2D combustor case – FV `surfaceFilm` modeling

Federico Piscaglia [*]
Dept. of Aerospace Science and Technology (DAER), Politecnico di Milano, Italy

Abstract. Lab handout for study of liquid fuel injection inside a combustor-like geometry using a lagrangian particle tracking approach and surface film modeling based on the finite-volume approach by using a secondary region mesh. The solver used is `sprayFoam`, that must be extended to support wall-film modeling.

Topics covered: creation of sets and zones, extrusion of a region mesh, extension of a solver, wall-film model. No combustion nor chemical reactions are activated.

1 Learning outcome

The software used is the open-source CFD software OpenFOAM®-7 by the OpenFOAM Foundation. In this Lab you will learn how to:

- extract sets and zones from the mesh
- set up a solver with a secondary FV mesh for applying region models.
- extend the `sprayFoam` solver with wall-film model
- run a simulation with lagrangian tracking and wall-film modeling

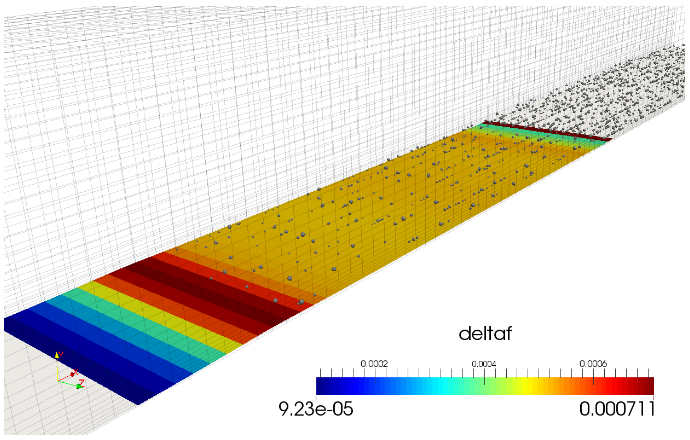

Figure 1: Liquid film and lagrangian particles

2 Introduction

In the combustor case, inject fuel and study fuel transport at low temperature. No combustion is to be simulated. Fuel deposits on the splitter patch and can be eventually stripped away.

[*]Tel. (+39) 02 2399 8620, E-mail: federico.piscaglia@polimi.it

2.1 Region model approach

Reference: K. Meredith and Y. Xin and J. De Vries, A Numerical Model for Simulation of Thin-Film Water Transport over Solid Surfaces, Fire Safety Science 10:415-428. 2001. 10.3801/IAFSS.FSS.10-415

In OpenFOAM®, starting from version 2.2.0, there is the option to model phenomena occurring on thin regions of the domain using the so-called `regionModel` approach. A secondary mesh is generated alongside the main FV domain by extruding a patch or a face zone. Then, a simplified set of governing equations, depending on the phenomenon to be simulated, is solved on the region mesh. Exchange of information with the primary FV domain is handled by mapped patches. At the moment the region models include:

- thermal 1D baffle
- pyrolisis region
- **liquid film** (Fig. **??**)

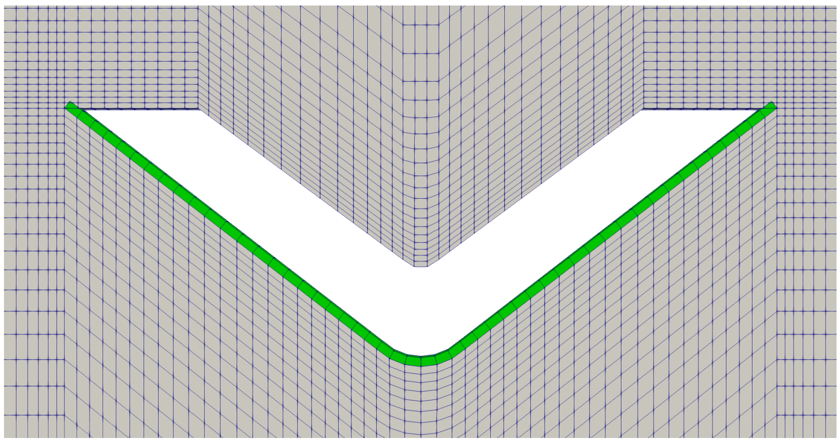

Figure 2: the region mesh (green) is extruded from the patch `splitter`.

2.2 Liquid film equations

The liquid film is modeled using a 1-cell-thick mesh extruded from selected patches (Fig. **??**) . The film thickness δ_f is a cell-centered variable for which a conservation equation is solved:

$$\frac{\partial \rho \delta_f}{\partial t} + \nabla \cdot (\rho \delta_f \boldsymbol{U}_f) = -\rho S_\rho \tag{1}$$

The velocity field is solved in the film surface, hence it is two-dimensional. The momentum equation reads:

$$\frac{\partial \rho \delta_f \boldsymbol{U}_f}{\partial t} + \nabla \cdot (\rho \delta_f \boldsymbol{U}_f \boldsymbol{U}_f) = -\delta_f \nabla p + \boldsymbol{S}_U \tag{2}$$

Similarly, the energy equation is:

$$\frac{\partial \rho \delta_f h}{\partial t} + \nabla \cdot (\rho \delta_f h) = S_h \tag{3}$$

Source terms includes effects of droplets impingement (source of mass, momentum and energy), pressure gradients, wall friction, gravity, aerodynamic interaction with the carrier gas, etc.

A region mesh is generated by extruding selected faces, that must be grouped into a faceZone.

2.3 Grouping mesh entities

Mesh entities (points, faces, cells) can be grouped into the so-called **sets** or **zones**. Sets are list of entities written in separate files into the `polyMesh/sets` subfolder: the name of the file is the name of the set, and the type of entities therein stored is stated in the file header:

```
 /*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  dev                                   |
|   \\  /    A nd           | Web:      www.OpenFOAM.com                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      binary;
    class       faceSet;
    location    "constant/polyMesh/sets";
    object      topBottomPatch;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //


19816
(
691540
691541
...
);
```

Sets are not automatically loaded with the mesh and are mainly used to pre-process the case (we will see an example in this Lab). If the mesh is decomposed or renumbered, sets are decomposed/renumbered too[1], but if the mesh is changing its topology ducing the simulation, sets are not updated. There is no limitation on the number of sets a case can contain, nor in the number of sets a single entity (e.g. a face) can belong to at the same time.

On the other hand, zones are identified by assigning a label to a mesh entity. Zones are automatically loaded in the solver alongside the mesh and are used to define particular properties: porous regions, baffles, moving reference frames, etc. Zones are automatically updated when a mesh changes during the simulation, and an entity cannot belong to more than one zone at the same time. Zones are defined in three files inside the “constant/polyMesh” folder: “pointZones”, “faceZones” and “cellZones”.

To generate a set the procedure is based on the use of topoSetSources: a series of classes used to select points, faces or cells on the basis of geometrical or topological considerations. All sources are listed in

`$FOAM_SRC/meshTools/sets/{pointSources,faceSources,cellSources}`

(see also Fig. **??**)

Sources are normally defined in the “system/topoSetDict” file. For each set the user must specify:

- The type of set (point-, face- or cellSet)
- The name of the set
- The action to be executed (new, delete, add, remove, invert, subset)
- The topoSetSource
- Any additional info needed by the source:

[1]in earlier versions of OpenFOAM® this was not possible, and sets had to be recreated after any decomposition/renumbering

```
actions
(
    {
      name    topBottomPatch;
      type    faceSet;
      action  new;
      source  patchToFace;
      sourceInfo
      {
        name lowerWall;
      }
    }

    {
      name topBottomPatch;
      type faceSet;
      action add;
      source patchToFace;
      sourceInfo
      {
        name topBottom;
      }
    }
);
```

(see the example dictionary in `$FOAM_UTILITIES/application/mesh/manipulation/topoSet`). The command to be run is: `topoSet`

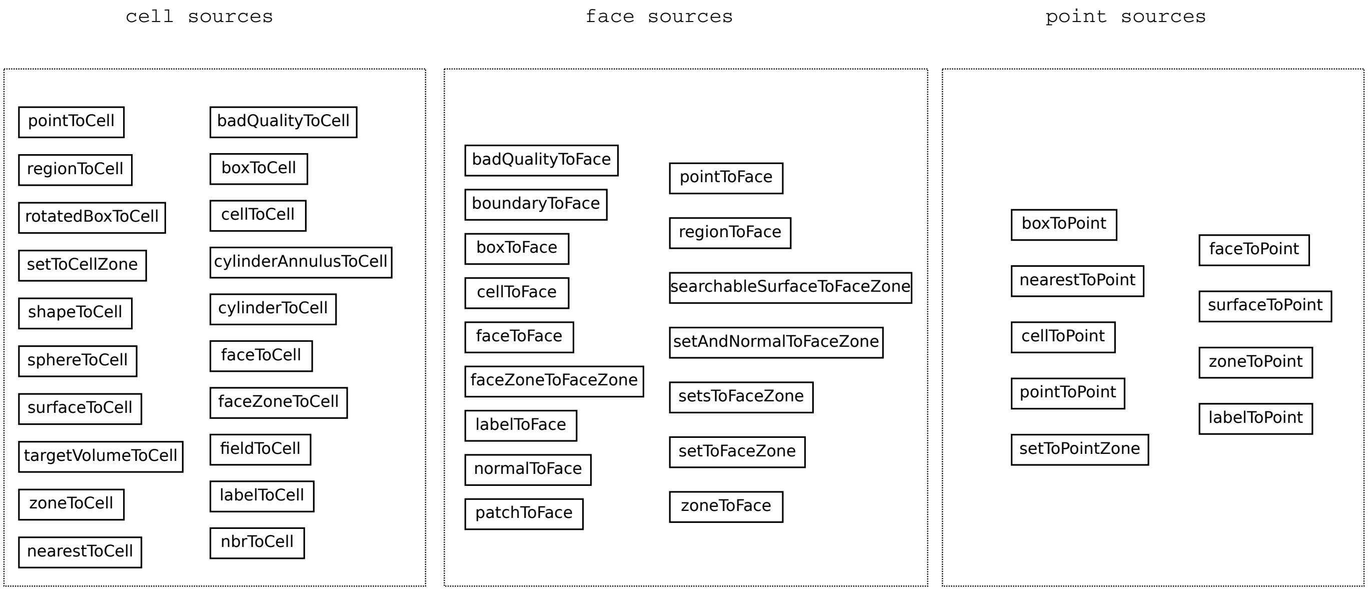

Figure 3: topoSetSources by category

The interactive counterpart of topoSet is “setSet”, which provides a command-line interface to the creation of sets. The same syntax as above becomes:

```
  $ setSet
readline> faceSet topBottomPatch new patchToFace lowerWall
readline> faceSet topBottomPatch add patchToFace topBottom
```

Zones are created by converting sets. This can be done automatically with topoSet using the “setsTo*Zone” sources or with the command “setsToZones”. Some mesh converters retain the zones information defined in the original mesh, so that they can be written in OpenFOAM format together with the mesh:

```
fluentMeshToFoam -writeZones
```

2.4 Multi-region case structure

If an OpenFOAM case is composed by more than one region (e.g. the main gas and the liquid film), the case structure is sligthly modified as follows. For each additional region, a subfolder is created into the “constant”, “system” and time folders, which contains all needed settings for the secondary region. The name of the subfolder is the one of the region. It is a good practice to link the controlDict in the region subfolder to the main one.

```
constant                           0                        system/
|-- combustionProperties           |-- alphat               |-- blockMeshDict
|-- g                              |-- epsilon              |-- controlDict
|-- polyMesh                       |-- k                    |-- createPatchDict
|    |-- boundary                  |-- N2                   |-- decomposeParDict
|    |-- cellZones                 |-- nut                  |-- extrudeToRegionMeshDict
|    |-- faces                     |-- O2                   |-- fvSchemes
|    |-- faceZones                 |-- p                    |-- fvSolution
|    |-- neighbour                 |-- T                    |-- topoSetDict
|    |-- owner                     |-- U                    `-- wallFilmRegion
|    |-- points                    |-- wallFilmRegion           |-- controlDict  -> ../controlDict
|    `-- pointZones                |   |-- deltaf               |-- fvSchemes
|                                  |   |-- Tf                   `-- fvSolution
|-- sprayCloudProperties           |   `-- Uf
|-- surfaceFilmProperties          |
|-- thermophysicalProperties       `-- Ydefault
|-- transportProperties
|-- turbulenceProperties
`-- wallFilmRegion
    `-- polyMesh
         |-- boundary
         |-- cellToPatchFaceAddressing
         |-- cellZones
         |-- faces
         |-- faceToPatchEdgeAddressing
         |-- faceToPatchFaceAddressing
         |-- faceZones
         |-- neighbour
         |-- owner
         |-- points
         |-- pointToPatchPointAddressing
         `-- pointZones
```

3 Generate the wall-film region

1. starting point: compressible simulation, low speed and low temperature at $t = 0.05$ s, with lagrangian tracking setup.

   i. `$ cp -r combustor2D-spray combustor2D-wf`

   ii. `$ cd combustor2D-wf`

   iii. `combustor2D-wf $ ./Allclean`

   iv. `combustor2D-wf $ mv 0 0.bak`

   v. `combustor2D-wf $ tar xzf initialConditions.tgz`

   vi. `combustor2D-wf $ mv initialConditions 0`

2. Create a topoSetDict file. We must create a set and a zone from the splitter patch.

   `$ cp system/controlDict system/topoSetDict`

   Content of the file:

```
actions
(
    {
        name    wallFilmFaceSet;
        type    faceSet;
        action  new;
        source  patchToFace;
        sourceInfo
        {
            name splitter;
        }
    }
    {
        name    wallFilmFaces;
        type    faceZoneSet;
        action  new;
        source  setToFaceZone;
        sourceInfo
        {
            faceSet wallFilmFaceSet;
        }
    }
);
```

3. Generate the set and zone:

   ```
   $ topoSet
   ```

4. Copy the dictionary for extrusion of the faceZone into a regionMesh

   ```
   cp $FOAM_TUTORIALS/lagrangian/reactingParcelFoam/cylinder/system/extrudeToRegionMeshDict system
   ```

   Change thickness to 1 mm

5. Extrude the mesh

   ```
   extrudeToRegionMesh -overwrite
   ```

   The patch that once was “splitter” now is `region0_to_wallFilmRegion_wallFilmFaces`[2]

6. Check:

   ```
   checkMesh
   ```

7. Change patch names globally:

   ```
   cd 0

   for i in *; do
   >    if [ -f $i ]; then
   >    sed -i s/splitter/region0_to_wallFilmRegion_wallFilmFaces/g $i
   >    fi
   >    done
   ```

8. Create BCs for the wall-film region

   ```
   0 $ mkdir wallFilmRegion
   0 $ cp -r $FOAM_TUTORIALS/lagrangian/reactingParcelFoam/cylinder/0.orig/wallFilmRegion .
   ```

   i. Change “frontAndBack” BC type to “empty”
   ii. Delete “sides”
   iii. Add “fuel_inlet”:
      - “zeroGradient” on Tf, deltaf and Uf (this is an outlet)

[2]The name is `<currentRegion>_to_<otherRegion>_<patchOnOtherRegion>`

9. The “system/wallFilmRegion” was automatically generated during extrusion. However, its content must be set.
   ```
   $ cd system/wallFilmRegion
   $ ln -s ../controlDict .
   $ cp -r $FOAM_TUTORIALS/lagrangian/reactingParcelFoam/cylinder/system/wallFilmRegion/fv* .
   ```

4 Set up the models

4.1 Lagrangian cloud

1. Locate the injector at $x = 0$, $y = -50$ mm.
2. Make the direction of injection parallel to y-axis
3. Activate the dispersion model: `stochasticDispersionRAS`
4. Disable the phase change
5. Activate the surface film interaction:

   ```
   surfaceFilmModel thermoSurfaceFilm;

   thermoSurfaceFilmCoeffs
   {
       active on;
       interactionType absorb;
   }
   ```

6. Change wall interaction type:

   ```
   standardWallInteractionCoeffs
   {
       type            escape;
   }
   ```

4.2 Wall film

1. copy the template dictionary:
   ```
   cp $FOAM_TUTORIALS/lagrangian/reactingParcelFilmFoam/cylinder/constant/surfaceFilmProperties constant
   ```
2. Change the component of the liquid phase from H2O to C7H16
3. Add dripping models: parcels are detached when the film exceeds a threshold height or when they reach the edge of the patch.

   ```
   injectionModels
   (
       drippingInjection
       patchInjection
   );

   patchInjectionCoeffs
   {
       patches (fuel_inlet);
   ```

```
        deltaStable 0;
        cloudName    sprayCloud;
        particlesPerParcel 1;

        parcelDistribution
        {
            type         fixedValue;

            fixedValueDistribution
            {
                value 1e-5;
            }
        }
    }

    drippingInjectionCoeffs
    {
        cloudName    sprayCloud;
        deltaStable  1e-4;
        particlesPerParcel 1;

        parcelDistribution
        {
            type         RosinRammler;
            RosinRammlerDistribution
            {
                minValue        1e-06;
                maxValue        1.5e-4;
                d               1.5e-4;
                n               3;
            }
        }
    }
```

5 Extending the solver

The "sprayFoam" solver does not contain the functions to interact with the wall-film model. One solver that instead is written to account for wall films is "reactingParcelFoam", that however models the lagrangian parcels as a "reactingCloud" instead of a "sprayCloud". This means that some phenomena (e.g. breakup) would be missing from the model. The solution is to extend the "sprayFoam" solver by integrating the wall-film modeling.

1. Select a location:

   ```
   $ run
   $ cd ..
   $ mkdir -p applications/solvers/lagrangian
   $ cd applications/solvers/lagrangian
   $ cp -r $FOAM_SOLVERS/lagrangian/sprayFoam sprayFoam-wf
   ```

2. Change the main source and executable names

   ```
   $ cd sprayFoam-wf
   $ mv sprayFoam.C sprayFoam-wf.C
   ```

   In Make/files change:

```
EXE = $(FOAM_APPBIN)/sprayFoam
```

to:

```
EXE = $(FOAM_USER_APPBIN)/sprayFoam-wf
```

3. Modify the `Make/options` included libraries from `$FOAM_SOLVERS`

```
  EXE_INC = \
    -I. \
    -I$(FOAM_SOLVERS)/lagrangian/reactingParcelFoam \
    -I$(FOAM_SOLVERS)/compressible/rhoPimpleFoam \
```

4. In createFields.H add the highlighted line:

```
    #include "createClouds.H"
    #include "createSurfaceFilmModel.H"
    #include "createRadiationModel.H"
    #include "createFvOptions.H"
```

5. Copy createSurfaceFilmModel.H

```
cp $FOAM_SOLVERS/lagrangian/reactingParcelFoam/createSurfaceFilmModel.H .
```

6. Copy createFieldRefs.H and check the presence of the following line

```
cp $FOAM_SOLVERS/lagrangian/reactingParcelFoam/createFieldRefs.H .
```

```
    regionModels::surfaceFilmModel& surfaceFilm = tsurfaceFilm();
```

7. In sprayFoam-wf.C add the highlighted line:

```
    Info<< "Time = " << runTime.timeName() << nl << endl;
    //...
    parcels.evolve();
    surfaceFilm.evolve();
    #inlude "rhoEqn.H"
```

8. In pEqn.H add the highlighted lines:

```
fvScalarMatrix pEqn
(
    fvm::ddt(psi, p)
  + fvm::div(phid, p)
  - fvm::laplacian(rhorAUf, p)
 ==
    parcels.Srho()
  + surfaceFilm.Srho()
  + fvOptions(psi, p, rho.name())
);

fvScalarMatrix pEqn
(
    fvm::ddt(psi, p)
  + fvc::div(phiHbyA)
  - fvm::laplacian(rhorAUf, p)
 ==
    parcels.Srho()
  + surfaceFilm.Srho()
  + fvOptions(psi, p, rho.name())
);
```

9. In rhoEqn.H check or add the highlighted line:

```
fvScalarMatrix rhoEqn
(
    fvm::ddt(rho)
  + fvc::div(phi)
  ==
    parcels.Srho(rho)
  + surfaceFilm.Srho()
  + fvOptions(rho)
);
```

10. Copy EEqn.H from reactingParcelFoam and check the highlighted line:

```
cp $FOAM_SOLVERS/lagrangian/reactingParcelFoam/EEqn.H .
```

```
fvScalarMatrix EEqn
(
    fvm::ddt(rho, he) + mvConvection->fvmDiv(phi, he)
  + fvc::ddt(rho, K) + fvc::div(phi, K)
  + (
        he.name() == "e"
      ? fvc::div
        (
            fvc::absolute(phi/fvc::interpolate(rho), U),
            p,
            "div(phiv,p)"
        )
      : -dpdt
    )
  - fvm::laplacian(turbulence->alphaEff(), he)
 ==
    rho*(U&g)
  + parcels.Sh(he)
  + surfaceFilm.Sh()
  + radiation->Sh(thermo, he)
  + Qdot
  + fvOptions(rho, he)
);
```

11. copy YEqn.H from reactingParcelFoam and add the highlighted line:

```
cp $FOAM_SOLVERS/lagrangian/reactingParcelFoam/YEqn.H .
```

```
fvScalarMatrix YEqn
(
    fvm::ddt(rho, Yi)
  + mvConvection->fvmDiv(phi, Yi)
  - fvm::laplacian(turbulence->muEff(), Yi)
 ==
    parcels.SYi(i, Yi)
  + surfaceFilm.Srho(i)
  + combustion->R(Yi)
  + fvOptions(rho, Yi)
);
```

12. Compile the solver

```
sprayFoam-wf $ wmake
```

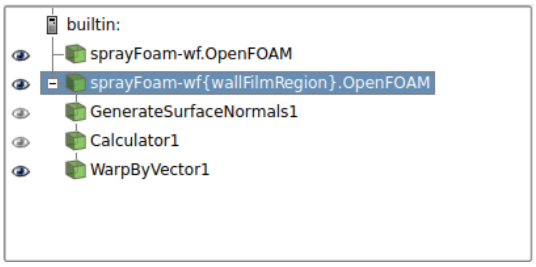

Figure 4: ParaView pipeline

6 Run the solver

Reduce the maximum CFL to 5 and. . .

```
sprayFoam-wf > log.sprayFoam
```

7 Post-processing

7.1 Load case using the OpenFOAM reader (paraFoam)

Thhe main mesh and the wall film must be opened in ParaView as they were two separate cases:

1. `paraFoam -touch`
2. `paraFoam -region wallFilmRegion -touch`
3. `paraview &`
4. Open `spray-wf.OpenFOAM` and `spray-wf{wallFilmRegion}.OpenFOAM`.
5. In the main case, select only the part you are interested in (e.g., all but the lagrangian cloud and the wall patches). Load all fields (volume and lagrangian). Visualization style: wireframe
6. In the region mesh, select only the coupled patch and load all fields. Visualization style: wireframe

7.2 Load case using the built-in paraview reader

Because of a bug in the builtin reader, you will not able to open lagrangian cases. Using foamToVTK is the only option:

1. `$ foamToVTK`
2. `$ foamToVTK -region wallFilmRegion`
3. Open VTK/sprayFoam-wf_*.vtk
4. Use a glyph to show the lagrangian particles
5. Open `VTK/wallFilmRegion/region0_to_wallFilmRegion_wallFilmFaces/*.vtk`
6. Visualization style: wireframe

7.3 Show film displacement

1. Select the wallFilm patch
2. Apply the following filters in sequence:

   → Generate Surface Normals

   → Calculator: point-centered field, `deltaf*Normals`

   → Warp by vector: last generated variable, scale factor = 20