A REVIEW ON THE IMPLEMENTATION ASPECTS OF COUPLED PROBLEMS IN CFD

Ramon Codina

CIMNE

Contents:

Problem statement

There are two main types of coupled problems:

Multiphysics in a single domain

Model equations:

$$
\begin{aligned}
\partial_t x &= F_x(x,y) && \text{in } \Omega \\
\partial_t y &= F_y(x,y) && \text{in } \Omega \\
B_x(x,y) &= 0 && \text{on } \partial\Omega \\
B_y(x,y) &= 0 && \text{on } \partial\Omega
\end{aligned}
$$

Examples:

- **Thermally coupled flows:** Navier-Stokes equations + heat equation
- **Magnetohydrodynamics:** Navier-Stokes equations + Maxwell equations
- **Reactive flows:** Navier-Stokes equations + reaction of chemical species

- **Aeroacoustics:** Navier-Stokes equations + (acoustic analogy) + Helmholtz equation

Different physics in multiple domains

Model equations:

$$\begin{aligned}
\partial_t x &= F_x(x,y) && \text{in } \Omega_1 \\
\partial_t y &= F_y(x,y) && \text{in } \Omega_2 \\
B_x^1(x) &= 0 && \text{on } \partial\Omega_1 \setminus \partial\Omega_1 \cap \partial\Omega_2 \\
B_x^2(x,y) &= 0 && \text{on } \partial\Omega_1 \cap \partial\Omega_2 \\
B_y^1(y) &= 0 && \text{on } \partial\Omega_2 \setminus \partial\Omega_1 \cap \partial\Omega_2 \\
B_y^2(x,y) &= 0 && \text{on } \partial\Omega_1 \cap \partial\Omega_2
\end{aligned}$$

Examples:

- **Fluid-structure interaction (including aeroelasticity):** Navier-Stokes equations + Solid Mechanics equations
- **Coupling of different flow behaviors:**
  - Viscous–inviscid flows
  - Compressible–incompressible flows
  - 3D Navier-Stokes–shallow water equations

Main coupling strategies

We will concentrate on the case of multiphysics in a single domain.

D_t^n will be a finite difference approximation to the temporal derivative, n being the time step counter.

Full coupling

After time discretization:

$$D_t^n x = F_x(x^n, y^n)$$
$$D_t^n y = F_y(x^n, y^n)$$

After linearization:

$$D_t^n \begin{bmatrix} x \\ y \end{bmatrix} = J \begin{bmatrix} x^n \\ y^n \end{bmatrix} + R^n$$

Main properties:

- Large systems of equations
- Difficult to implement

\+ Fast convergence

\+ Robust in time (stability)

Block iterative strategies

After time discretization:

$$D_t^{n,i} x = F_x(x^{n,i}, y^{n,i-1})$$
$$D_t^{n,i} y = F_y(x^{n,i}, y^{n,i}) \quad \text{(Block Jacobi)}$$

Each subproblem has to be linearized independently.

Main properties:

\+ Small systems of equations

\+ Easy to implement

– Convergence is at most linear

\+ Robust in time (stability)

Coupling in multi-stage time integration schemes

After time discretization:

$$D_t^{n+k/m} x = F_x(x^{n+k/m}, y^{n+(k-1)/m})$$

$$D_t^{n+k/m} y = F_y(x^{n+k/m}, y^{n+k/m})$$

Each subproblem has to be linearized independently.

Main properties:

+ Small systems of equations
+ Easy to implement

– The accuracy of the coupling depends on the time step size δt

– Robustness in time may deteriorate

Fractional step methods

After time discretization:

$$D_t^n x = F_x(x^n, y^{n-1})$$
$$D_t^n y = F_y(x^n, y^n)$$

Main properties:

+ Small systems of equations
+ Easy to implement
– The accuracy of the coupling depends on the time step size δt
– Robustness in time may deteriorate
$*$ Very popular in CFD

Data transfer

Single code strategy

+ No duplication of information exists

  Mesh data

  Domain arrays

+ No transfer of data code-to-code or computer-to-computer required

– Difficult to use when the discretization is different for the problems being coupled

– Software re-utilization is not possible

Master-slave approach

The idea is to write a master code that controls the transfer of information between slaves, which are the codes in charge of solving each subproblem.

Transfer of data code-to-code or computer-to-computer required (through MPI or PVM)

Slaves point of view

> **for** time steps **do**
>
> > **for** linearization steps **do**
> >
> > - Import coupling variable from Master
> > - Solve for the variable of the slave
> > - Export new solution to Master
> >
> > **End for**
>
> **End for**

Master's point of view

- Impose initial conditions

  for time steps **do**

  - Set iteration number $i = 0$

    While convergence not reached **do**

    - Export coupling variables to the slaves
    - Order the slaves to run (possibly in parallel)
    - Import coupling variables from the slaves
    - Process coupling variables to be used by the slaves (interpolation may be needed)
    - $i = i + 1$
    - Check convergence

    End while

  End for

Numerical examples

- Magma flow with ground deformation
  Equations: compressible flow for the magma, viscoelasticity for the ground
  Strategy: master-slave, block iterative coupling
- Filling of a mould
  Equations: incompressible flow (Navier-Stokes) and heat equation in the molten metal, free surface tracking, incompressible flow in the air
  Strategy: single code, block iterative coupling
- Flow in a stirred tank
  Equations: incompressible Navier-Stokes in a fixed reference system in a region surrounding the wall-welded baffles, incompressible Navier-Stokes in a rotating frame of reference in the interior of the tank
  Strategy: master-slave, block iterative coupling

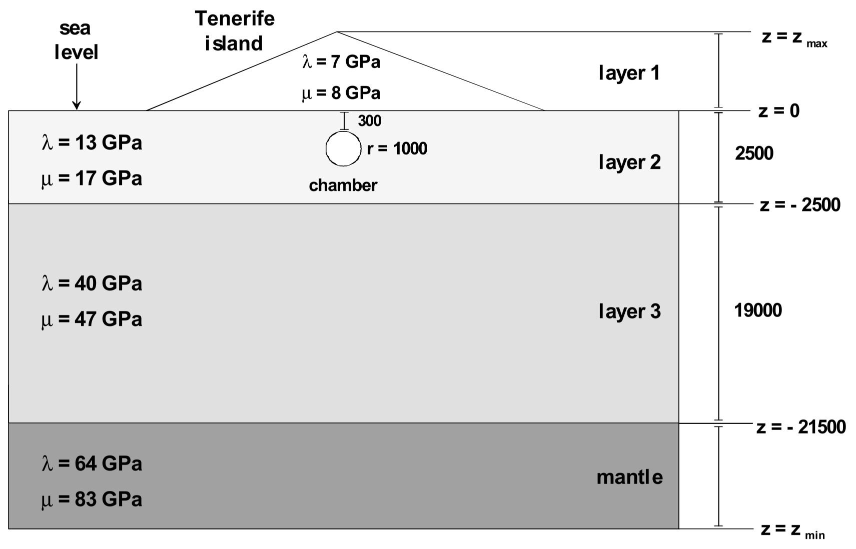
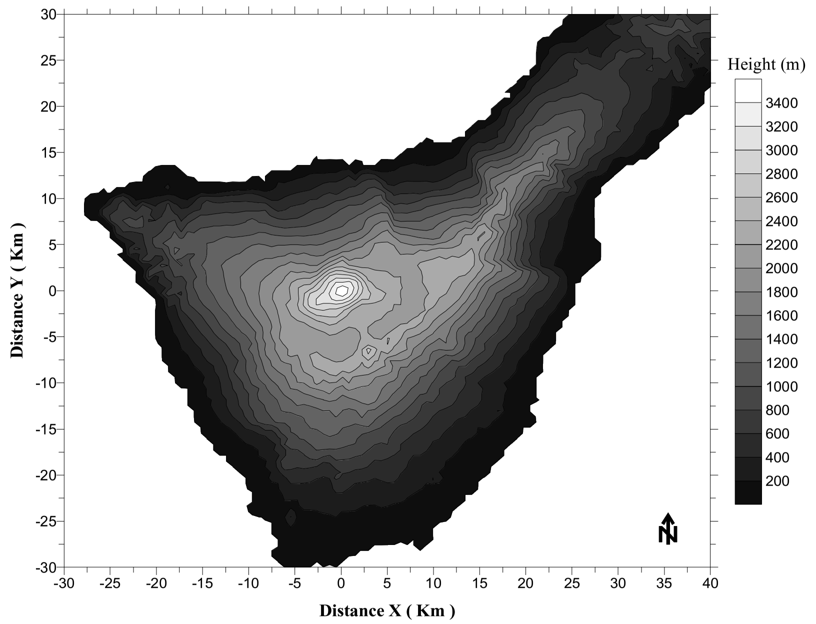


Figure 6.4.2. Schematic illustration (not on scale) of the crustal model used in computations. Four horizontal layers with different elastic Lamé parameters are assumed. The mantle is extended downwards to a distance z_{min}=-30Km, where no deformation is assumed to occur. All the distances are given in m.

The 3D computational domain is a brick with the x and y axis orientated along the WE and SN directions respectively. The z axis runs in the vertical direction (positive upwards) and the origin of coordinates has been placed at the sea level, just below the peak of the Teide. The 1/100.000 scale topography of the island has been added over the uppermost face of the brick. For simplicity, magma chamber is assumed to be a sphere with a radius of 1Km and with its top located at 4Km below the peak of the Teide (i.e. at approximately 300m below the sea level). The shape of the chamber is rather arbitrary, but its depth is deduced from petrological studies [*Wolf*, 1987; *Ablay et al.*, 1998]. Boundary conditions are imposed fixing to zero the displacement field at the bottom and at the lateral boundaries. In order to make negligible the contribution of these boundary conditions to the final solution, computational boundaries have been extended far enough from the pressure source. Thus, the domain has lengths of 70, 60 and 30 Km in the x, y and z directions respectively and almost the whole Tenerife island is covered. Figure 6.4.3 shows the topography of the area selected for computations. The uppermost part of the domain is considered as a free surface[1]. Finally, an overpressure of 10MPa is considered within the magma chamber.

[1] In fact, this face includes the island but also part of the sea. Since batimetry is not included, the sea is treated as a solid material. However, this approach is not so bad as could apparently seem since is done only at the most external parts of the domain, far away from the pressure source.



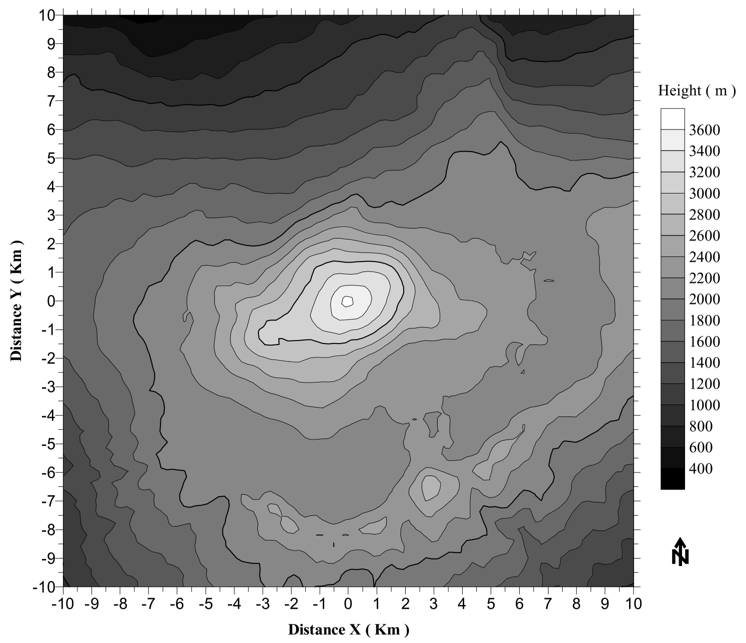


Figure 6.4.3. Top: map showing the topography (height in m) of the area selected for computations. Axis x and y are orientated WE and SN respectively, while axis z (not visible in the figure) runs vertically. The origin of coordinates is located just below the peak of the Teide, at the see level. Computational domain has a length of 70 Km in the x direction and 60 Km in the y direction so that almost the whole island is covered. Bottom: zoom of the topography in Las Cañadas caldera area.

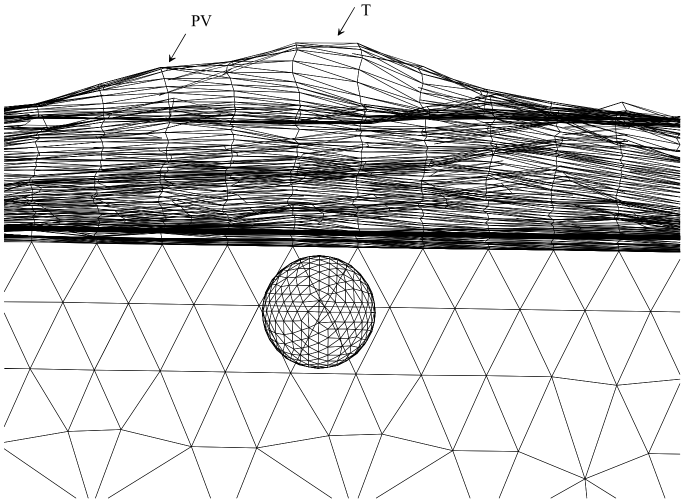


Figure 6.4.7. Close up view of the mesh from the E face around LCC area and without rendering in order to appreciate the magma chamber geometry. Magma chamber is assumed to be spherical (1Km radius) and located 4 Km depth just below the peak of the Teide (i.e. at approximately 300m below the sea level).

The program has been run using a SGI Origin 2000 (8 CPU R10000, 2 GB RAM). The mean required CPU time[1] is about 30-50 min. Part of the results are illustrated in figures 6.4.8 to 6.4.10 which show, respectively, the components of the displacement vector at surface for different characteristic times of $t = 0$, τ , 3τ and 5τ. As observed from these figures, most of the surface deformation is restricted to the vicinity of the Teide (little deformation occurs at distances greater than 5-7Km from the peak of the Teide), that is, the deformation field lies inside the caldera walls. The greater component of the displacement field is vertical (u_z) and has maximum values of about 3.75, 13.75, 21.5 and 34.5 cm for $t = 0$ (elastic displacement), τ, 3τ and 5τ respectively. This vertical deformation pattern is elongated through the NS direction and slightly shifted to the NW of the Teide's peak. In other words, the maximum vertical displacement is predicted to occur (at any time instant) at the NW flank of the volcano. It suggests this part of the volcano as the most suitable place to detect future possible vertical displacements. Another interesting result is the flattening of the vertical displacement field due to topographic effects. This is illustrated in figure 6.4.11.

[1] Obviously, this depends on the number of relaxation times considered. These values correspond to the standard case of 6+1 relaxation times, that is, includes the construction and solution of 7 elastic problems using a conjugate gradient solver and the subsequent Laplace inversion at each nodal point.

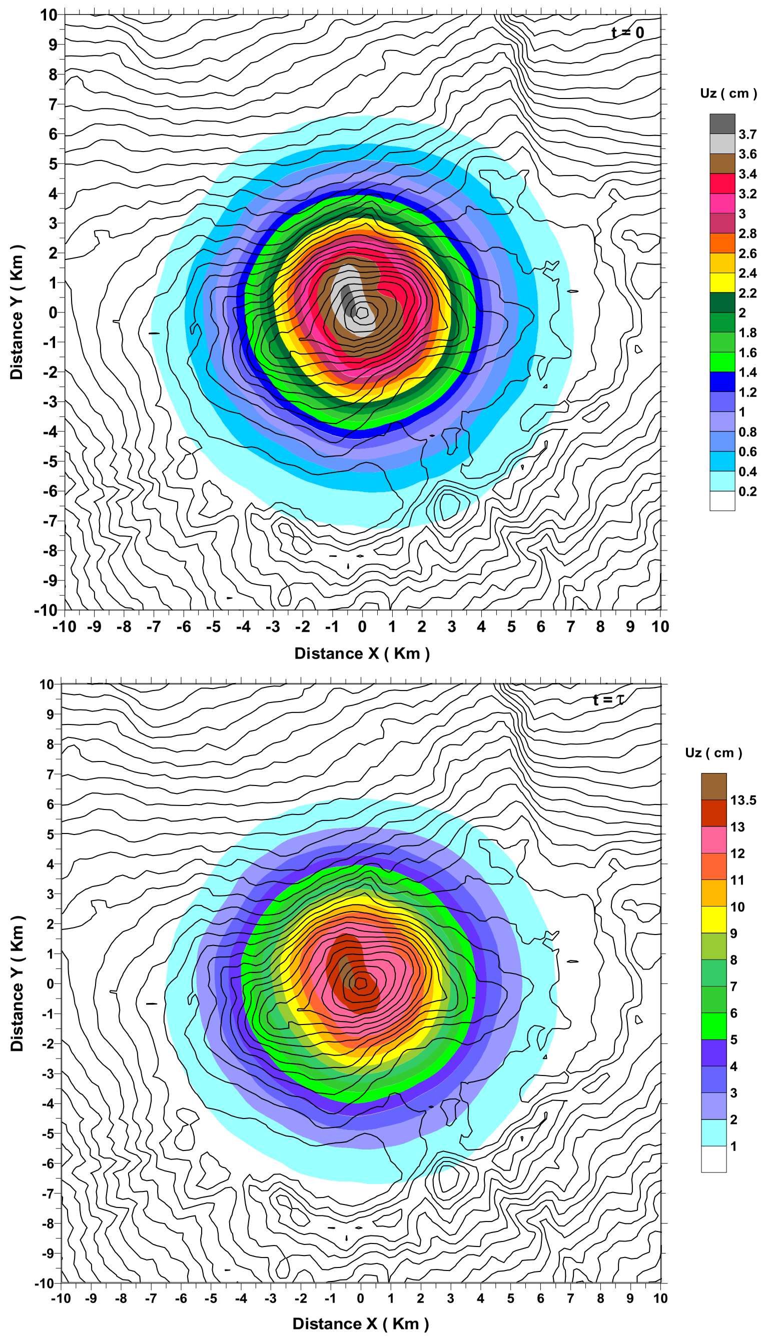
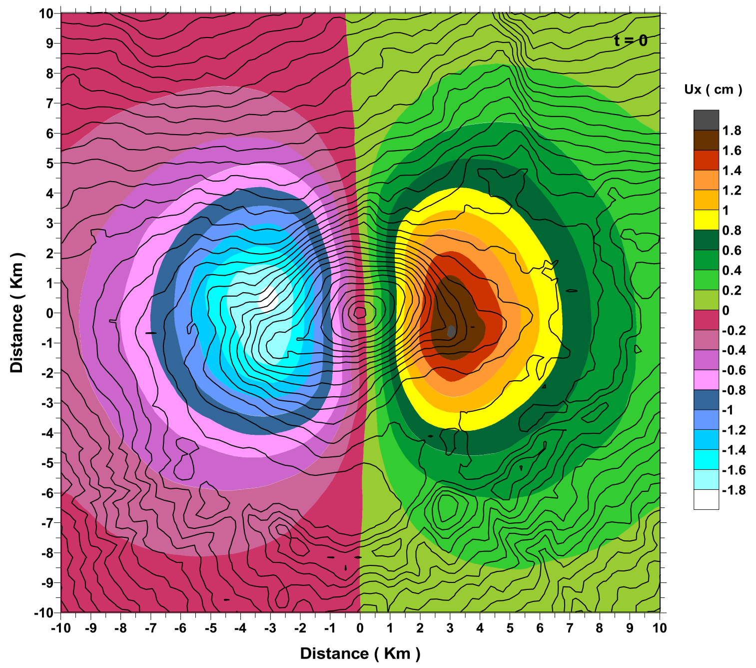


Figure 6.4.8. Vertical displacement u_z (cm) at surface for different time instants: t = 0 (elastic), t = τ , t = 3τ and t = 5τ.



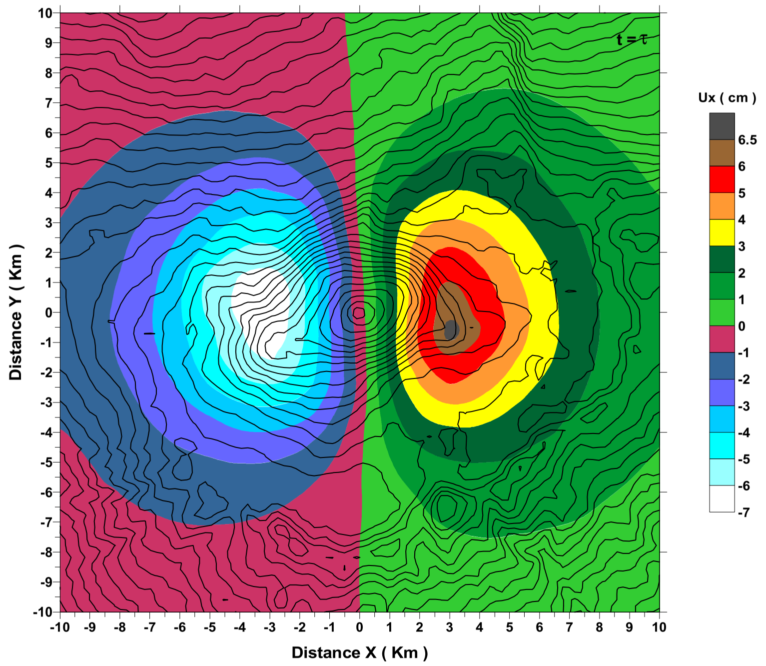
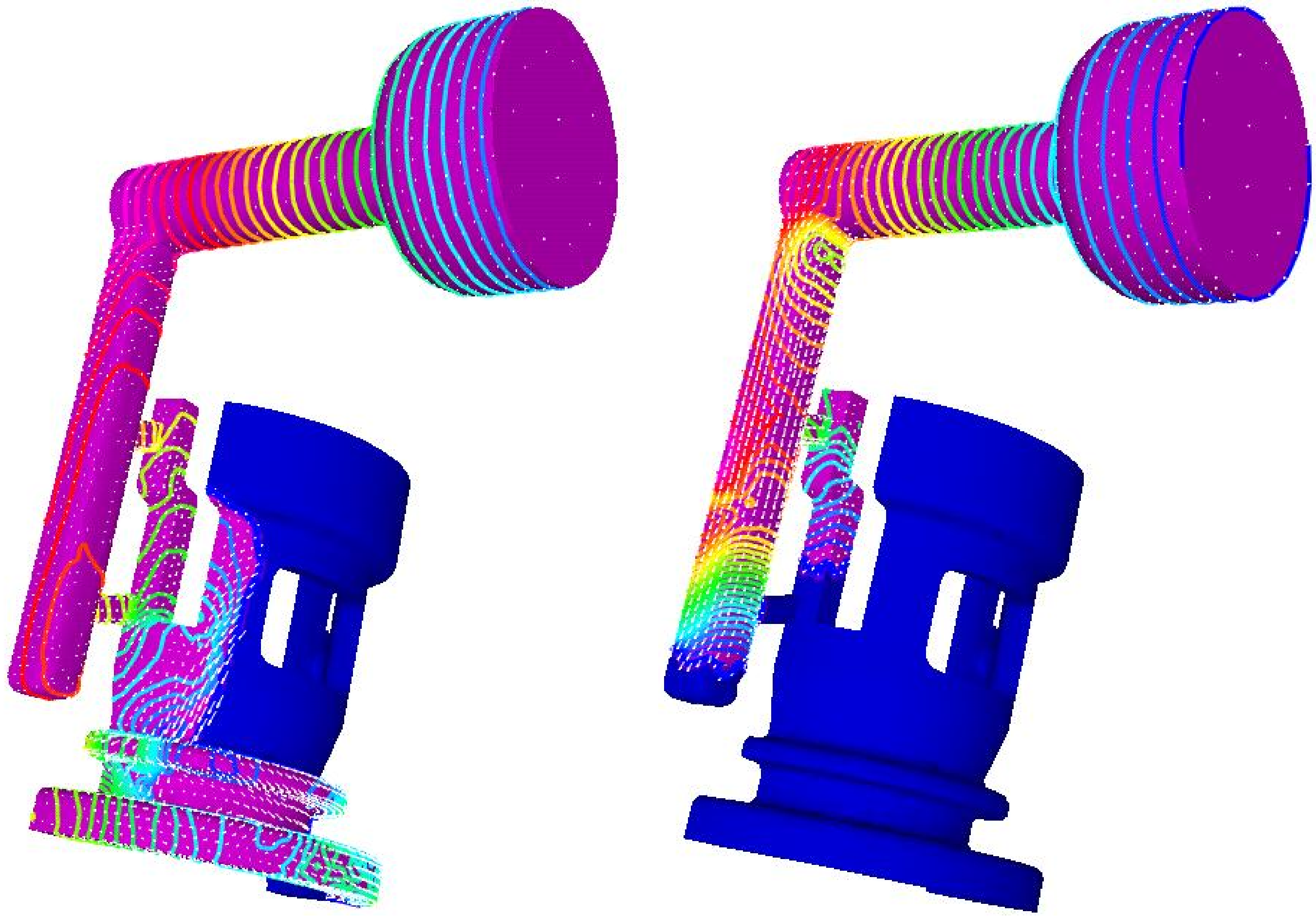
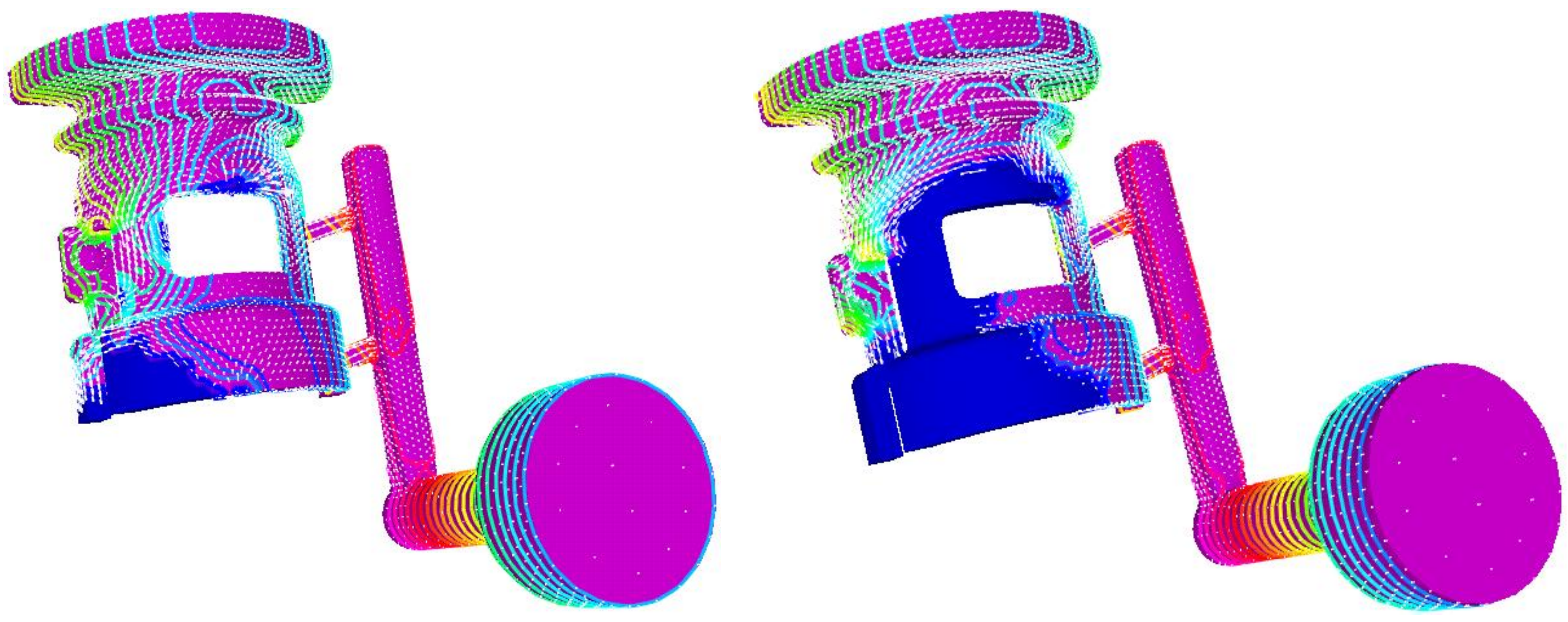
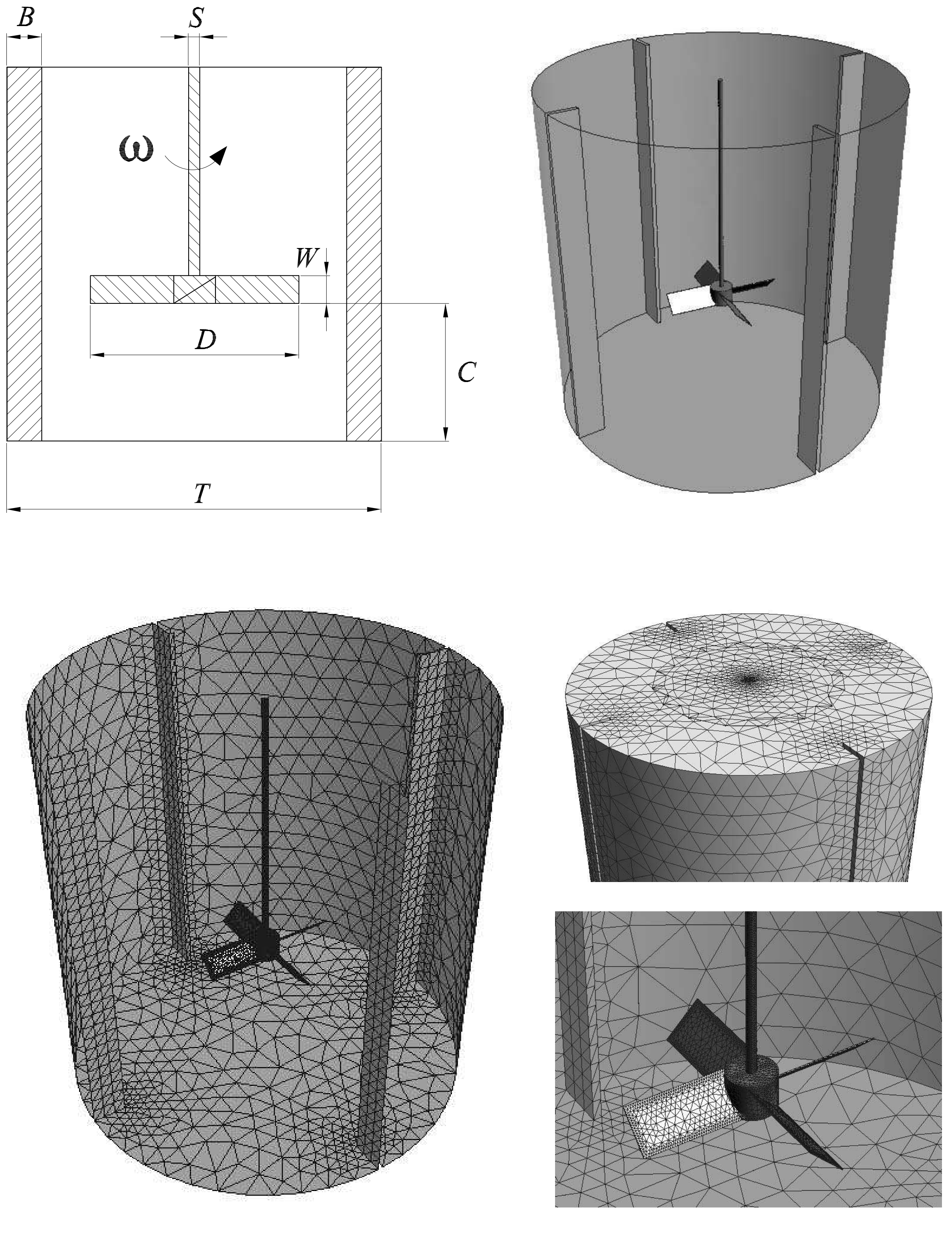
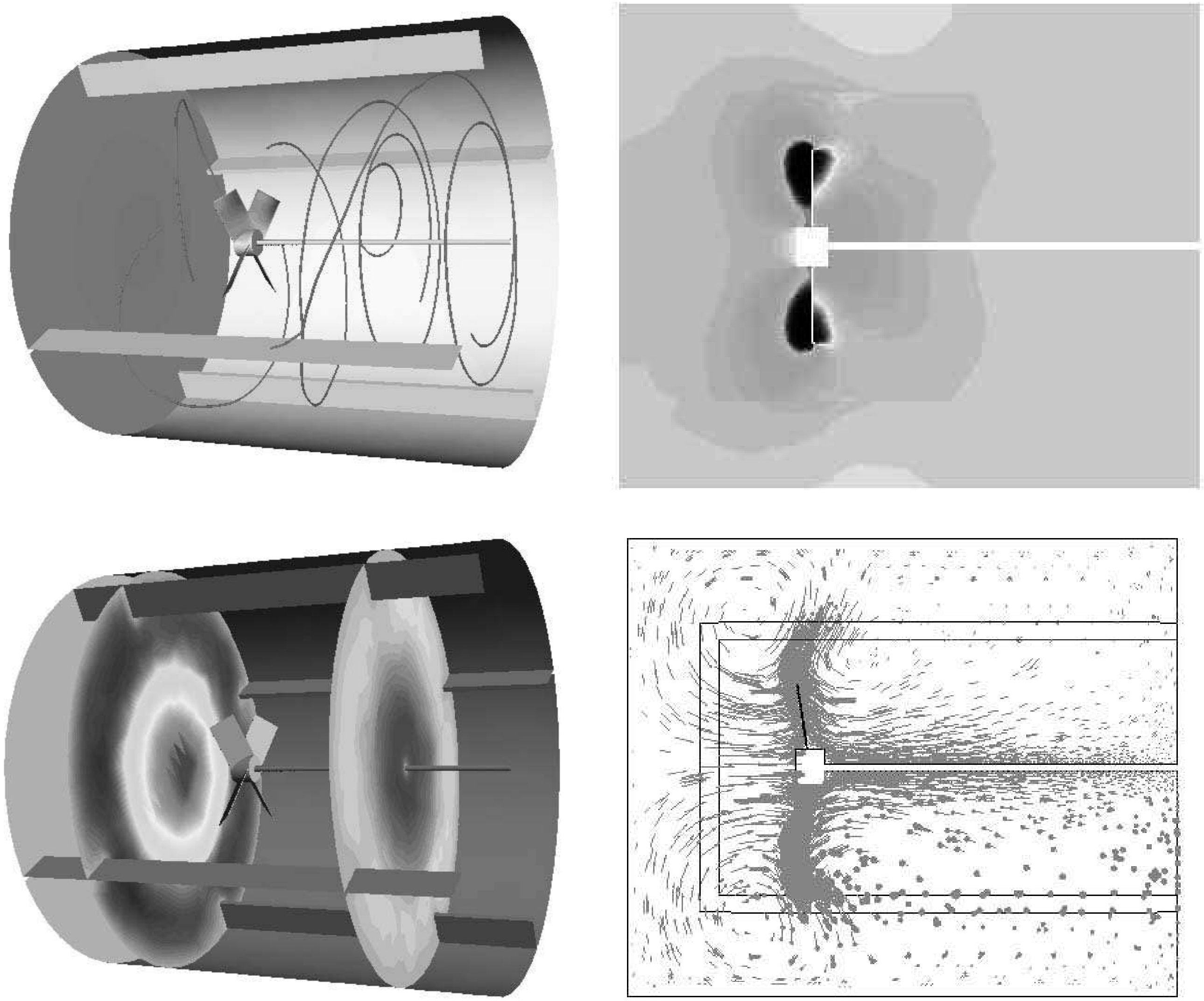


Figure 6.4.9. Horizontal displacement u_x (cm) at surface for different time instants: $t = 0$ (elastic), $t = \tau$, $t = 3\tau$ and $t = 5\tau$.






B
S
ω
W
D
C
T



Concluding remarks

Among the different coupling strategies, we favor the block-iterative coupling:

- Easy to code
- Robust in time (much more that coupling through time stepping)
- Adaptable to the master-slave approach
- Even though convergence is at most linear, obtaining quadratic convergence for fully coupled problems is extremely difficult

Concerning the data transfer, our approach is

- A single code for multiphysics in a single domain if the same mesh is used for all the coupled unknowns
- A master-slave algorithm otherwise (different meshes or different physics in different subdomains)