Pitz-Daily Turbulence Case

Jonathan Russell

Content

1) Description of the case

The goal of this simulation is to replicate experiment performed in 1983 by Robert W Pitz and John W Daily in which they aimed to assess the effect of combustion on the mean flowfield properties such as mixing layer growth, entrainment rate, and reattachment length. The turbulent mixing layer formed at the rearward facing step after combustion will be what is simulated and the data will be gathered in a similar fashion.

2) Hypothesis

Using the turbulent solvers made available in the open source software OpenFOAM, the experimentally gathered results from Pitz and Daily will be recreated in a simulation. The simulation consists of

- Incompressible flow
- Turbulent flow
- Bidimensional flow
- Viscous flow
- Steady flow

3) Physics of the problem

The simulation will occur in a two-dimensional mesh consisting of a short inlet, a backward facing step, and a converging nozzle as the outlet as shown below.

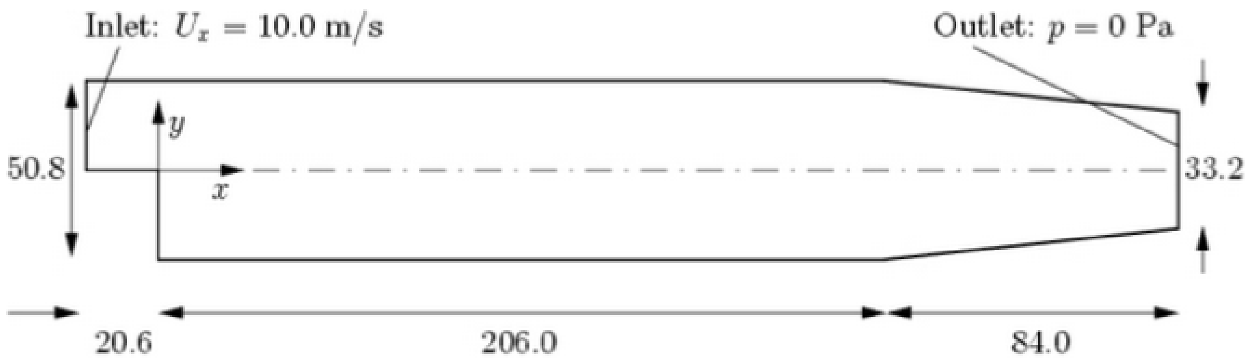

Figure 1: Representation of the model to be used (dimensions in mm)

The governing equations for the problem are as follows:

Mass continuity for incompressible flow

$$\nabla \bullet \mathrm{U} = 0$$

Steady flow momentum equation

$$\nabla \bullet (\mathrm{UU}) + \nabla \bullet \mathrm{R} = -\nabla p$$

Where p is the kinematic pressure, and **R** is the viscous stress term with an effective kinematic viscosity calculated from the transport and turbulence models.

Initial Conditions

- U = 0 m/s
- p = o Pa

Boundary Conditions

- U_{inlet} = (10,0,0) m/s
- p_{outlet} = 0 Pa

Transport Properties

- kinematic viscosity = 14.0 μm^2/s

Turbulence Model

- k-epsilon solver
- Coefficients
  - $C_\mu = 0.09$, C1 = 1.44, C2 = 1.92, α_k=1 $\alpha_{epsilon}$ = 0.76

Solver

- pisoFoam, Large Eddy Simulation

Firgure 2: Image of the initial condition (notice the slight increased velocity on the left)

4) Pre-processing

Directories	Constant	0	System
Sub-directories	transportProperties, turbulenceProperties, polyMesh	U, p, nut, nuTilda, k	blockMeshDict, fvSchemes, fvSolution

This is the file structure in which the simulation will be created in. The constant folder sets up the values for coefficients that will be used in the equations the solver uses, the 0 folder contains the initial/boundary conditions, and the system folder has the configuration files for how the mesh is made and how the solver will be executed. The preprocessing involves establishing proper settings for these files based on what scenario is being run. In this case, the values will be chosen to most closely match those given by the Pitz and Daily paper. As well, the data gathering will be performed in a way comparable to the results gathered from the paper to allow for good comparisons.

4a) Mesh Generation

The following code is the blockMeshDict file.

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  4.1                                   |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      blockMeshDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

convertToMeters 0.001;

vertices
(
    (-20.6 0 -0.5)
    (-20.6 3 -0.5)
    (-20.6 12.7 -0.5)
    (-20.6 25.4 -0.5)
    (0 -25.4 -0.5)
    (0 -5 -0.5)
    (0 0 -0.5)
    (0 3 -0.5)
    (0 12.7 -0.5)
    (0 25.4 -0.5)
    (206 -25.4 -0.5)
    (206 -8.5 -0.5)
    (206 0 -0.5)
    (206 6.5 -0.5)
    (206 17 -0.5)
    (206 25.4 -0.5)
    (290 -16.6 -0.5)
    (290 -6.3 -0.5)
    (290 0 -0.5)
    (290 4.5 -0.5)
    (290 11 -0.5)
    (290 16.6 -0.5)
    (-20.6 0 0.5)
    (-20.6 3 0.5)
    (-20.6 12.7 0.5)
    (-20.6 25.4 0.5)
    (0 -25.4 0.5)
    (0 -5 0.5)
    (0 0 0.5)
    (0 3 0.5)
    (0 12.7 0.5)
    (0 25.4 0.5)
```

```
    (206 -25.4 0.5)
    (206 -8.5 0.5)
    (206 0 0.5)
    (206 6.5 0.5)
    (206 17 0.5)
    (206 25.4 0.5)
    (290 -16.6 0.5)
    (290 -6.3 0.5)
    (290 0 0.5)
    (290 4.5 0.5)
    (290 11 0.5)
    (290 16.6 0.5)
);

blocks
(
    hex (0 6 7 1 22 28 29 23) (18 7 1) simpleGrading (0.5 1.8 1)
    hex (1 7 8 2 23 29 30 24) (18 10 1) simpleGrading (0.5 4 1)
    hex (2 8 9 3 24 30 31 25) (18 13 1) simpleGrading (0.5 0.25 1)
    hex (4 10 11 5 26 32 33 27) (180 18 1) simpleGrading (4 1 1)
    hex (5 11 12 6 27 33 34 28) (180 9 1) edgeGrading (4 4 4 4 0.5 1 1 0.5 1 1 1 1)
    hex (6 12 13 7 28 34 35 29) (180 7 1) edgeGrading (4 4 4 4 1.8 1 1 1.8 1 1 1 1)
    hex (7 13 14 8 29 35 36 30) (180 10 1) edgeGrading (4 4 4 4 4 1 1 4 1 1 1 1)
    hex (8 14 15 9 30 36 37 31) (180 13 1) simpleGrading (4 0.25 1)
    hex (10 16 17 11 32 38 39 33) (25 18 1) simpleGrading (2.5 1 1)
    hex (11 17 18 12 33 39 40 34) (25 9 1) simpleGrading (2.5 1 1)
    hex (12 18 19 13 34 40 41 35) (25 7 1) simpleGrading (2.5 1 1)
    hex (13 19 20 14 35 41 42 36) (25 10 1) simpleGrading (2.5 1 1)
    hex (14 20 21 15 36 42 43 37) (25 13 1) simpleGrading (2.5 0.25 1)
);

edges
(
);

boundary
(
    inlet
    {
        type patch;
        faces
        (
            (0 22 23 1)
            (1 23 24 2)
            (2 24 25 3)
        );
    }
    outlet
```

```
    {
        type patch;
        faces
        (
            (16 17 39 38)
            (17 18 40 39)
            (18 19 41 40)
            (19 20 42 41)
            (20 21 43 42)
        );
    }
    upperWall
    {
        type wall;
        faces
        (
            (3 25 31 9)
            (9 31 37 15)
            (15 37 43 21)
        );
    }
    lowerWall
    {
        type wall;
        faces
        (
            (0 6 28 22)
            (6 5 27 28)
            (5 4 26 27)
            (4 10 32 26)
            (10 16 38 32)
        );
    }
    frontAndBack
    {
        type empty;
        faces
        (
            (22 28 29 23)
            (23 29 30 24)
            (24 30 31 25)
            (26 32 33 27)
            (27 33 34 28)
            (28 34 35 29)
            (29 35 36 30)
            (30 36 37 31)
            (32 38 39 33)
            (33 39 40 34)
            (34 40 41 35)
            (35 41 42 36)
            (36 42 43 37)
```

```
            (0 1 7 6)
            (1 2 8 7)
            (2 3 9 8)
            (4 5 11 10)
            (5 6 12 11)
            (6 7 13 12)
            (7 8 14 13)
            (8 9 15 14)
            (10 11 17 16)
            (11 12 18 17)
            (12 13 19 18)
            (13 14 20 19)
            (14 15 21 20)
        );
    }
);

mergePatchPairs
(
);

// ************************************************************************* //
```

In OpenFOAM the mesh is required to be three-dimensional which is why under the vertices section all the values contain three entries. This will be ignored for the most part and the z dimension will be very small and the mesh will be only one unit in that direction.

To start the code, since the values given have been in millimeters, the convertToMeters function is called so all the millimeter positions will be properly used as meters in calculations.

The vertices section creates points in space at the given coordinates, anywhere there is a discontinuity or a new definition in the initial/boundary conditions a vertex will have to be placed.

The blocks code the creates the divisions used in the mesh. In this case, locations away from the edges all have a uniform grading, meaning the mesh is equally divided into (x y z) number of divisions. As can be seen in the code above, the mesh is more defined in areas of interest such as edges and the backward facing step.

The boundary section establishes which vertices are the outer boundaries of the wall, and if there are certain boundary conditions that will be applied to them (i.e. inlet, outlet, interface).

4b) Initial/Boundary Conditions

As stated earlier the parameters in these next files were made to be as similar to the Pitz-Dialy case as possible

Initial pressure file

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  4.1                                   |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       volScalarField;
    object      p;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

dimensions      [0 2 -2 0 0 0 0];

internalField   uniform 0;

boundaryField
{
    inlet
    {
        type            zeroGradient;
    }

    outlet
    {
        type            fixedValue;
        value           uniform 0;
    }

    upperWall
    {
        type            zeroGradient;
    }

    lowerWall
    {
        type            zeroGradient;
    }

    frontAndBack
    {
        type            empty;
    }
}

// ************************************************************************* //
```

This is the Velocity initial condition file

```
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  4.1                                   |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       volVectorField;
    object      U;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

dimensions      [0 1 -1 0 0 0 0];

internalField   uniform (0 0 0);

boundaryField
{
    inlet
    {
        type            turbulentInlet;
        referenceField  uniform (10 0 0);
        fluctuationScale (0.02 0.01 0.01);
        value           uniform (10 0 0);
    }

    outlet
    {
        type            inletOutlet;
        inletValue      uniform (0 0 0);
        value           uniform (0 0 0);
    }

    upperWall
    {
        type            noSlip;
    }

    lowerWall
    {
        type            noSlip;
    }

    frontAndBack
    {
        type            empty;
    }
}

// ************************************************************************* //
```

The inlet section was made to act as if it were the turbulent inlet velocity from the paper. It was treated as if it were a uniform velocity from top to bottom. The outlet is outputting to the atmosphere so it is initially static.

This is the initial turbulent kinetic energy (k) file.

```
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  4.1                                   |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       volScalarField;
    object      k;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

dimensions      [0 2 -2 0 0 0 0];

internalField   uniform 0;

boundaryField
{
    inlet
    {
        type            fixedValue;
        value           uniform 2e-05;
    }

    outlet
    {
        type            inletOutlet;
        inletValue      uniform 0;
        value           uniform 0;
    }

    upperWall
    {
        type            fixedValue;
        value           uniform 0;
    }

    lowerWall
    {
        type            fixedValue;
        value           uniform 0;
    }

    frontAndBack
    {
        type            empty;
    }
}

// ************************************************************************* //
```

the turbulent kinetic energy was set to be zero except at the inlet where it is assumed to be isotropic and have initially small fluctuations, hence the small value of k.

This is the turbulent viscosity file (note: nuT is used for LES and nuTilda is used for RAS)

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  4.1                                   |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       volScalarField;
    object      nut;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

dimensions      [0 2 -1 0 0 0 0];

internalField   uniform 0;

boundaryField
{
    inlet
    {
        type            zeroGradient;
    }

    outlet
    {
        type            zeroGradient;
    }

    upperWall
    {
        type            zeroGradient;
    }

    lowerWall
    {
        type            zeroGradient;
    }

    frontAndBack
    {
        type            empty;
    }
}

// ************************************************************************* //
```

This establishes the viscosity to not be spatially dependent, the value for it will be calculated by the solver using the relastionship nuT = l*k^0.5

4c) Physical Properties

This file establishes how the turbulence will be modeled by the simulation

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  4.1                                   |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "constant";
    object      turbulenceProperties;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

simulationType  LES;

LES
{
    LESModel        dynamicKEqn;

    turbulence      on;

    printCoeffs     on;

    delta           cubeRootVol;

    dynamicKEqnCoeffs
    {
        filter simple;
    }

    cubeRootVolCoeffs
    {
        deltaCoeff      1;
    }

    PrandtlCoeffs
    {
        delta           cubeRootVol;
        cubeRootVolCoeffs
        {
            deltaCoeff      1;
        }

        smoothCoeffs
        {
            delta           cubeRootVol;
            cubeRootVolCoeffs
```

```
            {
                deltaCoeff      1;
            }

            maxDeltaRatio   1.1;
        }

        Cdelta          0.158;
    }

    vanDriestCoeffs
    {
        delta           cubeRootVol;
        cubeRootVolCoeffs
        {
            deltaCoeff      1;
        }

        smoothCoeffs
        {
            delta           cubeRootVol;
            cubeRootVolCoeffs
            {
                deltaCoeff      1;
            }

            maxDeltaRatio   1.1;
        }

        Aplus           26;
        Cdelta          0.158;
    }

    smoothCoeffs
    {
        delta           cubeRootVol;
        cubeRootVolCoeffs
        {
            deltaCoeff      1;
        }

        maxDeltaRatio   1.1;
    }
}


// ************************************************************************* //
```

In this case Large Eddy Simulation (LES) is used as RAS requires large time steps and this problem is on a very small time scale. Specifically, the dynamic one equation eddy-viscosity model is used.

The values of coefficients and ratios were chosen based on other similar simulations of similar space-time scale.

The solution is using a dynamic subgrid-scacle model which computes coefficients dynamically as the simulation occurs, but a drawback of such is that it performs poorly near walls or transitional regimes.

This is the file that defines the values of viscosity for the simulation

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  4.1                                   |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "constant";
    object      transportProperties;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

transportModel  Newtonian;

nu              [0 2 -1 0 0 0 0] 1e-05;

// ************************************************************************* //
```

4d) Control Case

controlDict (probes, uniformsomething something), fvSchemes, fvSolution

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  4.1                                   |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      controlDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

application     pisoFoam;

startFrom       startTime;

startTime       0;

stopAt          endTime;

endTime         0.1;

deltaT          2e-05;

writeControl    timeStep;

writeInterval   10;

purgeWrite      0;

writeFormat     ascii;

writePrecision  6;

writeCompression off;

timeFormat      general;

timePrecision   6;

runTimeModifiable true;

functions
{
    probes
    {
        type            probes;
```

```
        libs            ("libsampling.so");
        writeControl    timeStep;
        writeInterval   1;

        fields
        (
            p
        );

        probeLocations
        (
            (0.0254 0.0253 0)
            (0.0508 0.0253 0)
            (0.0762 0.0253 0)
            (0.1016 0.0253 0)
            (0.127 0.0253 0)
            (0.1524 0.0253 0)
            (0.1778 0.0253 0)
        );

    }

    fieldAverage1
    {
        type            fieldAverage;
        libs            ("libfieldFunctionObjects.so");
        writeControl    writeTime;

        fields
        (
            U
            {
                mean        on;
                prime2Mean  on;
                base        time;
            }

            p
            {
                mean        on;
                prime2Mean  on;
                base        time;
            }
        );
    }

    surfaceSampling
    {
```

```
        // Sample near-wall velocity

        type surfaces;

        // Where to load it from (if not already in solver)
        libs            ("libsampling.so");
        writeControl    writeTime;

        interpolationScheme cellPoint;

        surfaceFormat vtk;

        // Fields to be sampled
        fields
        (
            U
        );

        surfaces
        (
            nearWall
            {
                type            patchInternalField;
                patches         ( lowerWall );
                distance        1E-6;
                interpolate     true;
                triangulate     false;
            }
        );
    }

    #includeFunc scalarTransport
}

// ************************************************************************* //
```

This file allows most of the control of the simulation's precision. As per the experiment, the simulation should take about 0.1 seconds. The write interval will determine how many times the data will be recorded throughout the 0.1 seconds, and the precision of the data recorded can also be altered with writeprecision and timeprecision. Choosing a proper deltaT is important, in this case based on the chosen division of spatial parameters, the maximum deltaT is as chosen (2e-05), but if more time temporal resolution is desired this can be decreased, but it comes at a significant simulation time cost.

In the paper, several probes were used to record the data, so instead of comparing the overall data from the simulation, looking at probes placed through the tube on the same axial point will be beneficial. The probes record the data at the defined cell throughout the entire simulation and is output to a file displaying the resulting pressure over time. As well, the fieldaverage method can be used to obtain the average values for velocity along the lowerWall over time. This can be compared to the velocity profiles gathered by Pitz and Daily and it is important because the solver is known to be weaker near the edges, so any validation here would hold strong weight.

This is the fvSchemes file

```
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  4.1                                   |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      fvSchemes;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

ddtSchemes
{
    default         backward;
}

gradSchemes
{
    default         Gauss linear;
}

divSchemes
{
    default         none;
    div(phi,U)      Gauss LUST grad(U);
    div(phi,k)      Gauss limitedLinear 1;
    div(phi,s)      bounded Gauss limitedLinear 1;
    div((nuEff*dev2(T(grad(U))))) Gauss linear;
}

laplacianSchemes
{
    default         Gauss linear corrected;
}

interpolationSchemes
{
    default         linear;
}

snGradSchemes
{
    default         corrected;
}


// ************************************************************************* //
```

The schemes used were chosen based upon other simulations of turbulent incompressible flow.

This is the fvSolution file

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  4.1                                   |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      fvSolution;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

solvers
{
    p
    {
        solver          GAMG;
        tolerance       1e-06;
        relTol          0.1;
        smoother        GaussSeidel;
    }

    pFinal
    {
        $p;
        smoother        DICGaussSeidel;
        tolerance       1e-06;
        relTol          0;
    }

    "(U|k|B|nuTilda|s)"
    {
        solver          smoothSolver;
        smoother        GaussSeidel;
        tolerance       1e-05;
        relTol          0;
    }
}

PISO
{
    nCorrectors     2;
    nNonOrthogonalCorrectors 0;
}


// ************************************************************************* //
```

For the turbulent solver a tolerance is required for some of the variables mentioned in the initial conditions and some specific to the turbulence calculations, for which 1e-05 is acceptable, though a slightly more tolerance is desired for p, pFinal especially since it has a low impact on computational time unlike U,k,B,nuT and s. The ncorrectors will determine how many times the pressure equation and momentum corrector, increasing this will have a significant computational time cost. The nNonOrthogonalCorrectors is usually set to 0 for steady-state.

5) Running the Case

To run the case, go back into the /$FOAM_RUN/pitz folder and run

blockMesh

checkMesh

pisoFoam

The checkMesh is just to ensure there are no glaring issues in the definition of the mesh and after running pisoFoam it should take a while (anywhere from 10 minutes to many hours depending on the precision, and space-time scale).

6) Post Processing

Once the simulation is finished the output files from the probes, field averaging and p/U files will be put into folders for the specific times at which they occurred (i.e. the final result will be in pitz/0.1. To view the overall simulation type paraFoam and the paraFoam data viewer will open and create a directory containing the data from the pitz case. To view this click on the pitz.openfoam in the left window and click apply. The initial time should now be shown. In the scroll down menu p, or U can be selected to see how they looked initially (remember pressure was set to be 0 so nothing should be notable about that).

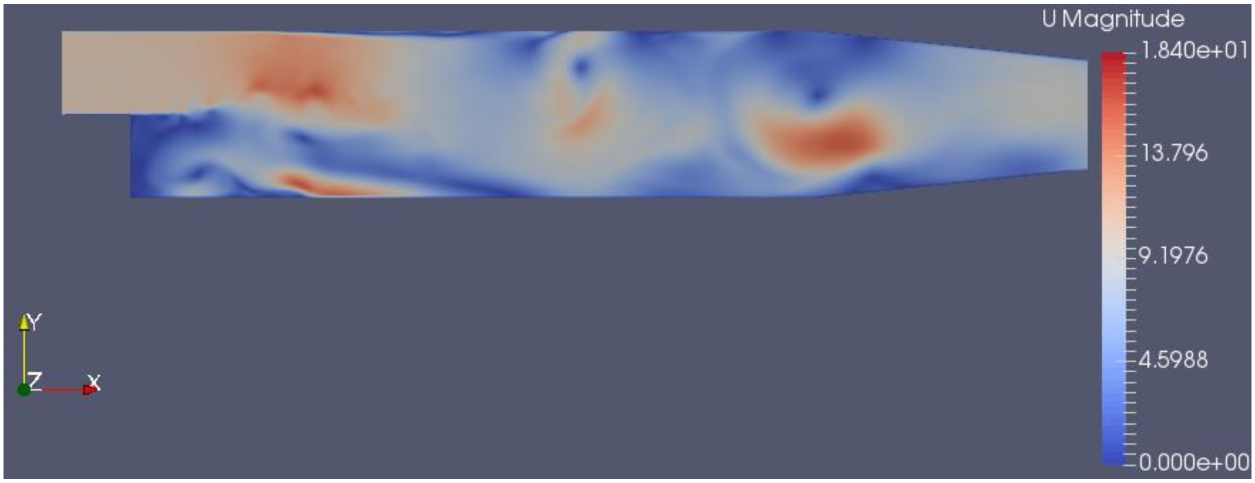

Figure 3: Image of the velocity distribution at the final time

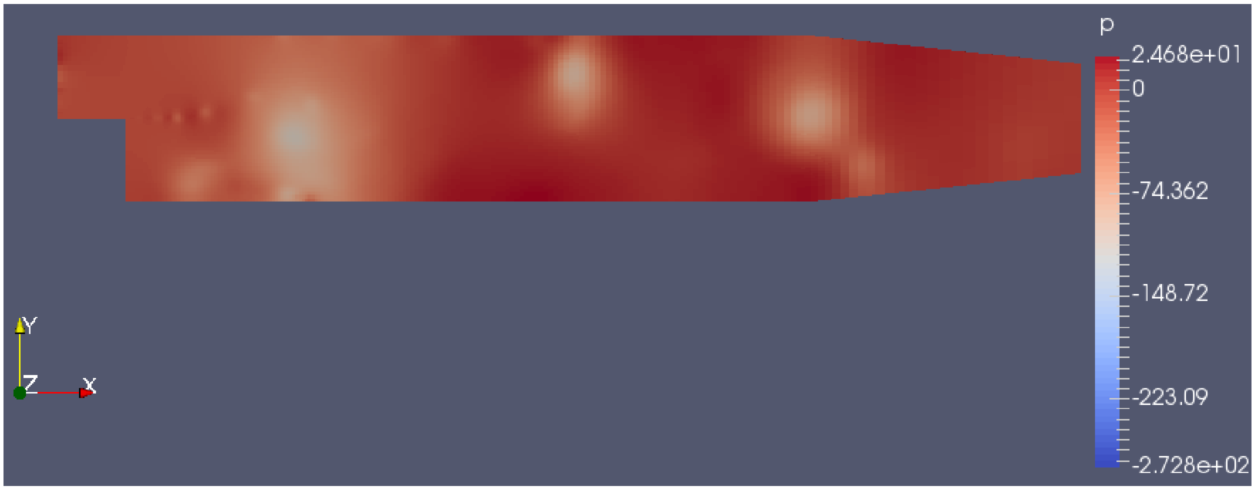

Figure 4: Image of the final pressure distribution