Coupling of Dakota and OpenFOAM for automatic parameterized optimization

Project presentation for MSc/PhD course in CFD with OpenSource software

Adam Jareteg

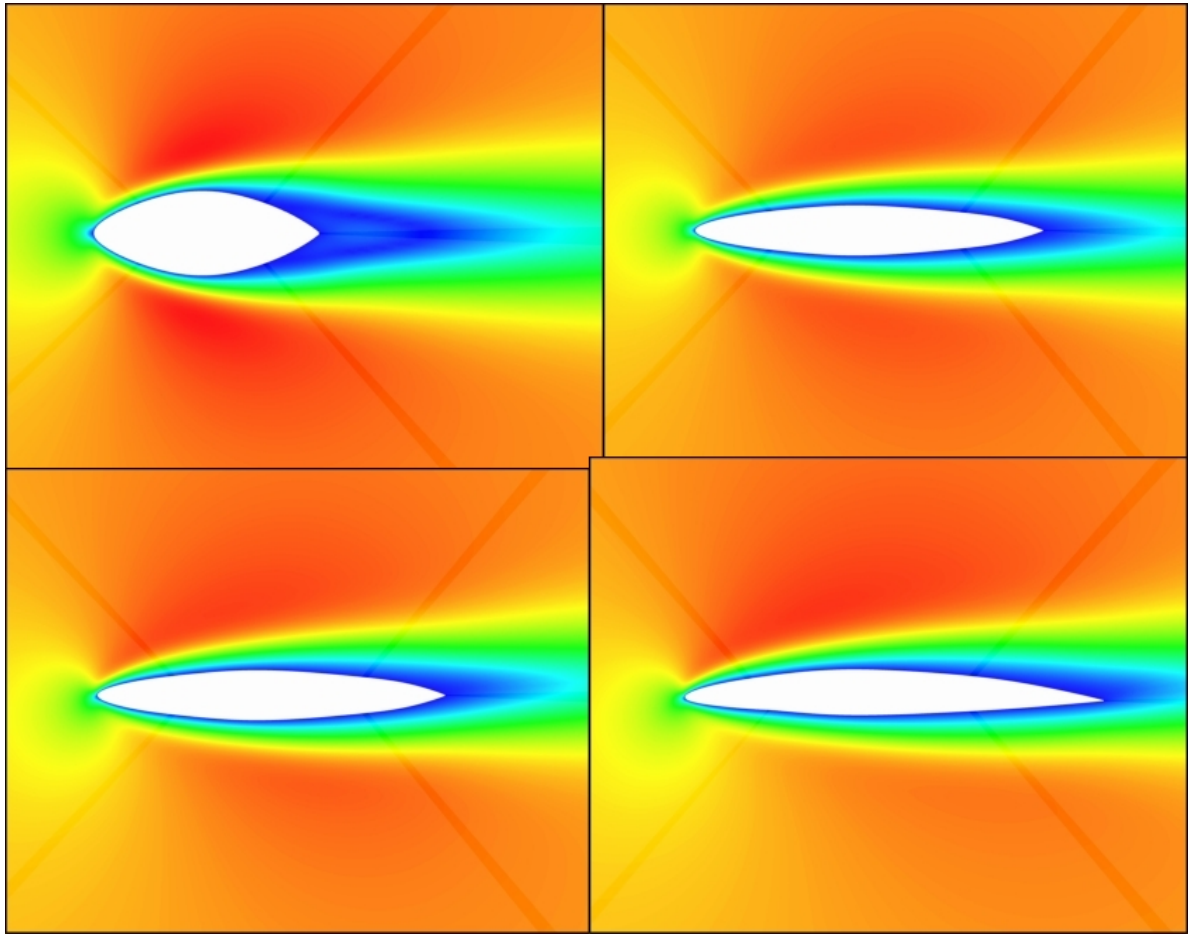
Chalmers University of Technology

2013-12-11

Presentation outline

- Introduction to the softwares
- The coupling
- Some important new files
- Run the case and look at the source code

Shape optimization



Shape optimizing capabilities in Dakota

- General optimization tool
- File and command based
- Only needs to be able to set input and get a response

Shape optimizing capabilities in OpenFOAM

- Motion solvers for dynamic meshes
- Parameterizing mesh generation
- Importing mesh from other softwares

What's missing?

A way for Dakota to control OpenFOAM

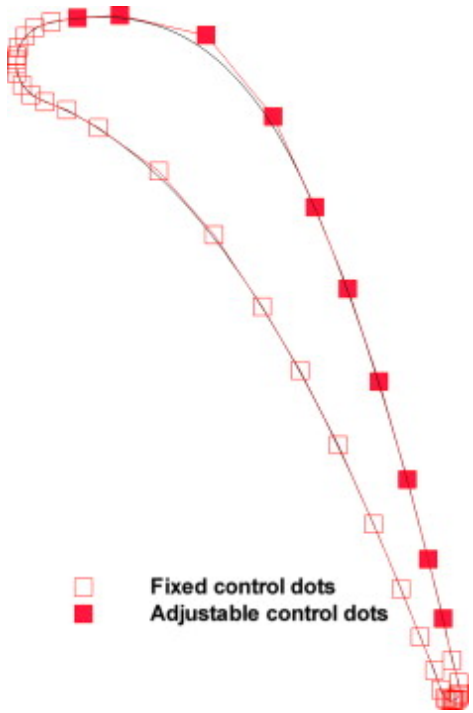
A way to change the shape of an object in OpenFOAM

Motion solver approach

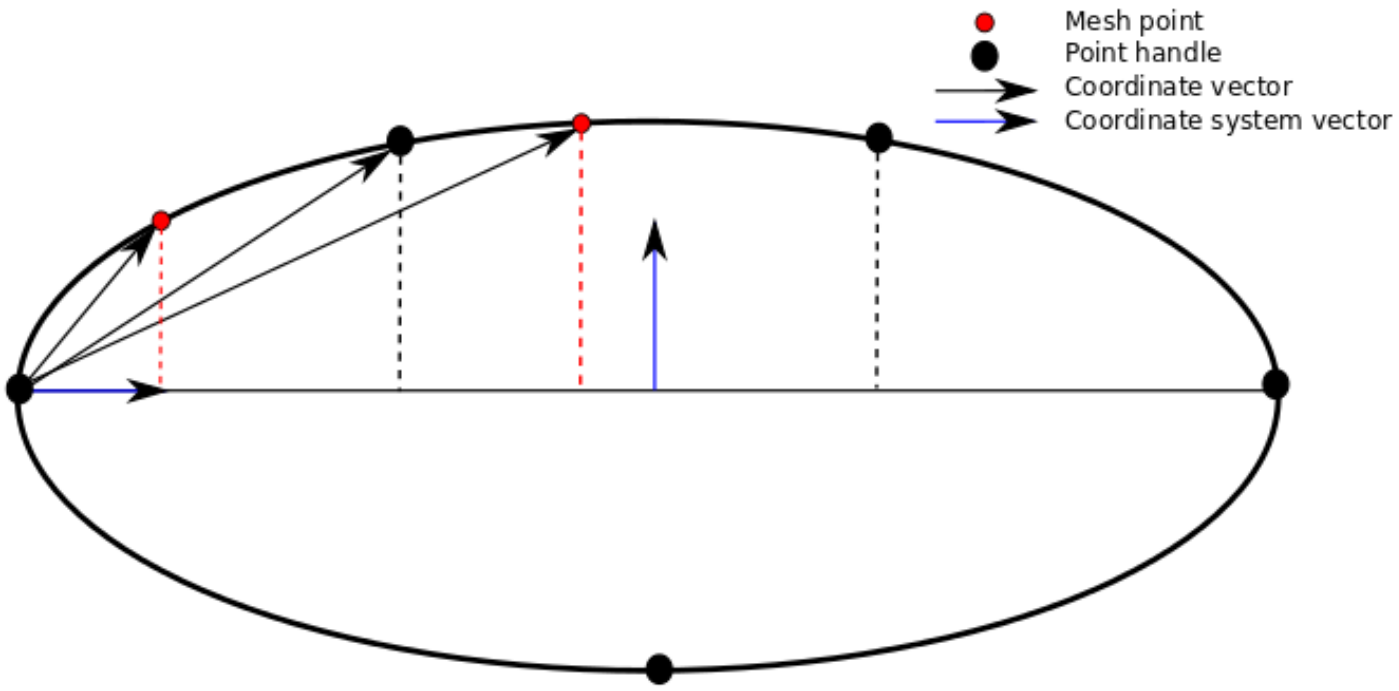
- Applying displacement on the boundary
- Benefits
  - Save meshing time
  - Great versatility
- Drawbacks
  - Possibly bad mesh

Parameterizing the shape

- Moving the boundary and using the built in motion solvers for the internal mesh
- Moving via prespecified motion handles (or points)
  - Determining the influence region
  - Determining the influence amount



Determining the influence of a motion handle



Downloading the case

Start by loading the `OF16ext` environment

Move the downloaded `pointCTut.tar` file to `$WM_PROJECT_USER_DIR`

Unpack and make:

```
tar -xvf pointCTut.tar
cd src
wmake libso
run
cd pointCTut
```

Description of the files

Files associated with Dakota:

- `cyl_show.in`
- `cyl_grad.in`
- `a_drive`

Files associated with OpenFOAM:

- `temp_dir`

The other are utility files for plotting and cleaning up!

Looking at some important new files

pointDisplacement (vim temp_dir/0/pointDisplacement):

```
    cylinder
    {
    type pointControlledDisplacement;
    value uniform (0 0 0);
    pointHandles
    (
      (-0.5 0 0)
      (0.5 0 0)
      (-0.4 0.25 0)
      (0.4 0.25 0)
      (0 0.5 0)
    );
    handleRelations
    (
      // (x y z), x-left boundary point, y-right boundary point, z-1=fix,0=variable
      (0 1 1)
      (0 1 1)
      (0 3 0)
      (2 1 0)
      (0 1 0)
    );
  }
```

Looking at some important new files

`cyl_show.in (vim cyl_show.in):`

```
# Usage:
#   dakota -i xxx.in -o run.out > stdout.out

strategy,
    graphics
    tabular_graphics_data
        tabular_graphics_file = 'table_out.dat'
    single_method

method,
  multidim_parameter_study
    partitions = 1 1 1

model,
  single

variables,
  continuous_design = 3
    lower_bounds   -0.1   -0.1   -0.4
    upper_bounds   0.1    0.1    0.4
    descriptors    'x1'   'x2'   'x3'
```

Looking at some important new files

`cyl_show.in (vim cyl_show.in):`

```
interface,
  fork
    analysis_driver = 'a_driver'
    parameters_file = 'params.in'
    results_file    = 'results.out'
    work_directory directory_tag
    template_directory = 'temp_dir'
    named 'workdir' file_save  directory_save

responses,
  num_objective_functions = 1
  no_gradients
  no_hessians
```

Looking at some important new files

`a_driver (vim a_driver):`

```
#!/bin/sh

# $1 is params.in FROM Dakota
# $2 is results.out returned to Dakota

# Replacing the variables with the current values
dprepro $1 pointMove.template pointMove.in

# Setting up the OpenFOAM run
cp pointMove.in 0.1/pointMove
icoDyMFoam > log

# Calculating output to Dakota
./calc_out.py $2
```

Looking at some important new files

pointMove.template (vim temp_dir/pointMove.template):

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                      |                                            |
| \\      /  F ield              | OpenFOAM Extend Project: Open Source CFD   |
|  \\    /   O peration          | Version:  1.6-ext                          |
|   \\  /    A nd                | Web:      www.extend-project.de            |
|    \\/     M anipulation       |                                            |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      pointMove;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

displacement
(
  (0 0 0)
  (0 0 0)
  (-{x1} {x1} 0)
  ({x2} {x2} 0)
  (0 {x3} 0)
);
```

Looking at some important new files

calc_out.py (vim temp_dir/calc_out.py):

```
#!/usr/bin/python

import numpy as np
import sys

arr=np.loadtxt('forces/0.1/forceCoeffs.dat', delimiter='\t')
cd=np.mean(arr[10:,1]);

f=open(str(sys.argv[1]),"w");
f.write(str(cd))
f.close();
```

Let's run the case

```
dakota -i cyl_show.in -o run.out
```

Thank you for your time!

Questions?