + \mathbf{u} \cdot \nabla f = \omega(f^{eq} - f) + F$$

- $\mathrm{Kn} = l_f / L \ll 1$, where l_f is replaced with Δx
- Weak compressibilty and small Mach number assumed
- Genuine Cartesian embedded boundary approach

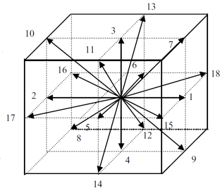
Equation is approximated with a splitting approach:

1.) Transport step solves $\partial_t f_\alpha + \mathbf{e}_\alpha \cdot \nabla f_\alpha = 0$

2.) Collision step $\partial_t f_\alpha = \omega(f_\alpha^{eq} - f_\alpha) + F_\alpha$

Macroscopic quantities from moments:

$$\rho(\mathbf{x},t) = \sum_{\alpha=0}^{18} f_\alpha(\mathbf{x},t), \quad \rho(\mathbf{x},t) u_i(\mathbf{x},t) = \sum_{\alpha=0}^{18} \mathbf{e}_{\alpha i} f_\alpha(\mathbf{x},t)$$

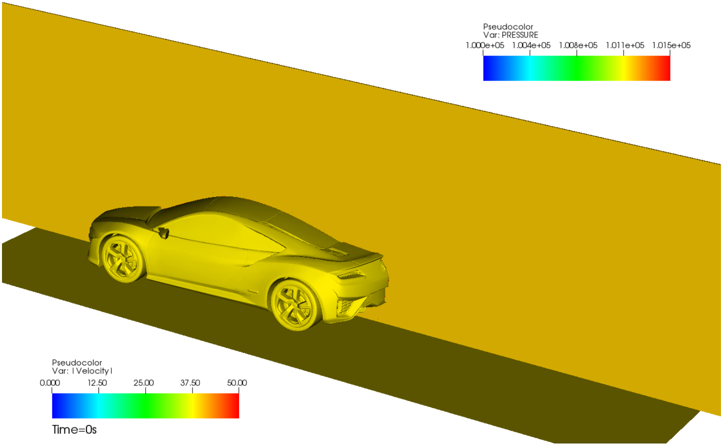


The LBM uses an explicit update based on the speed of sound of the gas.

Initial conditions are constructed for instance as $f_\alpha^{eq}(\rho(t=0), \mathbf{u}(t=0))$

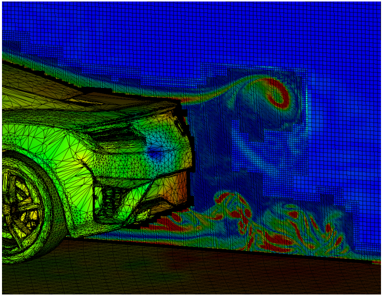
Wind tunnel simulation of a prototype car

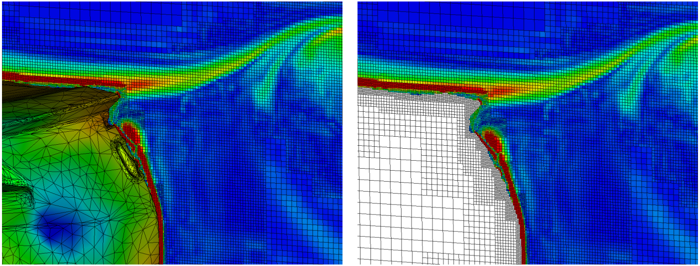
Fluid velocity and pressure on vehicle



- Inflow $40\,\mathrm{m/s}$. LES model active. Characteristic boundary conditions.
- To $t = 0.5\,\mathrm{s}$ ($\sim$ 4 characteristic lengths) with 31,416 time steps on finest level in $\sim 37\,\mathrm{h}$ on 200 cores ($7389\,\mathrm{h}$ CPU). Channel: $15\,\mathrm{m} \times 5\,\mathrm{m} \times 3.3\,\mathrm{m}$

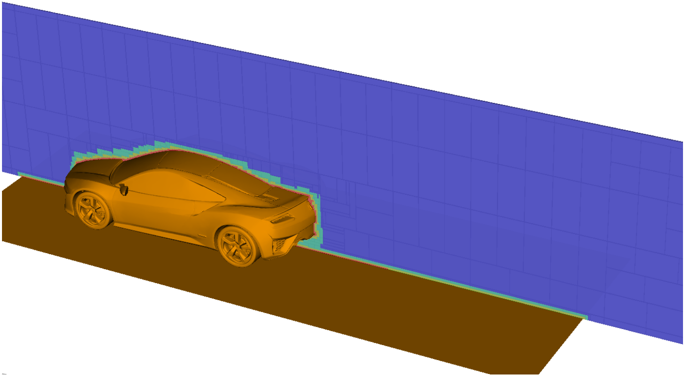
Mesh adaptation





Mesh adaptation

Used refinement blocks and levels (indicated by color)



- SAMR base grid 600 × 200 × 132 cells, 3 additional levels with factor 2 yielding finest resolution of $\Delta x = 3.125\,\mathrm{mm}$
- Adaptation based on level set and scaled gradient of magnitude of vorticity vector
- 236M cells vs. 8.1 billion (uniform) at $t = 0.4075\,\mathrm{s}$

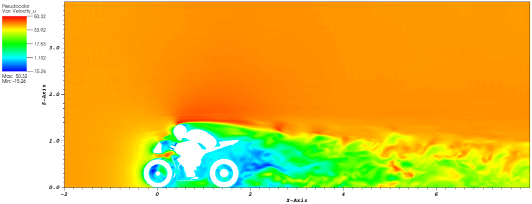
Refinement at $t = 0.4075\,\mathrm{s}$

Level	Grids	Cells
0	11,605	15,840,000
1	11,513	23,646,984
2	31,382	144,447,872
3	21,221	52,388,336

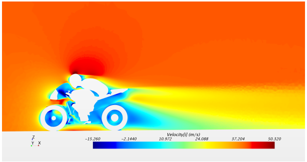
Flow over a motorcycle

- Inflow 40 $\mathrm{m/s}$. Bouzidi pressure boundary conditions at outflows. LES model active.
- SAMR base grid $200 \times 80 \times 80$ cells, 3 additional levels with factor 2 yielding finest resolution of $\Delta x = 6.25\,\mathrm{mm}$. 23560 time steps on finest level
- Forces in AMROC-LBM are time-averaged over interval $[0.5\mathrm{s}, 1\mathrm{s}]$
- Unstructured STAR-CCM+ mesh has significantly finer as well as coarser cells

AMROC-LBM LES at $t = 1\,\mathrm{s}$



STAR-CCM+ steady RANS



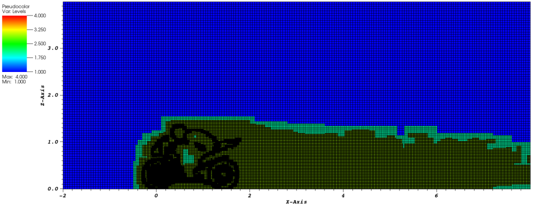
Velocity in flow direction

Variables	Forces (N)				Cores	Wall Time	CPU Time
	Drag	Sideforce	Lift	Total		h	h
STAR-CCM+	297	5	9	297	10	4.9	78
AMROC	297	10	23	298	64	10	635

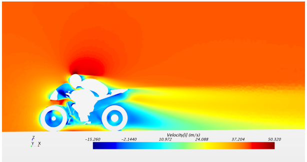
Flow over a motorcycle

- Inflow 40 $\mathrm{m/s}$. Bouzidi pressure boundary conditions at outflows. LES model active.
- SAMR base grid $200 \times 80 \times 80$ cells, 3 additional levels with factor 2 yielding finest resolution of $\Delta x = 6.25\,\mathrm{mm}$. 23560 time steps on finest level
- Forces in AMROC-LBM are time-averaged over interval $[0.5\mathrm{s}, 1\mathrm{s}]$
- Unstructured STAR-CCM+ mesh has significantly finer as well as coarser cells

AMROC-LBM LES at $t = 1\,\mathrm{s}$



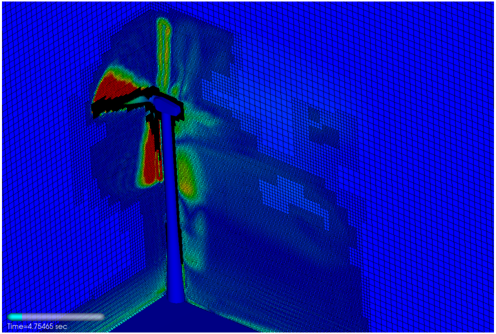
STAR-CCM+ steady RANS



Velocity in flow direction

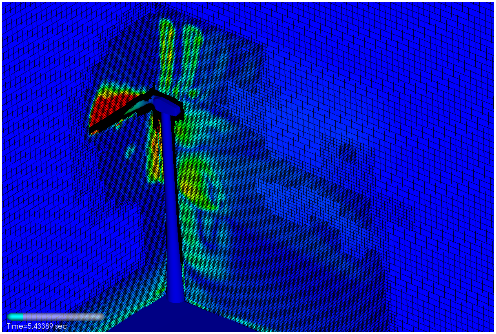
Variables	Forces (N) Drag	Sideforce	Lift	Total	Cores	Wall Time h	CPU Time h
STAR-CCM+	297	5	9	297	10	4.9	78
AMROC	297	10	23	298	64	10	635

Single Vestas V27 wind turbine



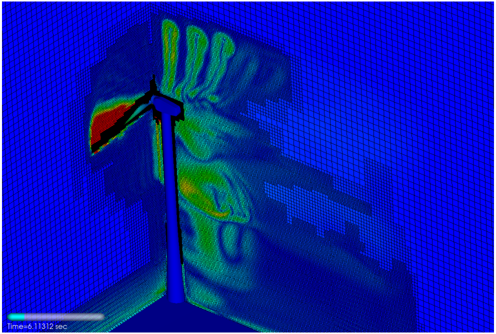
- Inflow velocity $u_\infty = 8\,\mathrm{m/s}$. Prescribed motion of rotor with $n_{\mathrm{rpm}} = 33$, $r = 14.5\,\mathrm{m}$: tip speed $46.7\,\mathrm{m/s}$, $\mathrm{Re}_r \approx 919,700$, tip speed ratio (TSR) 5.84
- Simulation with three additional levels with refinement factors 2, 2, 4
- Refinement based on vorticity and level set
- Computing 84,806 highest level iterations to $t_e = 18\,\mathrm{s}$
- ~ 24 time steps for 1^o rotation

Single Vestas V27 wind turbine



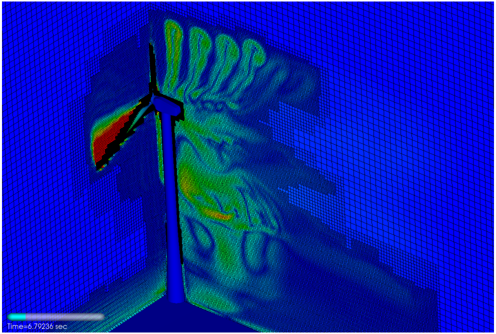
- Inflow velocity $u_\infty = 8\,\mathrm{m/s}$. Prescribed motion of rotor with $n_{\mathrm{rpm}} = 33$, $r = 14.5\,\mathrm{m}$: tip speed $46.7\,\mathrm{m/s}$, $\mathrm{Re}_r \approx 919,700$, tip speed ratio (TSR) 5.84
- Simulation with three additional levels with refinement factors $2, 2, 4$
- Refinement based on vorticity and level set
- Computing 84,806 highest level iterations to $t_e = 18\,\mathrm{s}$
- ~ 24 time steps for 1^o rotation

Single Vestas V27 wind turbine



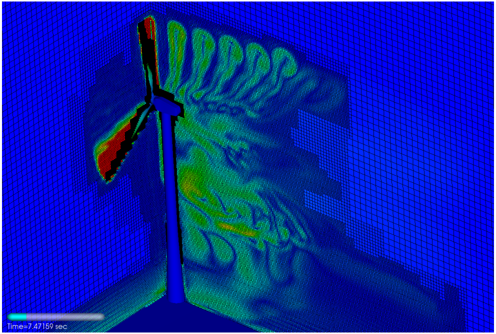
- Inflow velocity $u_\infty = 8\,\mathrm{m/s}$. Prescribed motion of rotor with $n_{\mathrm{rpm}} = 33$, $r = 14.5\,\mathrm{m}$: tip speed $46.7\,\mathrm{m/s}$, $\mathrm{Re}_r \approx 919,700$, tip speed ratio (TSR) 5.84
- Simulation with three additional levels with refinement factors $2, 2, 4$
- Refinement based on vorticity and level set
- Computing 84,806 highest level iterations to $t_e = 18\,\mathrm{s}$
- ~ 24 time steps for 1^o rotation

Single Vestas V27 wind turbine



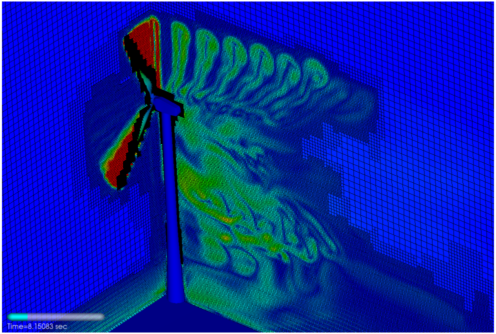
- Inflow velocity $u_\infty = 8\,\mathrm{m/s}$. Prescribed motion of rotor with $n_{\mathrm{rpm}} = 33$, $r = 14.5\,\mathrm{m}$: tip speed $46.7\,\mathrm{m/s}$, $\mathrm{Re}_r \approx 919,700$, tip speed ratio (TSR) 5.84
- Simulation with three additional levels with refinement factors $2, 2, 4$
- Refinement based on vorticity and level set
- Computing 84,806 highest level iterations to $t_e = 18\,\mathrm{s}$
- ~ 24 time steps for 1^o rotation

Single Vestas V27 wind turbine



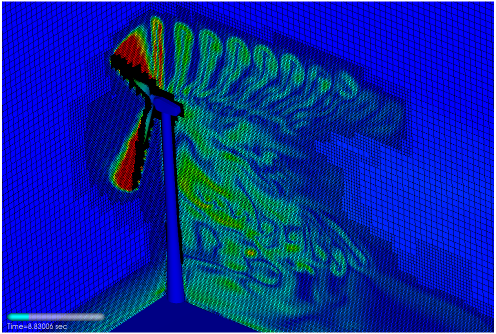
- Inflow velocity $u_\infty = 8\,\mathrm{m/s}$. Prescribed motion of rotor with $n_{\mathrm{rpm}} = 33$, $r = 14.5\,\mathrm{m}$: tip speed $46.7\,\mathrm{m/s}$, $\mathrm{Re}_r \approx 919,700$, tip speed ratio (TSR) 5.84
- Simulation with three additional levels with refinement factors $2, 2, 4$
- Refinement based on vorticity and level set
- Computing 84,806 highest level iterations to $t_e = 18\,\mathrm{s}$
- ~ 24 time steps for 1^o rotation

Single Vestas V27 wind turbine



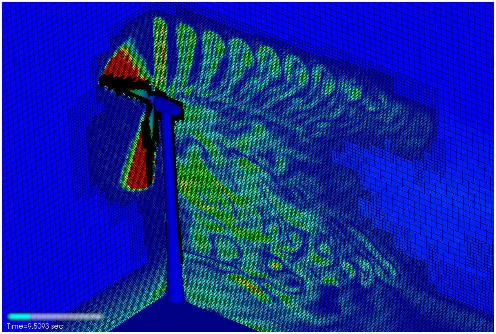
- Inflow velocity $u_\infty = 8\,\mathrm{m/s}$. Prescribed motion of rotor with $n_{\mathrm{rpm}} = 33$, $r = 14.5\,\mathrm{m}$: tip speed $46.7\,\mathrm{m/s}$, $\mathrm{Re}_r \approx 919,700$, tip speed ratio (TSR) 5.84
- Simulation with three additional levels with refinement factors $2, 2, 4$
- Refinement based on vorticity and level set
- Computing 84,806 highest level iterations to $t_e = 18\,\mathrm{s}$
- ~ 24 time steps for 1^o rotation

Single Vestas V27 wind turbine



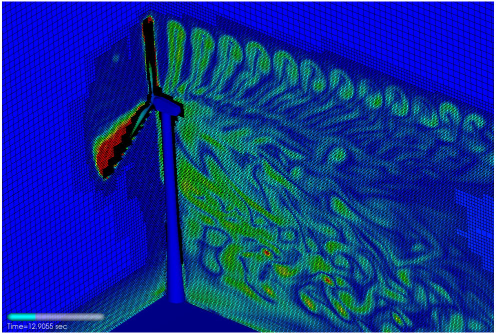
- Inflow velocity $u_\infty = 8\,\mathrm{m/s}$. Prescribed motion of rotor with $n_{\mathrm{rpm}} = 33$, $r = 14.5\,\mathrm{m}$: tip speed $46.7\,\mathrm{m/s}$, $\mathrm{Re}_r \approx 919,700$, tip speed ratio (TSR) 5.84
- Simulation with three additional levels with refinement factors $2, 2, 4$
- Refinement based on vorticity and level set
- Computing 84,806 highest level iterations to $t_e = 18\,\mathrm{s}$
- ~ 24 time steps for 1^o rotation

Single Vestas V27 wind turbine



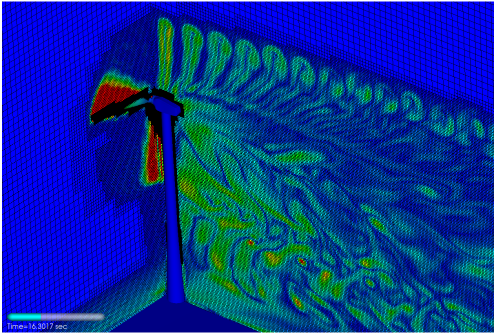
- Inflow velocity $u_\infty = 8\,\mathrm{m/s}$. Prescribed motion of rotor with $n_{\mathrm{rpm}} = 33$, $r = 14.5\,\mathrm{m}$: tip speed $46.7\,\mathrm{m/s}$, $\mathrm{Re}_r \approx 919,700$, tip speed ratio (TSR) 5.84
- Simulation with three additional levels with refinement factors $2, 2, 4$
- Refinement based on vorticity and level set
- Computing 84,806 highest level iterations to $t_e = 18\,\mathrm{s}$
- $\sim$ 24 time steps for 1^o rotation

Single Vestas V27 wind turbine



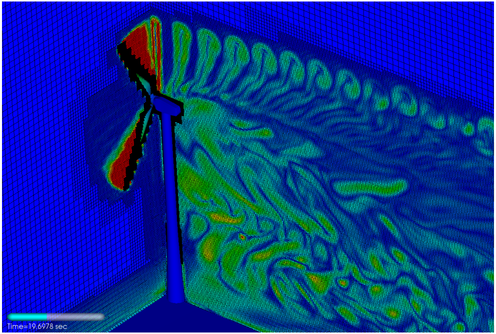
- Inflow velocity $u_\infty = 8\,\mathrm{m/s}$. Prescribed motion of rotor with $n_{\mathrm{rpm}} = 33$, $r = 14.5\,\mathrm{m}$: tip speed $46.7\,\mathrm{m/s}$, $\mathrm{Re}_r \approx 919,700$, tip speed ratio (TSR) 5.84
- Simulation with three additional levels with refinement factors $2, 2, 4$
- Refinement based on vorticity and level set
- Computing 84,806 highest level iterations to $t_e = 18\,\mathrm{s}$
- ~ 24 time steps for 1^o rotation

Single Vestas V27 wind turbine



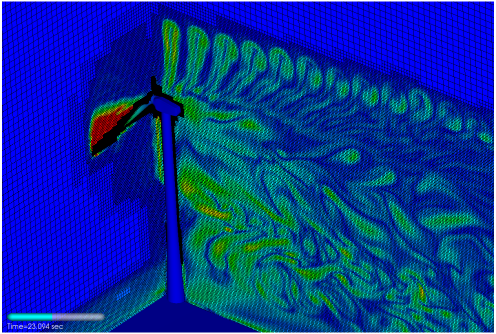
- Inflow velocity $u_\infty = 8\,\mathrm{m/s}$. Prescribed motion of rotor with $n_{\mathrm{rpm}} = 33$, $r = 14.5\,\mathrm{m}$: tip speed $46.7\,\mathrm{m/s}$, $\mathrm{Re}_r \approx 919,700$, tip speed ratio (TSR) 5.84
- Simulation with three additional levels with refinement factors $2, 2, 4$
- Refinement based on vorticity and level set
- Computing 84,806 highest level iterations to $t_e = 18\,\mathrm{s}$
- ~ 24 time steps for 1^o rotation

Single Vestas V27 wind turbine



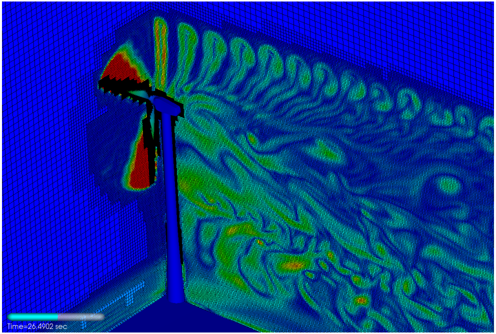
- Inflow velocity $u_\infty = 8\,\mathrm{m/s}$. Prescribed motion of rotor with $n_{\mathrm{rpm}} = 33$, $r = 14.5\,\mathrm{m}$: tip speed $46.7\,\mathrm{m/s}$, $\mathrm{Re}_r \approx 919,700$, tip speed ratio (TSR) 5.84
- Simulation with three additional levels with refinement factors $2, 2, 4$
- Refinement based on vorticity and level set
- Computing 84,806 highest level iterations to $t_e = 18\,\mathrm{s}$
- ~ 24 time steps for 1^o rotation

Single Vestas V27 wind turbine



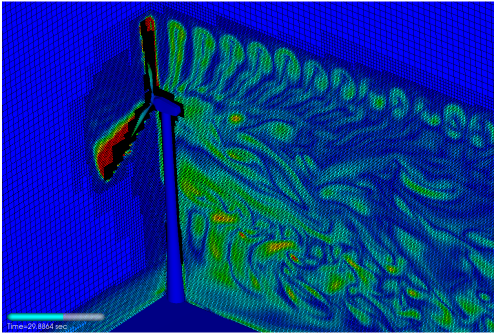
- Inflow velocity $u_\infty = 8\,\mathrm{m/s}$. Prescribed motion of rotor with $n_{\mathrm{rpm}} = 33$, $r = 14.5\,\mathrm{m}$: tip speed $46.7\,\mathrm{m/s}$, $\mathrm{Re}_r \approx 919,700$, tip speed ratio (TSR) 5.84
- Simulation with three additional levels with refinement factors $2, 2, 4$
- Refinement based on vorticity and level set
- Computing 84,806 highest level iterations to $t_e = 18\,\mathrm{s}$
- ~ 24 time steps for 1^o rotation

Single Vestas V27 wind turbine



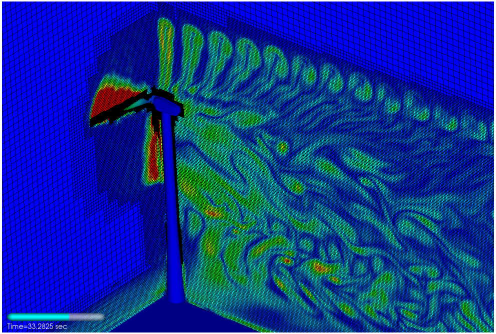
- Inflow velocity $u_\infty = 8\,\mathrm{m/s}$. Prescribed motion of rotor with $n_{\mathrm{rpm}} = 33$, $r = 14.5\,\mathrm{m}$: tip speed $46.7\,\mathrm{m/s}$, $\mathrm{Re}_r \approx 919,700$, tip speed ratio (TSR) 5.84
- Simulation with three additional levels with refinement factors $2, 2, 4$
- Refinement based on vorticity and level set
- Computing 84,806 highest level iterations to $t_e = 18\,\mathrm{s}$
- ~ 24 time steps for 1^o rotation

Single Vestas V27 wind turbine



- Inflow velocity $u_\infty = 8\,\mathrm{m/s}$. Prescribed motion of rotor with $n_{\mathrm{rpm}} = 33$, $r = 14.5\,\mathrm{m}$: tip speed $46.7\,\mathrm{m/s}$, $\mathrm{Re}_r \approx 919,700$, tip speed ratio (TSR) 5.84
- Simulation with three additional levels with refinement factors $2, 2, 4$
- Refinement based on vorticity and level set
- Computing 84,806 highest level iterations to $t_e = 18\,\mathrm{s}$
- ~ 24 time steps for 1^o rotation

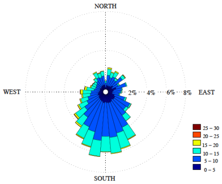
Single Vestas V27 wind turbine

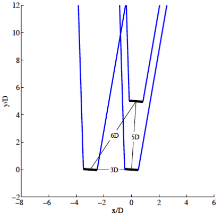


- Inflow velocity $u_\infty = 8\,\mathrm{m/s}$. Prescribed motion of rotor with $n_\mathrm{rpm} = 33$, $r = 14.5\,\mathrm{m}$: tip speed $46.7\,\mathrm{m/s}$, $\mathrm{Re}_r \approx 919,700$, tip speed ratio (TSR) 5.84
- Simulation with three additional levels with refinement factors $2, 2, 4$
- Refinement based on vorticity and level set
- Computing 84,806 highest level iterations to $t_e = 18\,\mathrm{s}$
- ~ 24 time steps for 1^o rotation

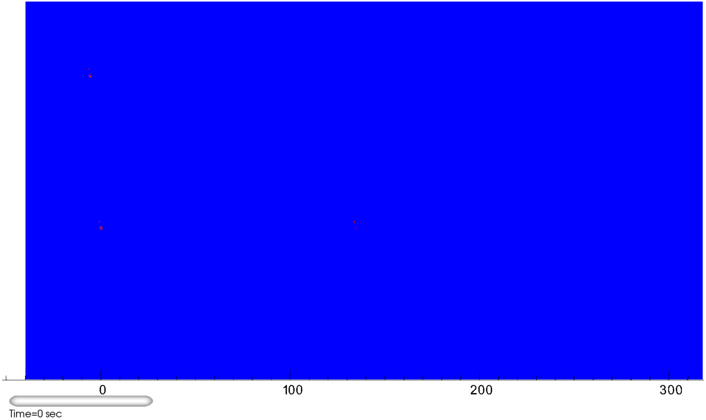
Simulation of the SWIFT array

- Three Vestas V27 turbines (geometric details prototypical). 225 kW power generation at wind speeds 14 to 25 $\mathrm{m/s}$ (then cut-off)
- Prescribed motion of rotor with 33 and 43 rpm. Inflow velocity 8 and 25 $\mathrm{m/s}$
- TSR: 5.84 and 2.43, $\mathrm{Re}_r \approx$ 919,700 and 1,208,000
- Simulation domain $448\,\mathrm{m} \times 240\,\mathrm{m} \times 100\,\mathrm{m}$
- Base mesh $448 \times 240 \times 100$ cells with refinement factors 2, 2,4. Resolution of rotor and tower $\Delta x = 6.25\,\mathrm{cm}$
- 94,224 highest level iterations to $t_e = 40\,\mathrm{s}$ computed, then statistics are gathered for $10\,\mathrm{s}$ [Deiterding and Wood, 2016]



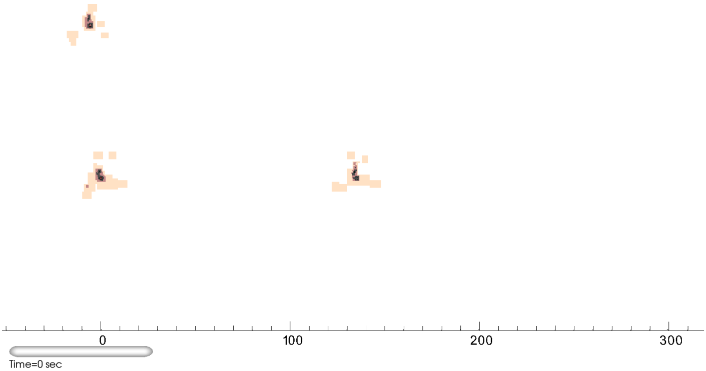


Vorticity – inflow at 30^o, $u = 8\,\mathrm{m/s}$, $33\,\mathrm{rpm}$



- Top view in plane in z-direction at $30\,\mathrm{m}$ (hub height)
- Turbine hub and inflow at 30^o yaw leads to off-axis wake impact.
- 160 cores Intel-Xeon E5 2.6 GHz, $33.03\,\mathrm{h}$ wall time for interval $[50, 60]\,\mathrm{s}$ (including gathering of statistical data)

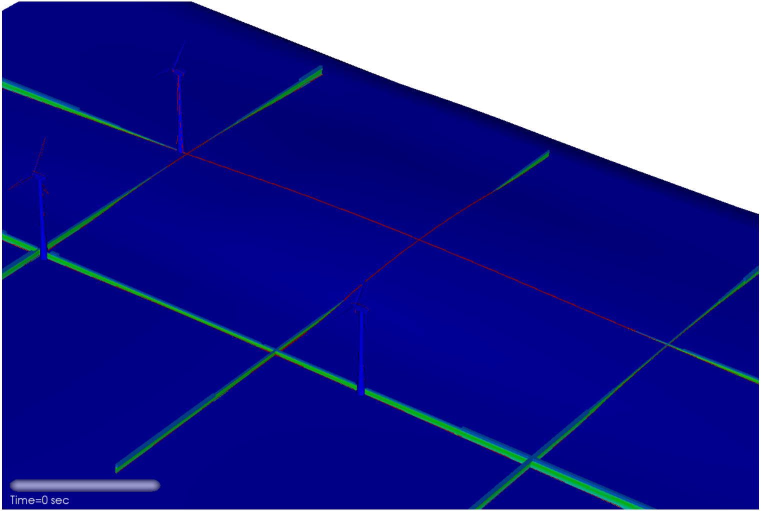
Levels – inflow at 30°, $u = 8\,\mathrm{m/s}$, $33\,\mathrm{rpm}$



- At $63.8\,\mathrm{s}$ approximately 167M cells used vs. 44 billion (factor 264)
- $\sim$ 6.01 h per revolution (961 h CPU) $\longrightarrow$ $\sim$ 5.74 h CPU/1M cells/revolution

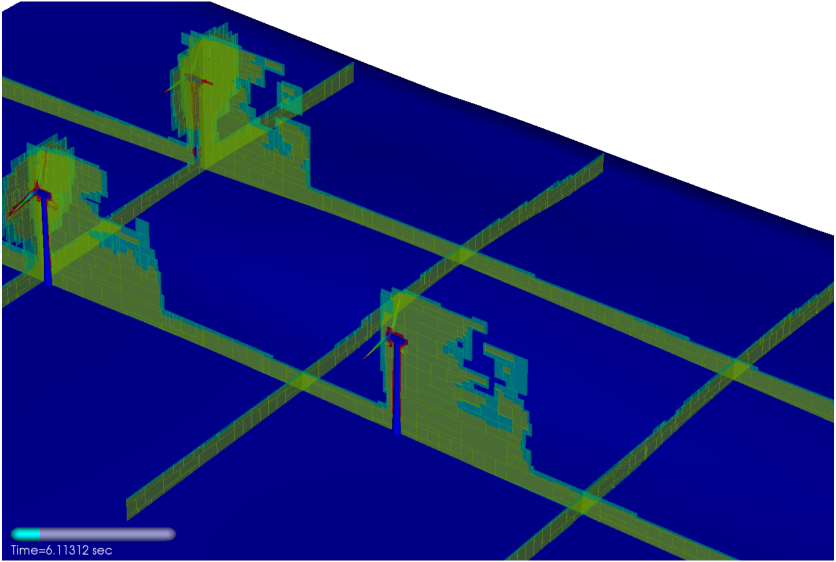
Level	Grids	Cells
0	2,463	10,752,000
1	6,464	20,674,760
2	39,473	131,018,832
3	827	4,909,632

Vorticity development – inflow at 0^o, $u = 8\,\mathrm{m/s}$, $33\,\mathrm{rpm}$

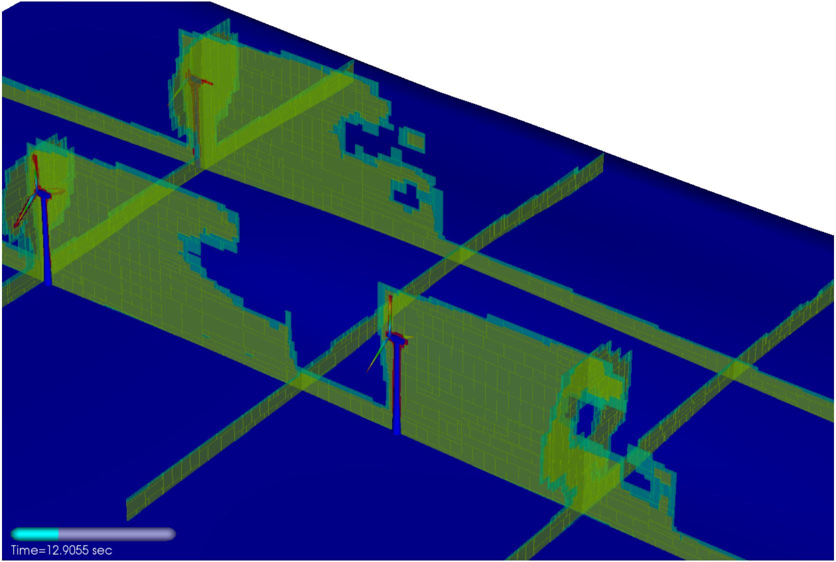


- Refinement of wake up to level 2 ($\Delta x = 25\,\mathrm{cm}$).
- Vortex break-up before 2nd turbine is reached.

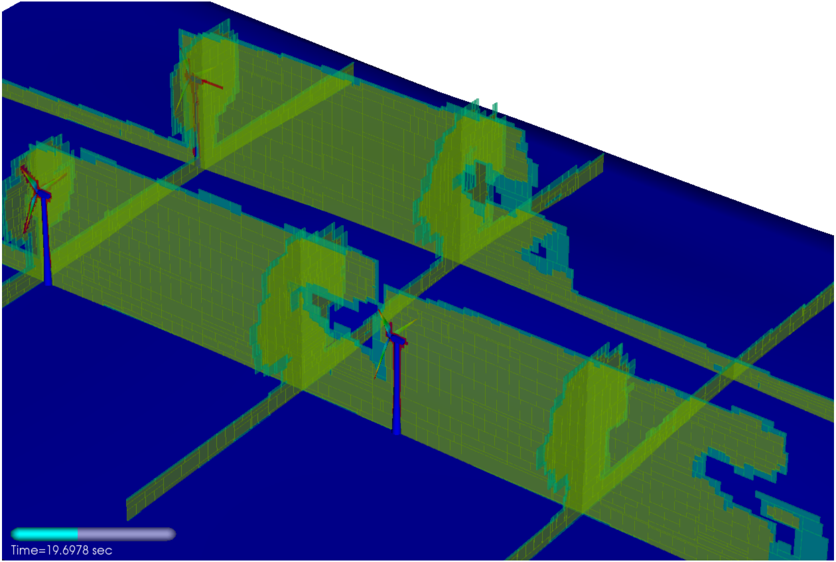
Refinement – inflow at 0^o, $u = 8\,\mathrm{m/s}$, $33\,\mathrm{rpm}$



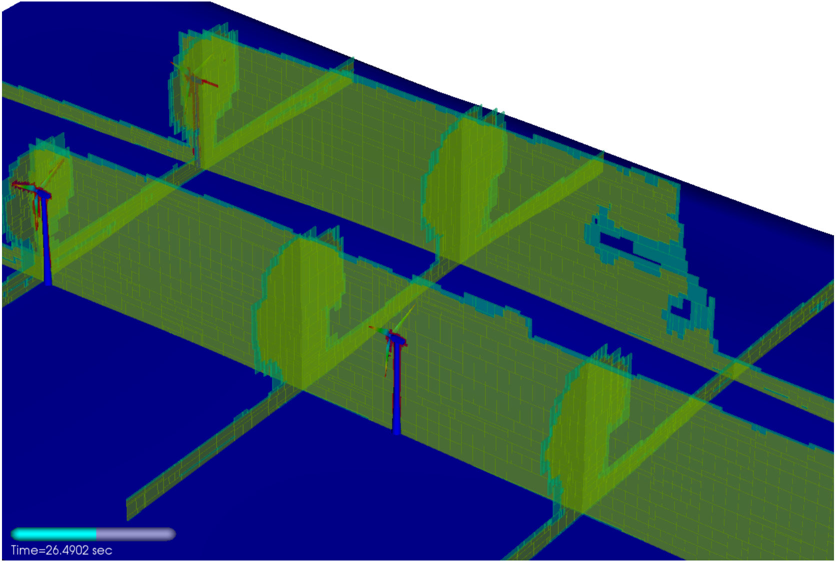
Refinement – inflow at 0^o, $u = 8\,\mathrm{m/s}$, $33\,\mathrm{rpm}$



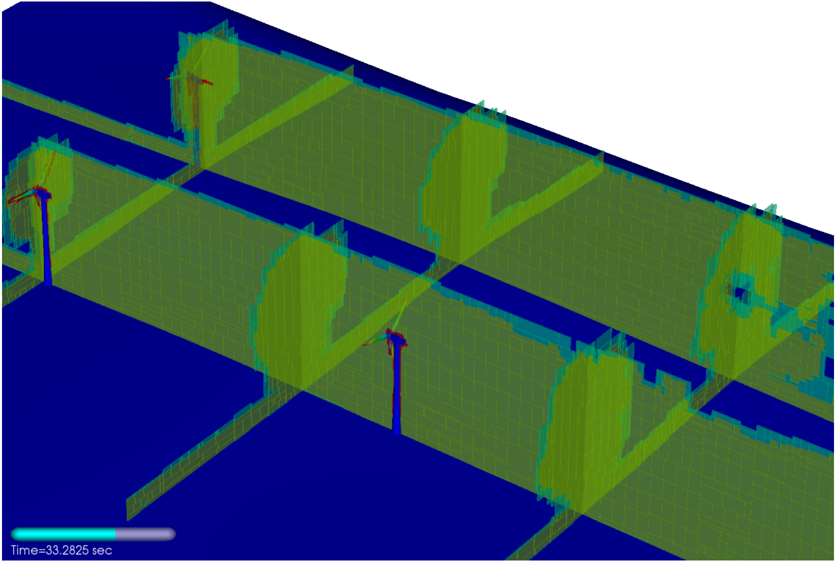
Refinement – inflow at 0^o, $u = 8\,\mathrm{m/s}$, $33\,\mathrm{rpm}$



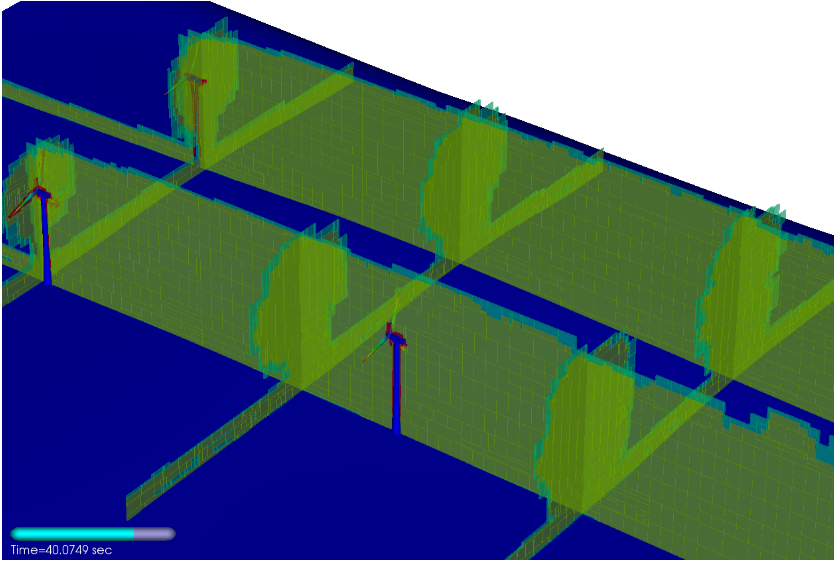
Refinement – inflow at 0^o, $u = 8\,\mathrm{m/s}$, $33\,\mathrm{rpm}$



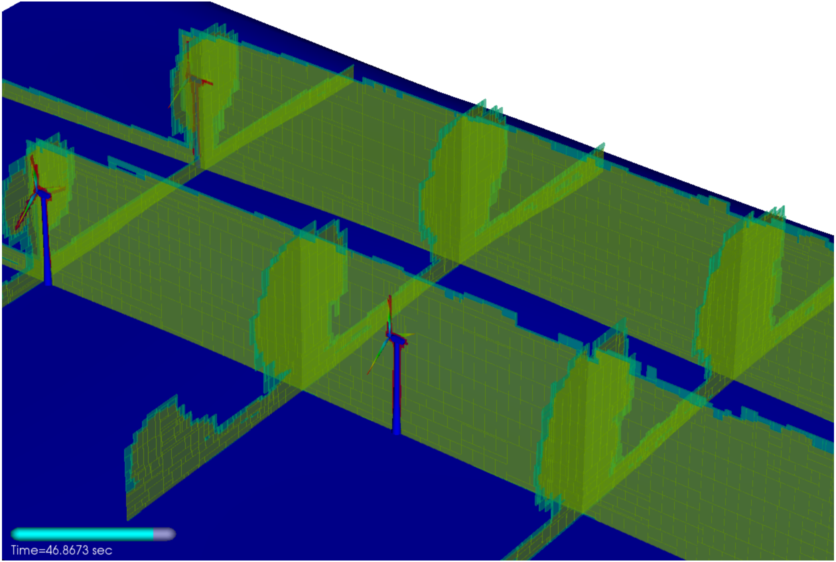
Refinement – inflow at 0^o, $u = 8\,\mathrm{m/s}$, $33\,\mathrm{rpm}$



Refinement – inflow at 0^o, $u = 8\,\mathrm{m/s}$, $33\,\mathrm{rpm}$

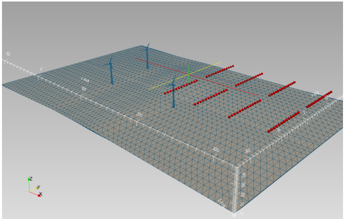


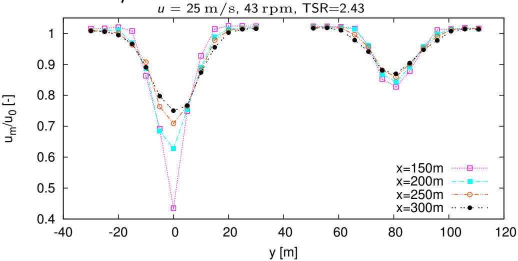
Refinement – inflow at 0^o, $u = 8\,\mathrm{m/s}$, $33\,\mathrm{rpm}$

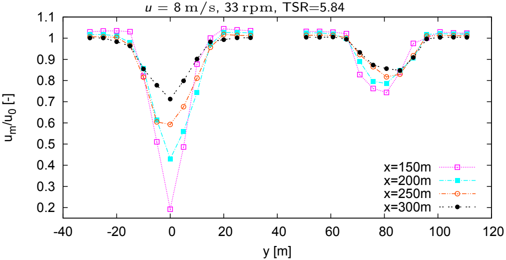


Mean point values – inflow at 0^o,

- Turbines located at $(0, 0, 0)$, $(135, 0, 0)$, $(-5.65, 80.80, 0)$
- Lines of 13 sensors with $\Delta y = 5\,\mathrm{m}$, $z = 37\,\mathrm{m}$ (approx. center of rotor)
- u and p measured over $[40\,\mathrm{s}, 50\,\mathrm{s}]$ (1472 level-0 time steps) and averaged



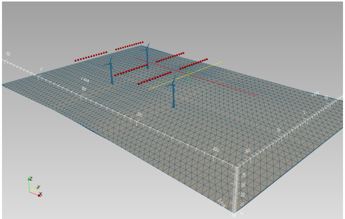


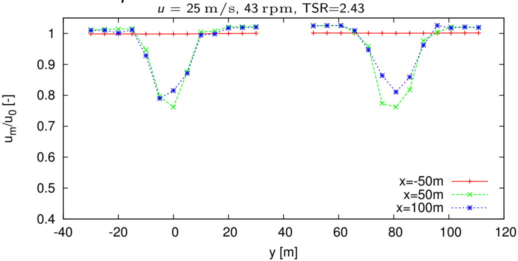


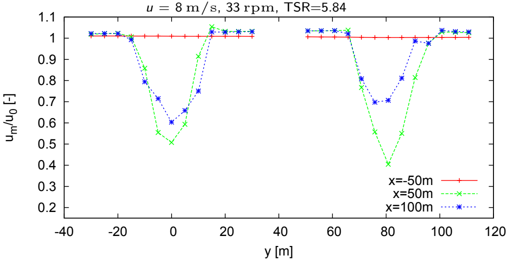
- Velocity deficits larger for higher TSR [Deiterding and Wood, 2016]

Mean point values – inflow at 0^o,

- Turbines located at $(0, 0, 0)$, $(135, 0, 0)$, $(-5.65, 80.80, 0)$
- Lines of 13 sensors with $\Delta y = 5\,\mathrm{m}$, $z = 37\,\mathrm{m}$ (approx. center of rotor)
- u and p measured over $[40\,\mathrm{s}, 50\,\mathrm{s}]$ (1472 level-0 time steps) and averaged







- Velocity deficits larger for higher TSR [Deiterding and Wood, 2016]
- Velocity deficit before 2nd turbine more homogenous for small TSR

Conclusions

- Cartesian CFD with block-based AMR has gained drastically in popularity in recent years
- Automatic mesh generation approach fits well into CAE tool chains

Conclusions

- Cartesian CFD with block-based AMR has gained drastically in popularity in recent years
- Automatic mesh generation approach fits well into CAE tool chains
- Outstanding potential for FSI problems with complex motion and/or compute-intensive multi-scale flow problems
- Patch-based solver approach allows easy incorporation of new methods

Conclusions

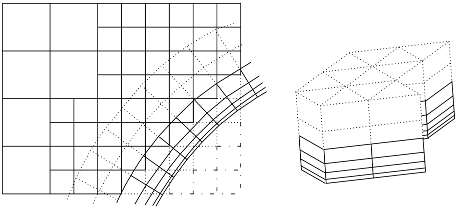
- Cartesian CFD with block-based AMR has gained drastically in popularity in recent years
- Automatic mesh generation approach fits well into CAE tool chains
- Outstanding potential for FSI problems with complex motion and/or compute-intensive multi-scale flow problems
- Patch-based solver approach allows easy incorporation of new methods
- High computational performance on modern massively parallel computer systems
- Hybrid MPI-GPU parallelization is "easier" for block-based AMR than for cell-based approaches

Conclusions

- Cartesian CFD with block-based AMR has gained drastically in popularity in recent years
- Automatic mesh generation approach fits well into CAE tool chains
- Outstanding potential for FSI problems with complex motion and/or compute-intensive multi-scale flow problems
- Patch-based solver approach allows easy incorporation of new methods
- High computational performance on modern massively parallel computer systems
- Hybrid MPI-GPU parallelization is "easier" for block-based AMR than for cell-based approaches

Ongoing work

- Turbulent wall function boundary condition models (particularly for LBM)
- Hybrid overset and "strand-type" meshing



References I

[Berger and Colella, 1988] Berger, M. and Colella, P. (1988). Local adaptive mesh refinement for shock hydrodynamics. *J. Comput. Phys.*, 82:64–84.

[Berger and Helzel, 2002] Berger, M. J. and Helzel, C. (2002). Grid aligned h-box methods for conservation laws in complex geometries. In *Proc. 3rd Intl. Symp. Finite Volumes for Complex Applications*, Porquerolles.

[Deiterding and Wood, 2013] Deiterding, R. and Wood, S. L. (2013). Parallel adaptive fluid-structure interaction simulations of explosions impacting on building structures. *Computers & Fluids*, 88:719–729.

[Deiterding and Wood, 2016] Deiterding, R. and Wood, S. L. (2016). An adaptive lattice Boltzmann method for predicting wake fields behind wind turbines. In Dillmann, A., Heller, G., Krämer, E., Wagner, C., and Breitsamter, C., editors, *New Results in Numerical and Experimental Fluid Mechanics X*, volume 132 of *Notes on Numerical Fluid Mechanics and Multidisciplinary Design*, pages 845–857. Springer.

[Fragner and Deiterding, 2017] Fragner, M. M. and Deiterding, R. (2017). Investigating side-wind stability of high speed trains using high resolution large eddy simulations and hybrid models. In Diez, P., Neittaanmäki, P., Periaux, J., Tuovinen, T., and Bräysy, O., editors, *Computational Methods in Applied Sciences*, volume 45, pages 223–241. Springer.

[Giordano et al., 2005] Giordano, J., Jourdan, G., Burtschell, Y., Medale, M., Zeitoun, D. E., and Houas, L. (2005). Shock wave impacts on deforming panel, an application of fluid-structure interaction. *Shock Waves*, 14(1-2):103–110.

[Hallquist and Lin, 2005] Hallquist, J. and Lin, J. I. (2005). A nonlinear explicit three-dimensional finite element code for solid and structural mechanics. Technical Report UCRL-MA-107254, Lawrence Livermore National Laboratory. Source code (U.S. export controlled) available for licensing fee from http://www.osti.gov/estsc.

[Laurence and Deiterding, 2011] Laurence, S. J. and Deiterding, R. (2011). Shock-wave surfing. *J. Fluid Mech.*, 676:369–431.

[Laurence et al., 2007] Laurence, S. J., Deiterding, R., and Hornung, H. G. (2007). Proximal bodies in hypersonic flows. *J. Fluid Mech.*, 590:209–237.

[Luccioni et al., 2004] Luccioni, B. M., Ambrosini, R. D., and Danesi, R. F. (2004). Analysis of building collapse under blast loads. *Engineering & Structures*, 26:63–71.

[Mader, 1979] Mader, C. L. (1979). *Numerical modeling of detonations*. University of California Press, Berkeley and Los Angeles, California.

References II

[Mauch, 2003] Mauch, S. P. (2003). *Efficient Algorithms for Solving Static Hamilton-Jacobi Equations*. PhD thesis, California Institute of Technology.

[Mittal and Iaccarino, 2005] Mittal, R. and Iaccarino, G. (2005). Immersed boundary methods. *Annu. Rev. Fluid Mech.*, 37:239–261.

[Ngo et al., 2007] Ngo, T., Mendis, P., Gupta, A., and Ramsay, J. (2007). Blast loading and blast effects on structures - an overview. *Electronic Journal of Structual Engineering*.

[Rendleman et al., 2000] Rendleman, C. A., Beckner, V. E., Lijewski, M., Crutchfield, W., and Bell, J. B. (2000). Parallelization of structured, hierarchical adaptive mesh refinement algorithms. *Computing and Visualization in Science*, 3:147–157.

[Roma et al., 1999] Roma, A. M., Perskin, C. S., and Berger, M. J. (1999). An adaptive version of the immersed boundary method. *J. Comput. Phys.*, 153:509–534.