POLITECNICO DI TORINO

Master of Science in Automotive Engineering

Master Thesis

Design and CFD analysis performance verification of an intake manifold for motorsport application

Supervisors:
Prof. Daniela Misul
Prof. Mirko Baratta
Ing. Massimiliano Garavaldi

Author:

Matteo Zambianchi

March 2021

ABSTRACT

In January 2020 a new project started in Autotecnica Motori: the development of a new V6 turbocharged spark ignition engine for motorsport application. The aim of this thesis project is to understand, by means of computational fluid dynamics simulations, which is the best intake manifold configuration for this engine. Constrained from the engine assembly, only two possible intake manifold configurations are possible: the first one with an air flow that enters in the intake manifold aligned with the axis of the engine bank, and the second one with an air flow that enters in the intake perpendicular to the engine bank. The aim is to evaluate the mass flow rate unbalance between the 3 cylinders of the same engine bank for each one of the two intake manifold geometries, and then to compare the two geometries in order to understand the configuration that, for the same pressure drop, provides the higher mass flow. Two different geometries have been produced and then analyzed by means of the open source CFD code OpenFOAM. At first, the steady-state model has been validated on an already existing intake manifold. By means of experimental test at the flow bench of Autotecnica Motori, it collected the mass flow rates for different pressure drops. Once the model has been validated, it has been transferred to the two produced geometry, that have been analyzed for each runner. For each single runner a characteristic flow rate versus pressure drop has been built: the analysis of the results is performed in terms of comparison of the characteristics of each single runner of the same intake manifold configuration, and then comparing the same runner for the two different configurations. Consequently, a transient CFD model has been produced. The transient model validation is complex: the idea is to make a comparison of the different lambda of each cylinder with the target lambda by endowing an already existing engine with lambda sensors and thermocouples on the exhaust manifold. Then it compares the percentage of variation of mass flow measured at the test bench and the percentage variation of mass flow obtained from the simulation. This procedure, in terms of a transient CFD model validation is of difficult application because the CFD model involves only the intake manifold, and this separation of the intake manifold from the whole engine system neglects causes that strongly affects the engine cylinder filling, like for example the intake ducts, the valve-train distribution system and the in-cylinder turbulence: it would be necessary to simulate the whole engine system in order to be sure to take in account all the possible factors that affects a possible unbalance. Anyway, the transient simulation has been performed on both the configurations for two fixed engine working points: the maximum torque and the maximum power. The steady-state simulation result analysis points out how the configuration

with an air flow aligned with the engine bank is the best one: the flow is better aligned towards the runners and the mass flow rate are consistently higher with respect to the second solution. The transient model confirms these considerations: the solution with an air flow aligned with the engine bank axis is the most efficient one, but differently from the steady-state simulations, the transient one can perceive, for each intake manifold geometry, a bigger unbalance in terms of integral average percentage variation of mass flow.

Summary

1 Computational Fluid Dynamics Introduction

The computational fluid dynamics simulations are part of the CAE process and involves numerical analysis and algorithms to solve fluid dynamics problems by means of a computer. The main usage of the CFD is to solve the Navier-Stokes equations: the analytical solution of these equations is possible only in simple laminar flow cases and with simple geometries, but in real cases it deals with turbulent flow that require a numerical approach.

It is possible to locate four different steps that are part of a computational fluid dynamic simulation:

- Choice of the mathematical model, that usually corresponds to the set of Navier-Stokes equations. It is a system of nonlinear partial differential equations since it is characterized by turbulence, shock waves so discontinuous variable fields, spontaneous unsteadiness of flow and non-uniqueness of the solution. Any modeling assumption is characterized by a level of empiricism, but it always contains a level of error if compared to the real world.

- Discretization phase: it involves both the domain discretization, so the definition of the calculation grid, and the equation discretization, therefore the definition of the numerical method. The discretization is the translation of geometrical and mathematical models into numbers: the set of points or elements that replace the continuity of the real space is called grid or mesh. The shape of the mesh has issues related to the numeric, hence the grid generation process is a delicate and time-consuming step. Once the grid is generated, all the mathematical operators, such as partial derivatives of the quantities, will have to be transformed into arithmetic operations on the mesh point values. There are 2 different approaches that involves the discretization phase: the first one is the *Finite Difference Discretization*, in which the quantities are defined over the points:

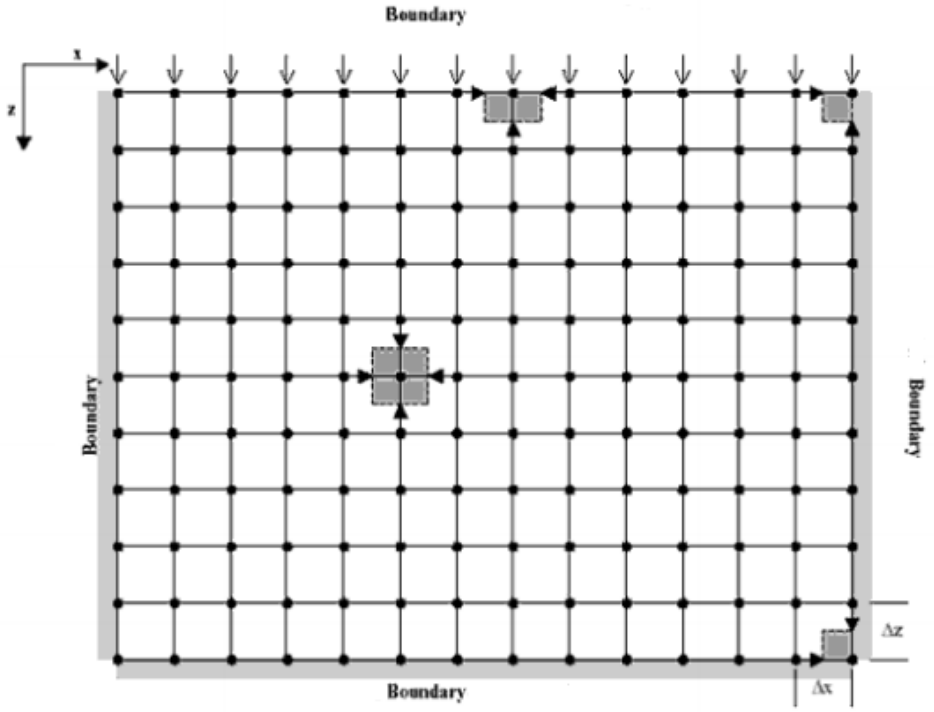


Figure 1.Finite Difference Discretization Schemes.

The second approach is the *Finite Volume Discretization*, in which the quantities are defined over the volume or referred to its centroid:

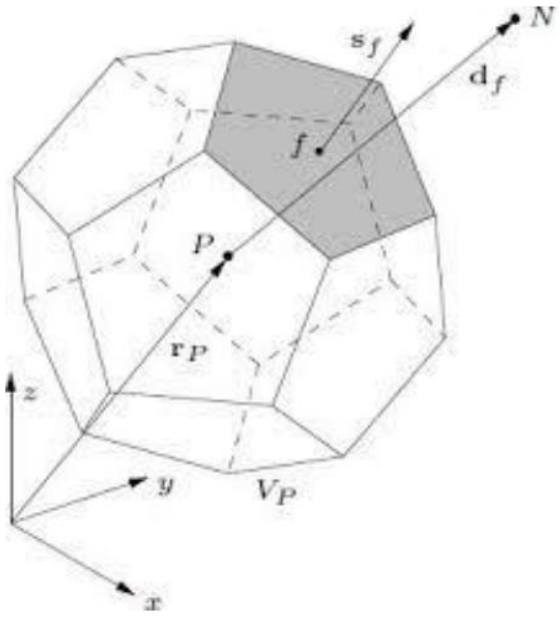


Figure 2. Finite Volume Discretization.

- Analysis of the chosen numerical scheme in terms of stability and accuracy: the numerical schemes establish a relation between neighboring mesh points. The numerical schemes have to satisfy three conditions: stability, convergence and consistence. Considering the linear convection equation:

$$\frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} = 0 \tag{1.1}$$

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} + a\frac{u_{i+1}^n - u_{i-1}^n}{2\Delta x} = 0 \tag{1.2}$$

The equation 1 is the linear convection equation in differential form while the equation 2 is the linear convection equation in discretized form. Considering the two equation, consistency is a condition of the numerical scheme and states that the numerical schemes must tend to the differential equation when the space and time discretization tend to zero. Stability is a condition on the numerical scheme and imposes that the round-off error due to the finite algebra remains bounded iteration after iteration:

$$\overline{\varepsilon}_n^i = u_i^n - \tilde{u}_i^n \tag{1.3}$$

$$\lim_{n\to+\infty} \overline{\varepsilon}_n^i \leq \mathrm{K} \tag{1.4}$$

Where $\tilde{u}$ represents the exact solution of the differential equation while $\widetilde{u_i}$ represents the exact solution of the discretized equation.

The convergence is a condition on the numerical solution and requires that the output of the simulation is a correct representation of the model it is solving. The numerical solution must tend to the exact solution of the mathematical problem when the space and the time discretization tend to zero:

$$\tilde{\varepsilon}_i^n = u_i^n - \tilde{u}(i\Delta x, n\Delta t) \tag{1.5}$$

$$\lim_{\Delta x, \Delta t \to 0} |\tilde{\varepsilon}_i^n| = 0 \tag{1.6}$$

- Definition of the solution of the problem: it corresponds to the choice of the time integration and of the matrix solution method. The solution algorithm depends on the type of the problem it is simulating: time-dependent, in which the field variables changes not only from point to point inside the domain but also change in each point by changing the time; steady-state flow, where the field variables change from point to point but remain the same in each point by changing the time.

1.1 Conservation laws

Computational fluid dynamics is based on the formulation of conservation of laws. Three equations can completely determine the behavior of a system:

- Conservation of mass: continuity equation. It is a conservation equation that is referred to a scalar property of the fluid ρ:

$$\frac{\partial \rho}{\partial t} + \vec{\nabla}(\rho \overrightarrow{U}) = 0 \tag{1.1.1}$$

- Conservation of momentum: momentum equation, that is referred to a vectorial property of the fluid $\varphi = \overrightarrow{\rho U}$. To define the momentum equation, it has to assume that the fluid is Newtonian, that means:

$$\bar{\bar{\sigma}} = -p\bar{\bar{I}} + \bar{\bar{\tau}} \tag{1.1.2}$$

Where $-p\overline{I}$ is the isotropic pressure component and $\overline{\tau}$ is the viscous shear stress tensor, and it represents the internal friction force of fluid layers against each other. Inside the control volume the internal forces cancel out while they do not have any counterpart on the volume surface. The integral momentum conservation equation referred to a control volume ω is:

$$\frac{\partial}{\partial} \int_V \rho \vec{U} \mathrm{d}\omega + \oint_S \rho(\vec{U}\vec{U}) \mathrm{d}\vec{S} = \int_V \rho \vec{f_e} d\omega + \oint_S \overline{\sigma}\, \mathrm{d}\vec{S} \tag{1.1.3}$$

By applying the Gauss theorem and considering the differential formulation, it gets:

$$\frac{\partial(\rho\vec{U})}{\partial t} + \vec{\nabla}\left(\rho\vec{U}\vec{U} + p\bar{\bar{I}} - \bar{\bar{\tau}}\right)\mathrm{d}\omega = \rho\vec{f_e} \tag{1.1.4}$$

- Conservation of energy: energy equation. The conserved quantity is the total energy, defined as the sum of the fluid internal energy plus its kinetic energy per unit of mass:

$$\mathrm{E} = e + \frac{1}{2}\vec{U}^2 \tag{1.1.5}$$

For the energy it considers two types of fluxes, convective and diffusive:

$$\overrightarrow{F_c} = \rho E\vec{U} \tag{1.1.6}$$

$$\overrightarrow{F_d} = -k\vec{\nabla}T \tag{1.1.7}$$

The volume source terms are the work of the volume forces f_e and the heat transfer sources (radiation, reactions or electrical resistance):

$$Q_v = \rho\overrightarrow{f_e}\vec{U} + q_H \tag{1.1.8}$$

The last term refers to the surface sources $\overrightarrow{Q_s}$, that are the result of the work done on the fluid by the internal shear stress acting on the surface of the control volume:

$$\overrightarrow{Q_s} = -p\vec{U} + \bar{\bar{\tau}}\vec{U} \tag{1.1.9}$$

Grouping all the energy contribution terms, it obtains the energy conservation equation (expressed in differential form):

$$\frac{\partial \rho E}{\partial t} + \vec{\nabla}\left(\rho E \vec{U}\right) = \vec{\nabla}\left(k\vec{\nabla}T\right) + \vec{\nabla}\left(\bar{\bar{\tau}}\vec{U}\right) + \rho\overrightarrow{f_e}\vec{U} + q_H \tag{1.1.10}$$

It can be useful to express the conservation equation with respect to the enthalpy:

$$\frac{\partial \rho H}{\partial t} + \vec{\nabla}\left(\rho H \vec{U} - k\vec{\nabla}T + \bar{\bar{\tau}}\vec{U}\right) = \frac{\partial p}{\partial t} + \rho\overrightarrow{f_e}\vec{U} + q_H \tag{1.1.11}$$

Grouping together the equation 7, the equation 10 and the equation 16, it gets the Navier-Stokes equation.

It still needs to define the constitutive law and the definition of fluid properties as a function of flow quantities. It will consider a perfect gas and a constant viscosity.

1.2 OpenFOAM

To perform CFD simulations, the opensource code OpenFOAM has been used. OpenFOAM is a C++ library, used primarily to create executable, called applications. The applications fall in two categories: solvers, each one designed to solve a specific problem in continuum mechanics; and utilities, that are designed to perform tasks that involve data manipulation. The OpenFOAM distribution contains numerous solvers and utilities covering a wide range of problems. The software is available only for Linux operating system or for WSL on Windows.

OpenFOAM is structured in the following way:

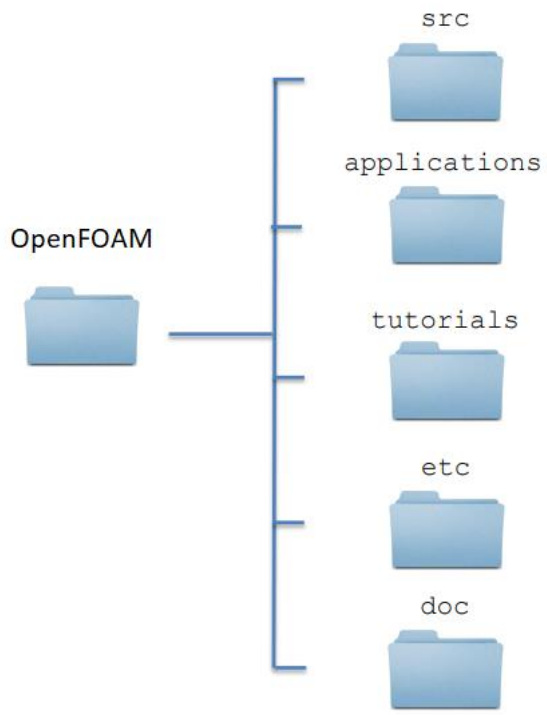


Figure 3. OpenFOAM structure.

Each one of the five folder is a fundamental part of OpenFOAM:

- Src: it contains the implementation of the OpenFOAM libraries.

- Applications: it contains solvers, utilities and tests developed based on the OpenFOAM libraries.

- Tutorials: it contains tutorials for a wide range of physical problems.

- Etc: it contains additional code for library setup, examples or useful data.

- Doc: it contains the Doxigen documentation.

The case setup is based on 3 folders, from which a certain command launched on the terminal makes the software to read in this folder and the to start the simulation:

- The first one is the folder "0": it contains all the text files that define the boundary conditions of fluid domain. Openfoam requires the specification of each one of the quantities involved in the problem because there are not predefined values assigned to each field variable. For example, for a compressible, turbulent and steady solver, it requires the pressure, the temperature, the velocity and all the turbulence parameters (depending on the type of turbulence model used).

- The second folder is the folder "*constant"*: in this folder it defines the turbulence model and the thermophysical properties of the problem.

- The third folder is the folder "*system*". In this folder there are three main text notes, the finite volume schemes (*fvSchemes*), the finiteVolume solution (*fvSolution*) and the controlDictitonary (controlDict). These three note blocks permit to set the simulation in terms of number of iteration (in case of steady-state simulation) or time of the simulation (in case of transient simulation), of solvers and numerical schemes. Moreover, the system folder contains the text note files that define the mesh parameters. Eventually, it is also possible to define functions that permits to compute, during the real time of the simulation, surface field

values like the average pressure over a boundary, streamlines, measurement probes or magnitude quantities of vectorial fields.

Figure 4. OpenFOAM case set-up folder.

2 Steady-state Model

The steady-state simulation is a simulation where the flow filed variables of the fluid domain are different from point to point of the fluid domain, but they do not vary with time. Therefore, in a steady-state simulation the concept of time is replaced by the iteration, that represents the number of times it solves the equations chosen to represent the physical phenomenon it is investigating.

The necessity to develop a steady-state model comes from the need to evaluate the pressure losses, and so the fluid dynamics resistance, of air flowing inside an intake manifold conceived for motorsport application. There are two possibilities to perform this analysis:

- First method: imposing a pressure drop between the inlet and outlet and evaluating the outgoing mass flow, therefore for the same pressure drop, the higher is the mass flow, the lower are the losses inside the intake manifold.

- Second method: imposing a mass flow and evaluating the pressure losses between inlet and outlet, so the higher will be the mass flow, the lower will be the losses inside the intake manifold.

To give validity to this model it is necessary to be sure it correctly represents the physic of the problem it is trying to reproduce by the CFD simulation. As a result, it is necessary to compare the simulation results with experimental data. To do this, it necessary a flow test bench. The flow test bench permits to gather experimental data following the first method, and therefore the geometry layout choice and the geometrical optimization will be performed evaluating the mass flow of each kind of configuration.

2.1 Flow test bench

To perform the experimental test, it used the flow bench present in AutotecnicaMotori:

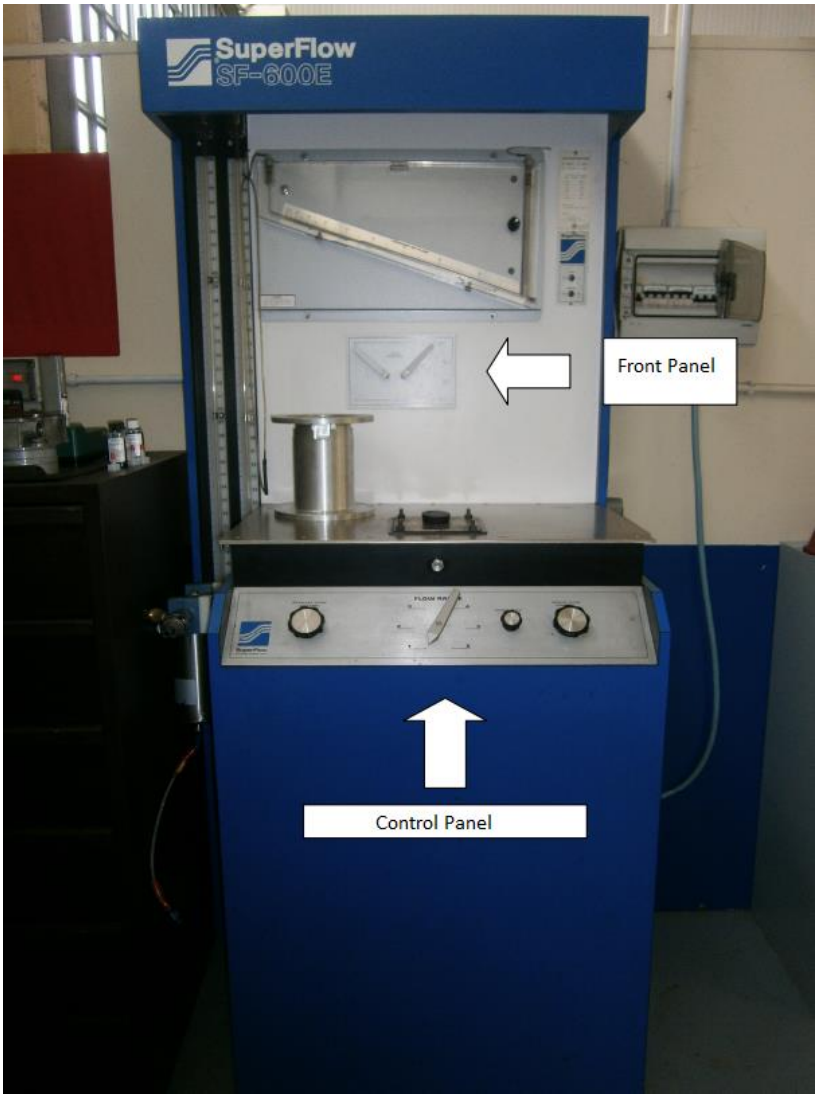


Figure 5. Flow Bench at AutotecnicaMotori.

The flow bench is controlled by means of the Control Panel and the Front Panel. On the control panel are present 4 different knobs:

- “Intake Flow Control”, “Exhaust flow Control”: this knob permits to control the test pressure, respectively of intake or exhaust.

- Range lock: it is necessary to pull up this knob when it wants to modify the range of the flow bench.

- Flow range: There are six different range of operation, and this knob permits to rotate a plate below the plane of the test bench. This plate has different holes of variable diameter, and so they permit to get different mass flow.

On the frontal panel it finds:

- “Intake”, “Exhaust” knob: this knob controls a selector behind the bench, that permits a disc rotation; this disk puts in connection different pipes that are pressure outlet for the vertical pressure gauge.

- “Intake Above 150”,” Intake Below 150”: it is used when it varies the operating range.

The test is performed by selecting a certain range by means of the flow range knob, and then by the Intake Flow Control knob it is possible to vary the test pressure. The test pressure cannot be varied arbitrarily, because there are predefined values of pressure at which correspond, for each range, a volumetric flow rate:

Table 1. Flow Value at 25''

Intake	RANGE	Exhaust
17	1	19.12
33.58	2	37.48
70.8	3	74.58
139.72	4	150.57
208.16	5	225.62
284.89	6	326.16

Table 2. Flow Value at 15"

Intake	RANGE	Intake
17.17	1	18.96
33.92	2	37.18
71.51	3	73.98
141.11	4	149.36
210.24	5	223.81
287.73	6	323.55

The tables show, respectively for 25 inches of column water and 15 inches of column water, the volumetric flow at the intake and the volumetric flow at the exhaust for the different ranges. There are different tables, from 5 inches per column water up to 48 inches per column water.

During the experimental test it has to match, by means of the Intake flow control knob, the pressure level indicated in the tables, in order to know the ideal volumetric flow that is flowing through the intake manifold. The desired pressure level can be controlled in a pressure Gauge located on the left side of the flow bench:

Figure 6. Pressure Gauge for the test pressure control.

Once reached the desired pressure level, the pressure gauge in the middle of the frontal panel measures the percentage of volumetric flow with respect to the ideal values that can be read in the tables:

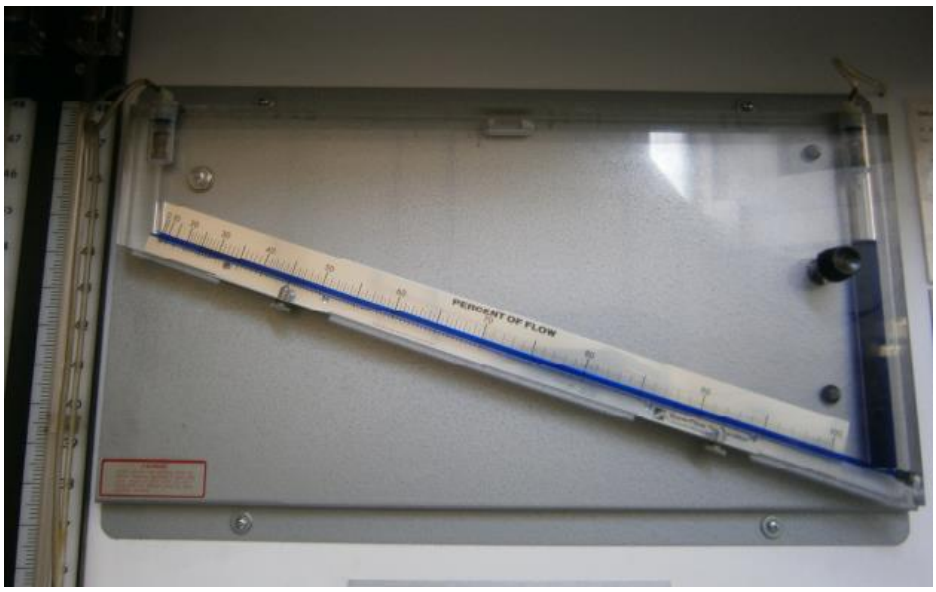


Figure 7. Pressure Gauge for the volumetric flow correction.

2.2 Set-up of the Experimental Test

The intake manifold that will be used at the test flow bench is the intake manifold of ATM-AR-F3R engine: it is a turbocharged 4-cylinders in-line spark ignition engine used in the Formula Regional Championship. The Formula Regional Championship has been introduced in the 2019 and it is an intermediate championship in between the Formula3 International and the Formula 4. The ATM-AR-F3R intake manifold is composed of a plenum made of carbon fibers and an aluminum flange at its base that is produced by CNC machine. The carbon fiber plenum is then sticked on the aluminum flange, that then is screw to the cylinder block. The picture below shows the ATM-AR-F3R intake manifold cad:

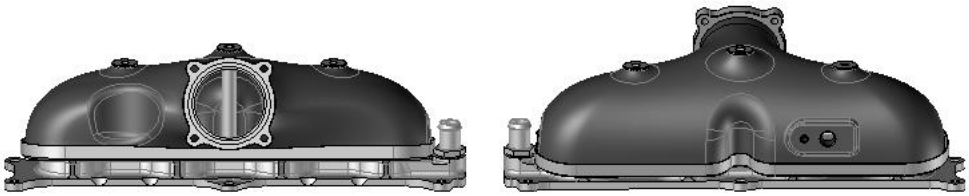


Figure 8. ATM-AR-F3R intake manifold.

To perform the test, it is necessary to create a joint to connect the intake manifold to the bench, but also 3 plugs to close the remaining runners: these components have been designed and then 3D printed.

The following pictures show the intake manifold installation on the test bench:

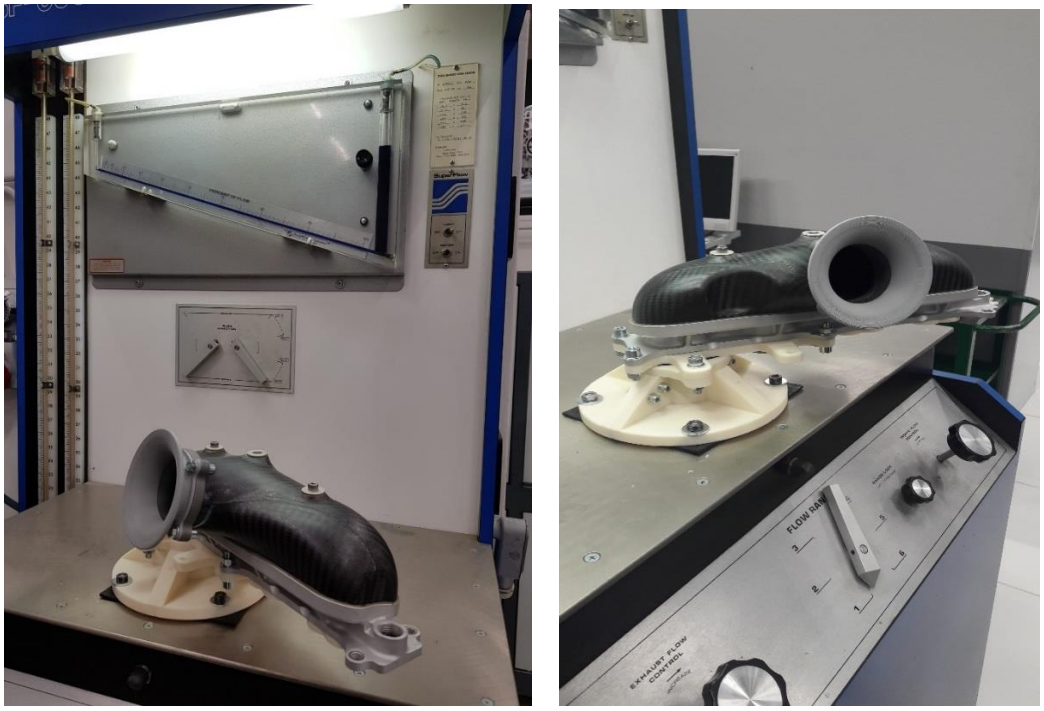


Figure 9. Intake Manifold Installation

The environmental conditions during the execution of the test are shown below:

Table 3. Environmental Test Conditions

Temperature	°C	21.5
Ambient Pressure	Pa	102100
Density	kg/m^3	1.187

2.3 Execution of the test and Results

The table below shows the data of the experimental test:

Table 4. Data of the Experimental Test

Pressure kPa	Range	Percentage of volumetric flow %	Ideal Flow l/s	Volumetric Flow Rate l/s
1.25	range3	85.3	72.15	61.54
2.45	range4	59	141.68	83.59
3.73	range4	71.5	141.12	100.90
4.98	range4	84.5	140.42	118.65
6.22	range5	62.5	208.16	130.10

In the first column it has the pressure level imposed during the test while in the second column it has the corresponding range. The third column represents the measurement of percentage of volumetric flow with respect to the ideal flow, that is shown in the fourth column while the remaining one is the real volumetric flow that results from the experimental test: it is obtained by the product of the ideal flow by the percentage of volumetric flow.

By converting the pressure from inches per water column to Pa and the mass flow from l/s to kg/s, it gets the following data:

Table 5. Input and Output for the simulation.

Outlet Pressure kPa	Mass Flow Rate kg/s
100.85	0.0693
99.61	0.0976
98.36	0.1191
97.12	0.1374
95.87	0.1530

The left column represents the pressure at the Outlet of the intake manifold that has been imposed by the flow bench while the right column represents the mass flow. With reference to the atmospheric pressure measured during the execution of the test, it obtains a mass flow – pressure drop characteristic:

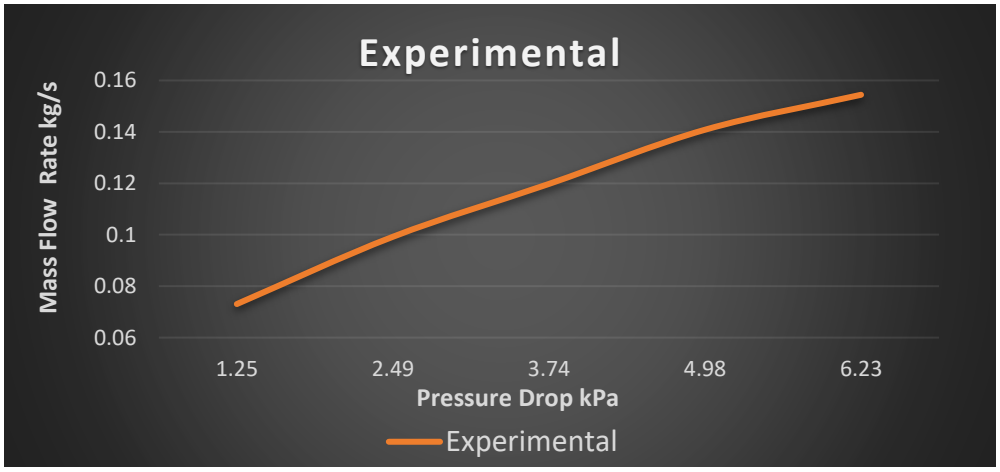


Diagram 1. Mass Flow Rate - Pressure Drop Experimental Characteristic.

This is the characteristic that it will try to match by means of the simulations, in order to validate the steady-state model.

2.4 Geometry Preparation

Starting from the original geometry, the intake manifold has been edited to produce a geometry compliant with a CFD simulation. Only the internal volume of the intake has been taken, while the runner length has been increased accordingly to the height of the joint that connected the intake manifold to the flow bench. To remove the pressure losses at the intake during the test, a mouthpiece has been installed on the intake manifold, therefore the same has been done during the editing of the geometry. Finally, it gets the following geometry:

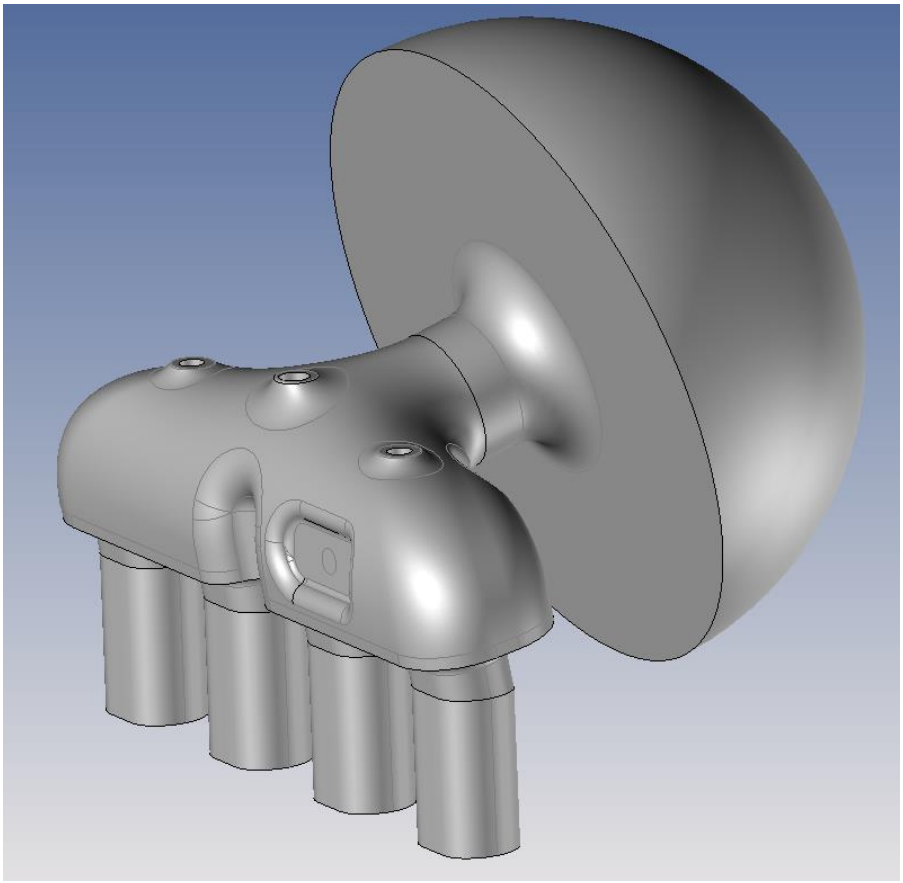


Figure 10. ATM-AR-F3R intake manifold edited.

The semi-sphere that is present at the inlet represents the atmosphere, therefore it has a volume significantly larger than the intake manifold volume.

2.5 Meshing Process

In OpenFOAM, there are 3 different mesh utilities:

- blockMesh: it is a mesh utility that permits to create simple geometry grids like cube, spere, rectangles and also profiles described by mathematical functions as polynomial of different order. It main aims is to create the background mesh from which the snappyHexMesh utilities will create the mesh of the interested geometry.

- SnappyHexMesh: snappyHexMesh is the main meshing tool of OpenFOAM. The snappyHexMesh mesh generation process is basically divided in three steps: the first is the castellated mesh, in which it uses the background mesh to fill the volume of the geometry we want to mesh, but since it uses only the cells of the background volume, the surfaces of the geometry are not well described. Then there is the snappy process, in which it cuts the cell that extend outside the surface of the geometry: this process is usually consuming in terms of computational time, and moreover the surface is of the geometry is not well described. The last step of the snappyHexMesh process is the layer addition, where the cells on the boundary are extruded towards the internal part of the geometry in order to produce the region for the layer addition. It is important to remark that the better the mesh on the surface will be, the better will be this extrusion process and so the better will be the boundary layer coverage. By default, snappyHexMesh does not exploit all the processors of the machine it is using to generate the mesh, but it gives the possibility of decomposing the background mesh produced by blockMesh in different parts and then to associate to each processor a reduced part of the snappy process. This is a procedure that can be adopted only when computational time for the mesh is no more acceptable because the mesh reconstruction process after the multi-parallel meshing process is difficult and usually produces a lower compliant mesh.

- CfMesh: this utility is present only in theOpenFOAM version v2006 (the version that has been used in this thesis project) and it is indicated for generating the volume and the surface mesh. Moreover, it uses all the available processors of the machine it is exploiting. As a result, the computational time for the meshing process is much lower with respect to the standard adoption of snappyHexMesh, unless it decomposes the background mesh with the

well-known risks. Additionally, CfMesh presents a sub-dictionary part in which it is possible to set the prism layer meshing controls. The prism layer addition of CfMesh does not permit to set mesh compliancy criteria. As a result, it will always have a 100% boundary layer coverage, but the mesh checks will underline possible bad cells that could impair the stability of the numerical solution.

In order to generate a mesh compliant with a stable numerical computation, it proceeds with the following step:

1. Volume and surface mesh generation performed by CfMesh: the surface approximation is better than the one obtained by snappyHexMesh, and the cell faces are more directed in a direction perpendicular to the flow direction. Moreover, the computational time in generating the mesh is reduce of 80% with respect to the adoption of the blockMesh and then the castellating and snappy processes of snappyHexMesh;

2. Prism Layer addition by snappyHexMesh: the castellated mesh and the snappy utilities of snappyHexMesh are not necessary anymore since the volume mesh has been already produced by CfMesh. The snappyHexMesh dictionary permits to perform the snappyHexMesh utility only for the prism layer addition, neglecting the two previous processes. This procedure permits to achieve boundary layer coverage close to 100%. The main reason of an almost perfect coverage even maintaining very constrained mesh acceptance limit is related to the extrusion process by which snappyHexMesh produces the layers. Due to the better surface approximation achieved by CfMesh, the cells on the intake manifold surface are perpendicular to the extrusion direction. This produces a negligible number of bad cells on which the layer are not added, therefore the boundary layer coverage is never below 99%.

Below it shows the mesh parameters:

Table 6. Mesh Parameters.

	K-Omega SST	Realizable K-Epsilon
Plenum Mesh Size	1.2mm	1.2mm
Atmosphere Volume Mesh size	3mm	3mm
Number of prism layer	6	2
Prism layer Expansion ratio	1.2	1.3
Total Number of cells	1.9 million	1.29 million

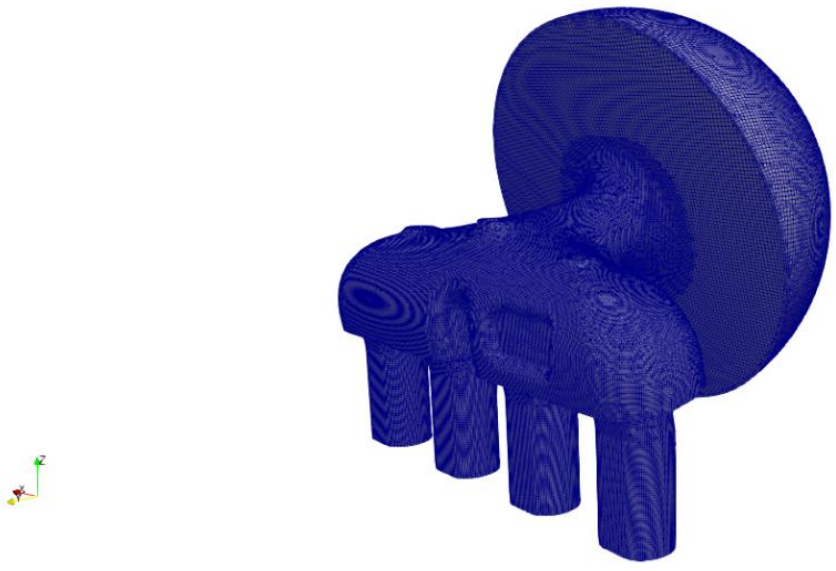


Figure 11. ATM-AR-F3R intake manifold mesh.

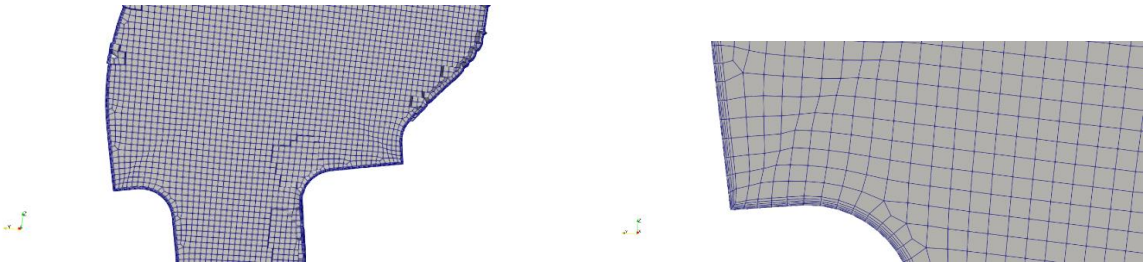


Figure 12. Detail of the mesh (left) and of the prism Layer for the K-Omega SST (right)

The mesh parameters are the same both for the K-Omega SST model and for the realizable K-Epsilon model except for the prism layers. The K-Omega SST model needs a higher number of layer because it directly solves the viscous sub-layer. In CFD it is of paramount important the y+, that represents a non-dimensional wall distance for a wall-bounded flow:

$$y^+ = \frac{u_* y}{\nu} \tag{2.5.1}$$

Where u_* is the friction velocity at the nearest wall, y is the distance to the nearest wall and ν is the local kinematic viscosity of the fluid. The K-Omega SST model application for this kind of simulation is of the low-Re type since the velocities inside the intake manifold rarely overcome a Mach number higher than 0.4. Consequently, a low-Re simulation that uses fully resolved boundary layers requires a refined boundary layer mesh with the first cell at the walls at a y+ less than one. Moreover, the expansion ratio with respect to the layer nearest to the wall has to be kept below 1.25. This is the reason why the k-Omega SST model mesh has a higher number of prism layers and a lower expansion ratio.

2.6 Thermophysical Properties

The thermophysical models are concerned with energy, heat, and physical properties. For this simulation, it chooses a thermophysical model for fixed composition, based on compressibility $\gamma = (RT)^{-1}$, where R is Gas Constant and T is the temperature.

It chose a pure mixture, which represents a mixture with fixed composition, and it sets the following coefficient:

```
mixture
{
  specie
  {
        nMoles          1;
      molWeight   28.86;
  }
}
```

Then it has to choose the transport model, that deals with the evaluation of the dynamic viscosity μ: it chose the const transport model, that assumes a constant dynamic viscosity μ and the Prandtl number $Pr = c_p\mu/k$, which is specified by the keywords mu and Pr:

```
transport
  {
      mu      1.8e-05;
        Pr      0.7;
  }
```

Afterwards, it had to specify the thermodynamic models, that concerns with the evaluation of the specific heat c_p and the heat of fusion H_f. It chose the hConst model, that assumes constant both c_p and H_f:

```
thermodynamics
```

```
{
    Cp          1004.5;
    Hf          2.544e+06;
}
```

Finally, it is necessary to choose the equation of state: it chose the perfect gas law $\rho = \frac{1}{p}RT$.

2.7 Turbulence Model

The necessity of a turbulence model comes from the application of the Reynolds Averaged Navier-Stokes equations (RANS). For example, considering the Equation 1, it is written for the instantaneous velocity.

In the RANS approach the Navier-Stokes equations are no more written as a function of instantaneous quantities, but they are averaged with respect an interval of time, that is short enough with respect to the phenomenon it is investigating but long enough with respect to the turbulence disturbances. For many practical applications the knowledge of mean quantities is accurate enough. The RANS approach has the advantage of significantly reducing the computational time since the turbulent scales of the mean flow are much bigger than the scales of the turbulence fluctuation. In fact, the instantaneous turbulent flow variables can be written as the sum of a mean quantity and its fluctuation along the mean:

$$U_i = \langle u_i \rangle + u_i \tag{2.7.1}$$

$$\rho = \langle \rho \rangle + \rho' \tag{2.7.2}$$

Where:

- $\langle u_i \rangle$ = mean velocity.
- $\langle \rho \rangle$ = mean density.
- u_i = velocity fluctuation.
- ρ' = density fluctuation.

The Equation 1.1.1 will result:

$$\frac{\partial\langle\rho\rangle}{\partial t}+\frac{\partial}{\partial x_i}[\langle\rho\rangle\langle u_i\rangle]+\frac{\partial}{\partial x_i}(\langle\rho' u_i\rangle)=0 \tag{2.7.3}$$

The last term depends on the turbulent flow field, and even if it considers an incompressible flow, so ρ' disappeares, the velocity fluctuation remains in the momentum conservation equation. Since it wants to solve the equations with respect to the mean velocity, the velocity fluctuation introduces another variable, therefore it is necessary another equation.

The turbulence model provides the closure equation. There are different turbulence models, but the most used ones are the eddy viscosity models. In the eddy viscosity models it associates the velocity and density fluctuations to the turbulent viscosity of the mean flow:

$$-\rho u_i u_j = -\frac{2}{3}\rho k\delta_{ij}+2\mu_t(S_{ij}-\frac{1}{3}S_{kk}\partial_{ij}) \tag{2.7.4}$$

Therefore, now it has to introduce two equations: one for the turbulent viscosity μ_t, and one for the turbulent kinetic energy k.

For the steady-state validation it decided to test two different turbulence models:

- Realizable k-Epsilon model;

- K-Omega SST model.

In the realizable k-Epsilon model, it uses the following equation:

$$\mu_t = C_\mu \frac{k^2}{\varepsilon} \tag{2.7.5}$$

Where ε is the dissipation rate and represents the rate at which the turbulent kinetic energy is converted into thermal internal energy while k represents the turbulent kinetic energy of the fluid and C_μ is a constant that needs to be calibrated. Since it adds two unknowns to model the turbulent viscosity, i.e., ε and k, two equations are necessary to close the model:

$$\frac{\partial(\rho k)}{\partial t} + \frac{\partial}{\partial x_j}\left(\rho\langle U_i\rangle k - \frac{\rho\nu_t}{\sigma_k}\frac{\partial k}{\partial x_j}\right) = P_k - \rho\varepsilon \tag{2.7.6}$$

$$\frac{\partial(\rho\varepsilon)}{\partial t} + \frac{\partial}{\partial x_j}\left(\rho\langle U_i\rangle \varepsilon - \frac{\rho\nu_t}{\sigma_\varepsilon}\frac{\partial \varepsilon}{\partial x_j}\right) = C_{\varepsilon l}\frac{\varepsilon}{k}P_k - C_{\varepsilon 2}\rho\frac{\varepsilon^2}{k} \tag{2.7.7}$$

For what concerns the K-Omega SST model, the equation 2.7.5 is substituted by the following equation:

$$\nu_t = \frac{a_1}{\max(a_1\omega, SF_2)} \tag{2.7.8}$$

Both the models are available in OpenFOAM, and they are used in combinations with wall functions boundary conditions: the realizable k-Epsilon model completely exploits the wall functions while the K-Omega SST model gives the possibility to directly solve the boundary layer when the first cell near the wall is close to y+=1.

2.8 Boundary conditions

For the steady-state model it adopts the RhoSimpleFoam solver, that requires the following boundary conditions:

- T = temperature °K;
- V = velocity m/s;
- P = pressure Pa;
- k = turbulent kinetic energy m^2/s^2.
- nut = turbulent viscosity m^2/s.
- omega = turbulent specific dissipation rate s^{-1}.
- alphat = turbulent thermal diffusivity m^2/s.

The table below shows the boundary conditions used in the simulation:

Table 7. Boundary Conditions.

	Inlet	Outlet	Wall
T	Total Temperature=294,65 °K	InletOutlet	zeroGradient
V	zeroGradient	PressureInletOutletVelocity	uniformFixedValue (0 0 0)
P	Total Pressure= 10200 Pa	uniformFixedValue	zeroGradient
k	turbulentIntensityKineticEnergyInlet	inletOutlet	kqRWallFunction
nut	calculated	calculated	nutkWallFunction
omega	turbulentMixingLengthFrequencyInlet	inletOutlet	omegaWallFunction
alphat	calculated	calculated	compressible::alphatWallFunction

As far as the outlet pressure boundary condition is concerned, it uses the values of pressure of the table 6, that are the pressure levels imposed during the test at the flow bench. For the last three values, it performed a false transient to not have an immediate pressure drop that could create numerical instability.

2.9 Numerical schemes

Inside the system folder there is the fvSchemes text file, in which it is necessary to specify the numerical schemes for terms, such as derivatives in equations, that are calculated during a simulation. In particular, the numerical schemes that must be specified are divided in the categories below:

- time schemes: they are specified in the ddtSchemes sub-dictionary. Since it is dealing with a steady state simulation, it used the *steadyState* time schemes, that sets the time derivative to zero.

- Gradient schemes: it is a sub-dictionary that contains gradient terms. In this simulation it used the following gradient schemes:

  ```
  default         Gauss linear;
  grad(U)         cellLimited Gauss linear 1;
  grad(k)         cellLimited Gauss linear 1;
  grad(omega)     cellLimited Gauss linear 1;
  ```

  It decided to use the *Gauss linear* scheme as the default schemes for all the gradients. The Gauss entry specifies the standard finite volume discretization of Gaussian integration, that requires the interpolation of values from cell centers to face centers. The interpolation scheme is given by the linear entry, that means linear interpolation. As far as the gradient of velocity, turbulent kinetic energy and the specific dissipation rate is concerned, it preferred to use a more limited scheme, the *cellLimited Gauss linear 1*: it is a more limited scheme because when the cell values are extrapolated to faces using the calculated gradient, the face values do not fall outside the bounds of values in surrounding cells, and so 1 represents boundness while a zero would mean no boundness.

- Divergence schemes: it contains divergence terms, including both advection terms, so the terms of the kind $\nabla(U \cdot x)$, where x is a whichever scalar variable and the velocity U provides the advective flux, and other terms, that are usually diffusive in nature. The keyword identifier for the advective terms are for example div(phi,U) for velocity or div(phi,k) for the turbulent kinetic energy: phi denotes the volumetric flux of velocity on the cell faces for

constant-density flows while it represents the mass flux for compressible flow. In this simulation it decided to use the *limitedLinear* scheme both for advective and diffusion scheme. The *limitedLinear* scheme is a linear scheme, so a second order scheme, that limits towards upwind (first-order bounded) in regions of rapidly changing gradients. It needs a coefficient from 0 to 1 to specify the level of boundness: 1 is the strongest limiting while the scheme tend to linear when the coefficient is close to zero. In this simulation, it sets the coefficient to 0.5.

- Surface normal gradient schemes: it represents the numerical scheme for the gradients evaluated at a cell face. The calculation of a gradient at a face is to subtract the value at the cell center on one side of the face from the value in the center on the other side and divide by the distance. It sets the surface normal gradient schemes to *limited corrected 0.5*, that is a second order accurate scheme but with an amount of correction with respect to the pure orthogonal scheme. This need was related to the fact that the mesh non-orthogonality is about 59, and so the angle between the cell-cell center and face normal vector is not zero: this would produce numerical instability, and so in order to have a convergent solution, a correction was necessary.

- Laplacian schemes: these are the numerical schemes for the Laplacian terms like, for example, the diffusion term $\nabla \cdot (\nu \nabla U)$ of the momentum equations. In this simulation, it used the Gauss linear corrected 0.5, where 0.5 is a correction based on the level of non-orthogonality.

- Interpolation schemes: it contains terms that are interpolation of values typically from cell centers to face centers. They are typically used in the interpolation of velocity to face centers for the calculation of the volumetric flux in case of incompressible fluid, or the calculation of mass flux, so when it deals with compressible fluids. It sets the interpolation schemes to *linear.*

2.10 Solver

Since it is dealing with a compressible, subsonic turbulent flow, it decided to use the solver rhoSimpleFoam, that is a steady-state solver for turbulent flow of compressible fluids. This solver is based on the SIMPLE algorithm. To understand how the SIMPLE algorithm works, it considers the mass conservation equation and the conservation of momentum:

$$\overrightarrow{\nabla u} = 0 \tag{2.10.1}$$

$$\frac{\overrightarrow{\partial v}}{\partial t} + \left(\vec{v}\vec{\nabla}\right)\vec{v} = -\frac{1}{\rho}\vec{v} + \nu\Delta\vec{v} \tag{2.10.2}$$

If it assumes to solve a 2-dimensional problem, it gets 3 unknow: p, u_x and u_y. Therefore, it is possible to bring out an equation for pressure by performing the divergence of the momentum equation, that in technical way is named "divergence free velocity condition:

$$\nabla\frac{\overrightarrow{\partial v}}{\partial t} = 0 \tag{2.10.3}$$

$$\nabla(\nu\Delta\,\vec{v}) = 0 \tag{2.10.4}$$

Therefore, it gets:

$$\left(\vec{v}\vec{\nabla}\right)\vec{v} = -\frac{1}{\rho}\overrightarrow{\nabla \mathrm{p}} \tag{2.10.5}$$

This is the Poisson equation of the pressure for a given velocity field. But this is not enough, because it is necessary a pressure correction method to be sure that the continuity equation will be satisfied.

Starting from the Navier-Stokes equation, in the first iteration, at the time zero, it has a guess of pressure and velocity p_n and v_n, given as boundary conditions: these are not the correct values of velocity and pressure, but they are only a starting guess. From the Navier-Stokes equation, it gets an intermediate velocity v^*:

$$\frac{\overrightarrow{v^* - v^n}}{\Delta t} = -\left(\overrightarrow{v^n} \cdot \vec{\nabla}\right)\overrightarrow{v^n} - \frac{1}{\rho}\vec{\nabla}p^n + v\nabla\overrightarrow{v^n} \tag{2.10.5}$$

By this value of the intermediate velocity v^*, it enters in the Poisson equation and it computes a new value for the pressure, that is the pressure correction p'. By this p', it corrects the initial pressure p_0, obtaining:

$$p^{n+1} = p^n + p' \tag{2.10.6}$$

Once it has the corrective pressure p', it can write the following equation, that represents the Navier-Stokes equation with the corrected pressure:

$$\frac{\overrightarrow{v^{n+1}} - \overrightarrow{v^n}}{\Delta t} = -\left(\overrightarrow{v^n}\nabla\right)\overrightarrow{v^n} - \frac{1}{\rho}\vec{\nabla}p^{n+1} + v\Delta\overrightarrow{v^n} \tag{2.10.7}$$

By subtracting the equation (2.10.7) to the equation (2.10.5), it gets the equation to compute the velocity correction v':

$$\overrightarrow{v'} = -\frac{\Delta t}{\rho}\vec{\nabla}p' \tag{2.10.8}$$

Once v' has been computed, it corrects the intermediate velocity v^*, obtaining:

$$\overrightarrow{v^{n+1}} = \overrightarrow{v^*} + \overrightarrow{v'} \tag{2.10.9}$$

It gets the velocity for the new time step, and then the iteration restarts. This is the basis algorithm used to perform the pressure-velocity coupling, that is the basis of the simple algorithm.

To summarize, the SIMPLE algorithm is made of the following steps:

- It guesses the pressure field p*;
- It solves the momentum equation to get u* and v*;
- It uses u* to compute p' in the pressure correction equation;
- It updates the pressure: p=p*+p';
- It computes the new velocity $u_p=u_p*+u'$;
- It goes to the next iteration, taking as new pressure p as the new guess of pressure (in the previous iteration it was p*).

2.11 Steady-state Simulation Results

Once all the set-up is done, it is possible to start with the simulations. In particular, it performed a set of 5 simulations for each turbulence model, where only the right runner is considered as outlet while the remaining ones are treated as walls. Each simulation is characterized by a constant inlet total pressure of 102000 Pa, i.e. the atmospheric pressure measured during the test, while the outlet pressure is taken, for each simulation, from the pressures of Table 6. The aim of each simulation is to match the mass flow measured at the flow bench.

Below it shows the velocity field for the first simulation, concerning an Outlet Pressure of 100854.59 Pa:

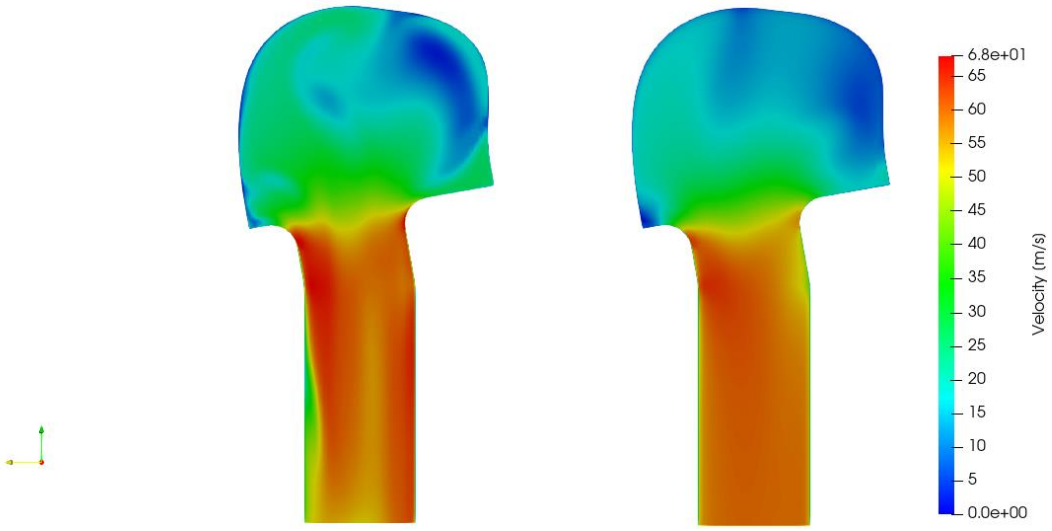


Figure 13. Velocity profile for K-Omega SST (left) and realizable K-E (right).

Regarding the mass flow, the diagram below shows the trend from the beginning of the simulations up to convergence:

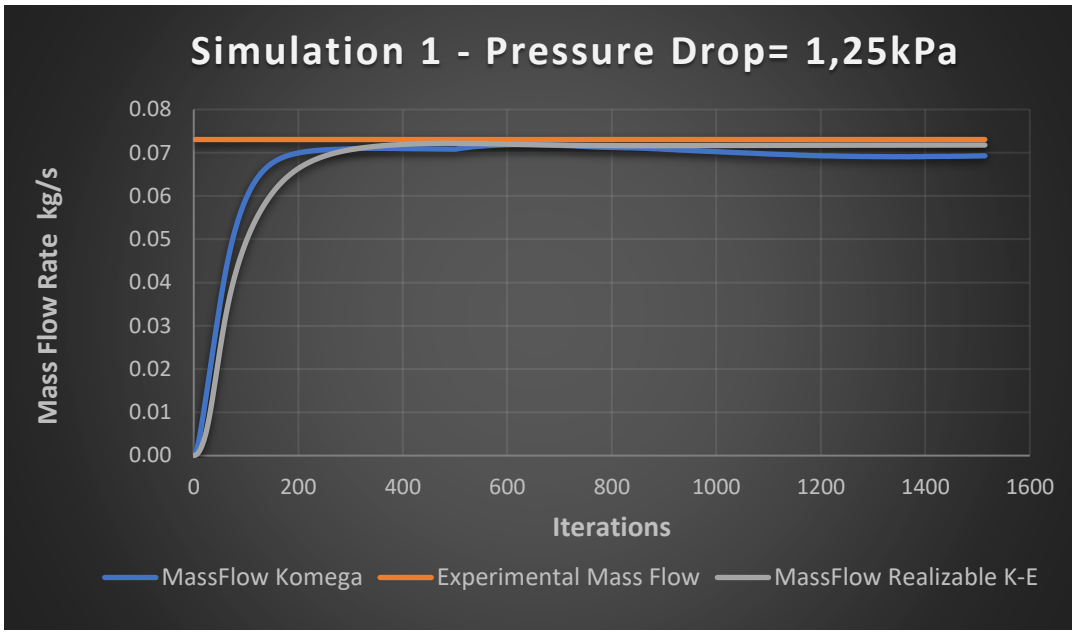


Diagram 2. Mass Flow Rate Convergence for 1.25 kPa pressure drop.

Looking at the diagram, the simulation stabilizes after 1500 iterations, and the mass flow remains stable. Considering the experimental mass flow, that is 0.069 kg/s, for both the turbulence models the measured flow rate and the computed flow rate are inside a percentage variation range with respect to the target of 5%. In particular, the K-Omega SST model produces an underestimation of the mass flow rate due to the development of an intense turbulent region slightly before the region of the plenum above the runner:

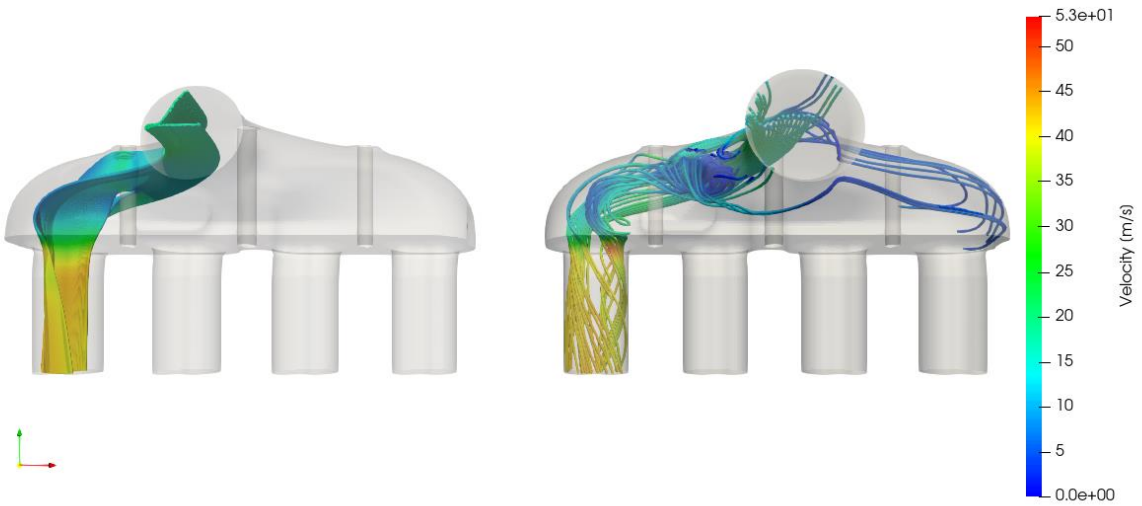


Figure 14. Streamlines k-Omega SST (right) and realizable K-Epsilon.

This turbulent region induces a slight decrease of the mass flow, reducing the uniformity of the velocity distribution at the outlet.

The second simulations concern a pressure drop equal to 2.49 kPa. The simulations converge after 1600 iterations, and below it shows the trend of the mass flow:

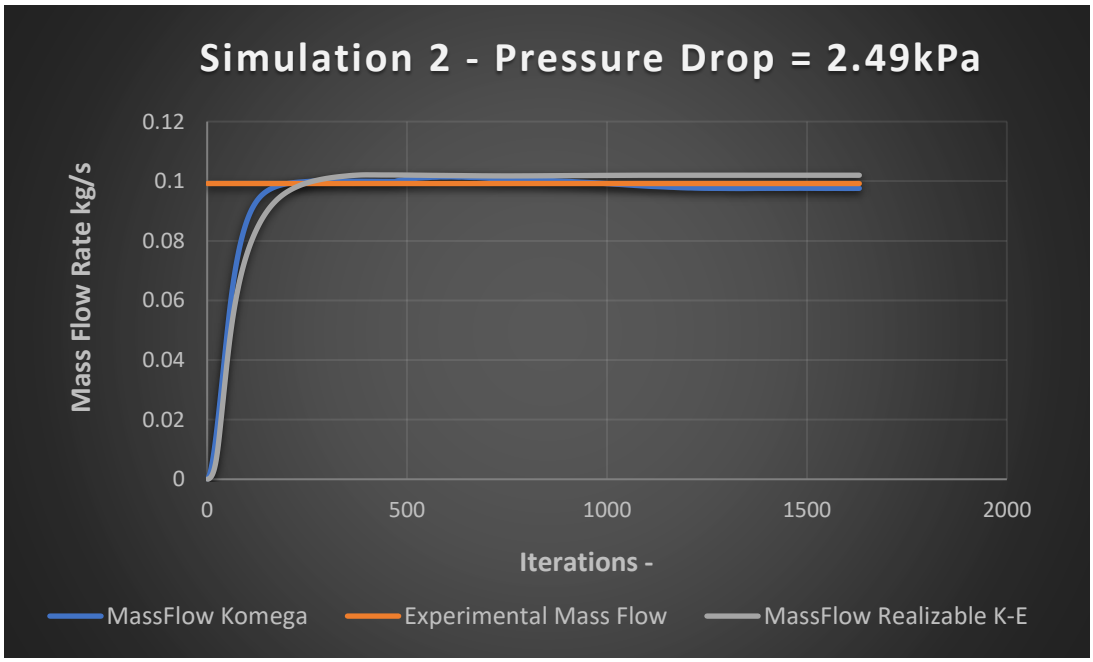


Diagram 3. Mass Flow Rate Convergence for 2.49 kPa pressure drop.

The realizable k-Epsilon turbulence model always gives an overestimation of the real mass flow, while the k-Omega model is more conservative. Anyway, for both the simulations, the percentage error with respect to the target is below the threshold limit of 5%.

The velocity field and the streamlines are shown below:

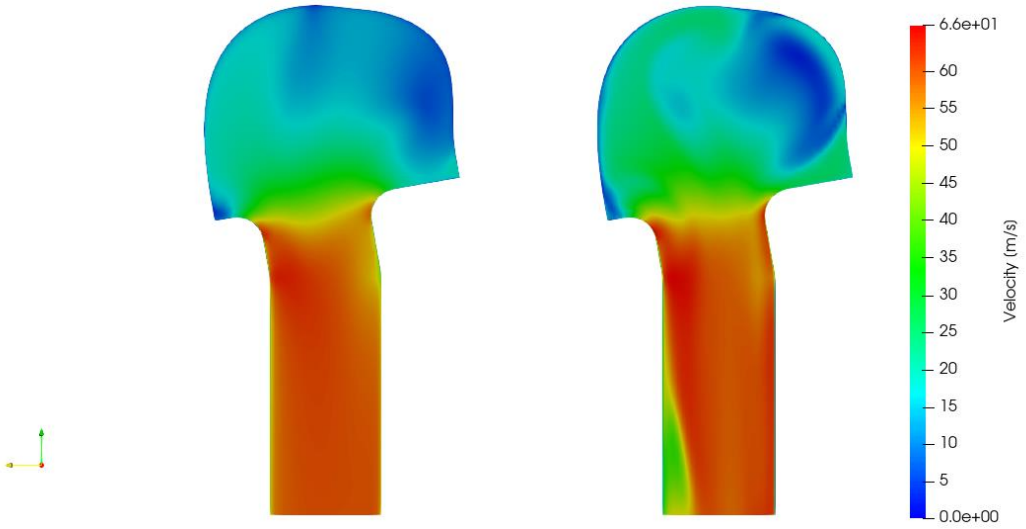


Figure 15-Velocity field K-Omega SST (right) and realizable K-E (left).

The velocity field distribution is similar to the previous simulation, but the k-Omega model provides a higher flow detachment from the internal runner surfaces with respect to the realizable k-epsilon.

Below it shows the streamlines:

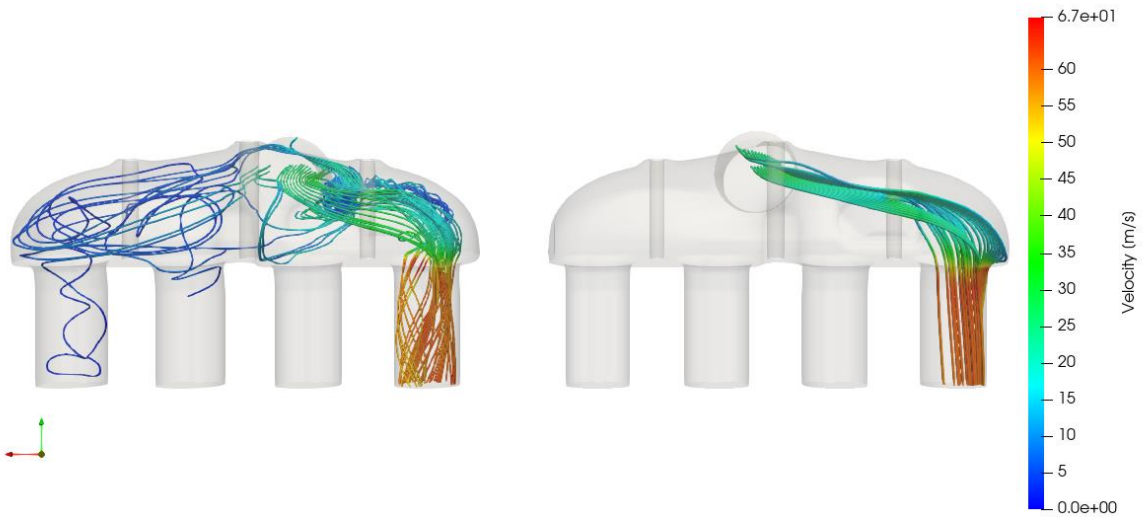


Figure 16. Streamlines K-Omega SST (left) and realizable K-E (right).

The k-Omega SST model provides a much higher turbulent region inside the fluid domain, and therefore the mass flow exiting from the runner outlet is lower with respect the realizable k-Epsilon.

The next simulations follow the trend of the previous ones, with the only change in the field variable values due to the increase of the pressure drop.

The diagrams below show the resulting mass flows with respect to the target one:

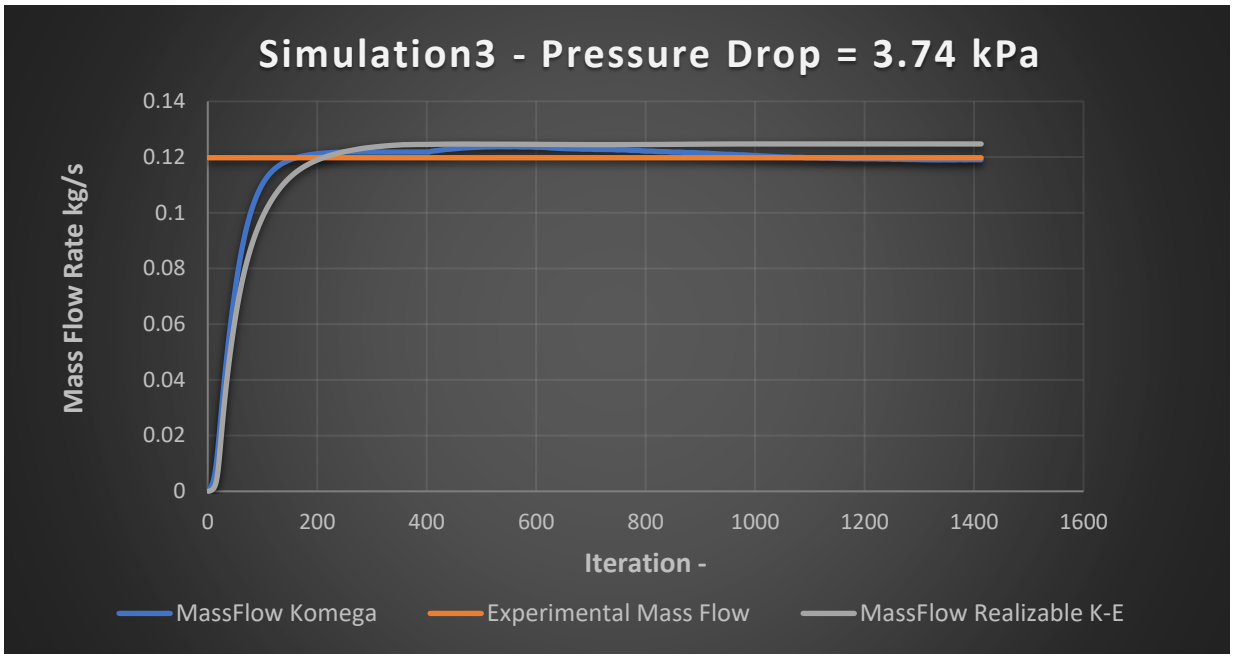


Diagram 4. Mass Flow Rate Convergence for 3.74 kPa pressure drop.

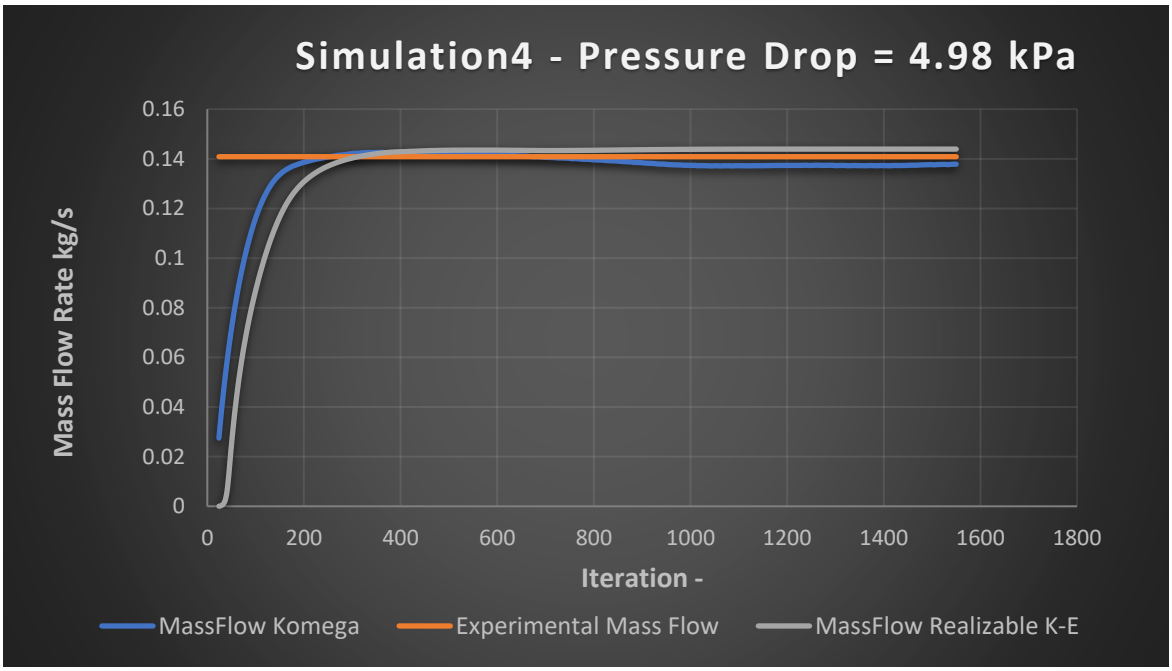


Diagram 5. Mass Flow Rate Convergence for 4.98 kPa pressure drop.

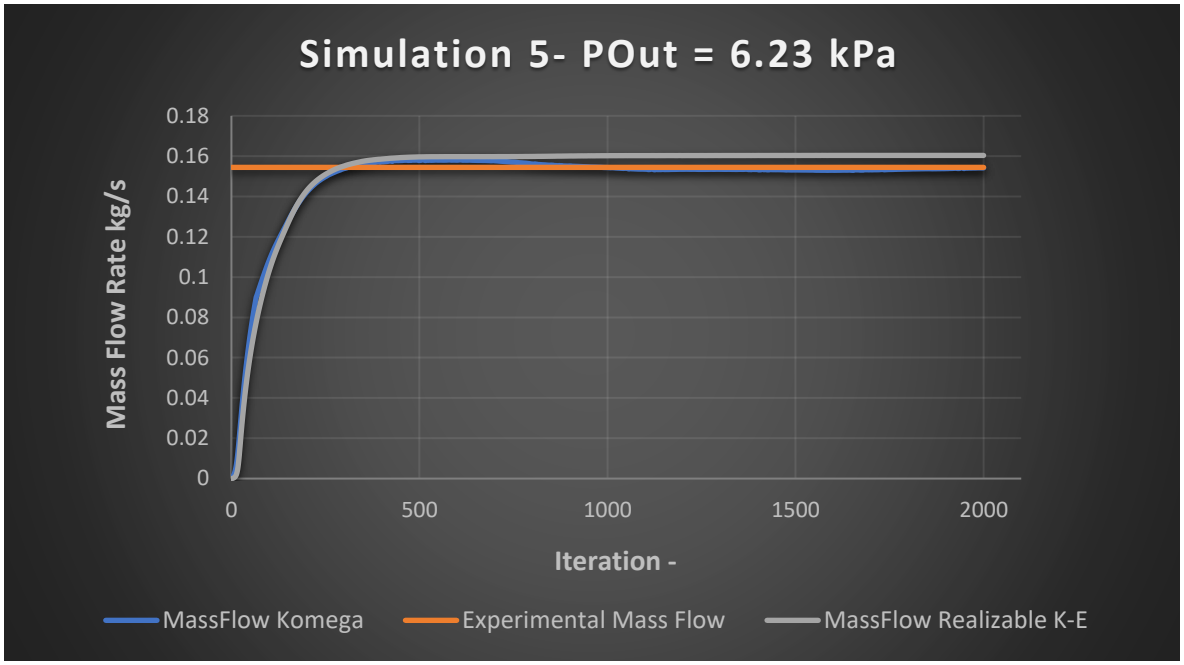


Diagram 6. Mass Flow Rate Convergence for 6.23 kPa pressure drop.

2.12 Conclusion on the steady-state validation

The table below summarizes the results of the steady state simulation, compared to the target values:

Table 8. Summary Comparison Experimental Data - Simulation Results

Pressure Drop [kPa]	**Measured Mass Flow [kg/s]**	**Mass Flow k-Omega SST kg/s]**	**Percentage Erorr k-ω [%]**	**Mass Flow realizable K- ε [kg/s]**	**Percentage Error realizable k- ε [%]**
1.25	0.0730	0.0693	**-5.15**	0.0718	**-1.75**
2.49	0.0992	0.0976	**-1.62**	0.1020	**2.80**
3.74	0.1198	0.1191	**-0.60**	0.1247	**4.14**
4.98	0.1408	0.1374	**-2.45**	0.1439	**2.18**
6.23	0.1544	0.1530	**-0.91**	0.1604	**3.84**

Both the turbulence models show a good numerical stability. The K-Omega model provides a higher turbulence of the flow, that reflects on a less uniform outlet velocity distribution. This can be visualized both qualitatively but also quantitatively:

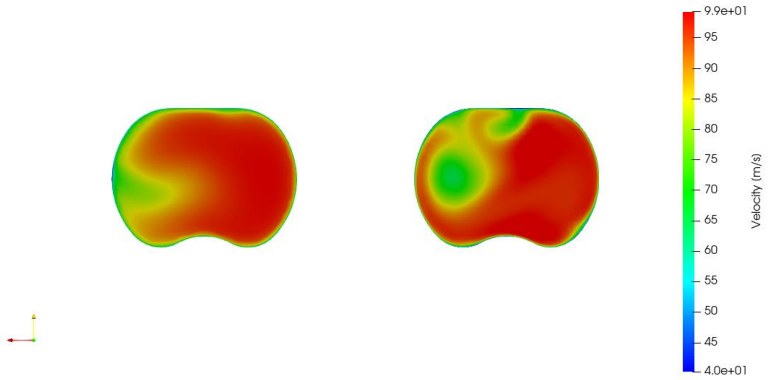


Figure 17. Velocity distribution on the outlet section - realizable K-Epsilon (left) and K-Omega SST (right).

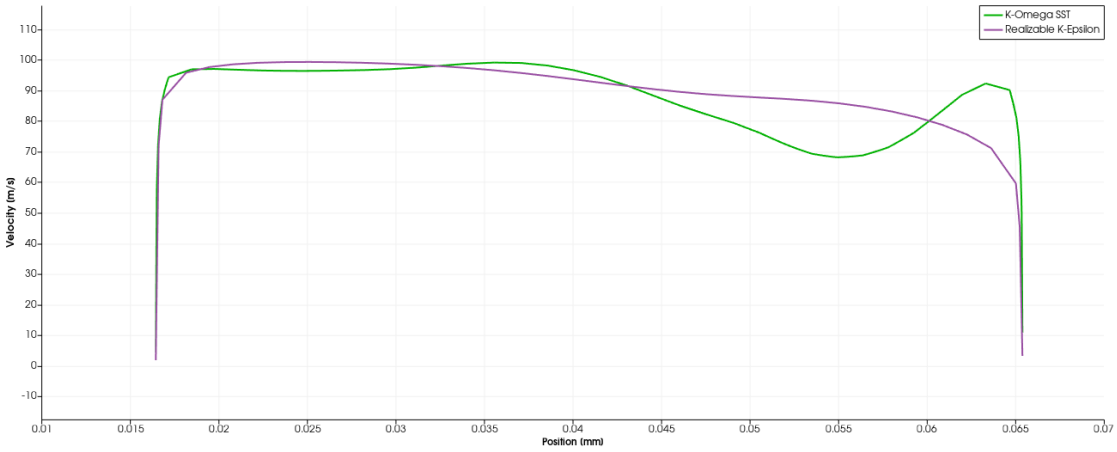


Figure 18: Velocity profile on a line passing for the midline of the outlet section of the runner.

The diagram above shows the velocity distribution along a line passing in the middle of the outlet section of the runner. The velocity profile is equal on the right side of the outlet section, but from the middle of the section and going on a turbulent vortex, present in the fluid domain of the K-Omega model, creates a drop of velocity with respect to realizable K-Epsilon, that has a much more uniform velocity distribution. This different velocity distribution confirms how the K-Omega model is a good approximation of a turbulent flow when there are high pressure gradients while the realizable K-Epsilon is more indicated when it deals with well-developed flows.

In conclusion, both the turbulence models can validate the steady state model, but the K-Omega model, since it directly resolves the boundary layer when the y+ is below 1, needs a thicker prism layer, increasing therefore the cell numbers and so the computational time. Moreover, the realizable K-Epsilon shows a higher numerical stability with respect to the K-Omega SST model. Anyway, the K-

Omega SST model provides a better prediction for flow detachment from the surfaces, and this is a condition that usually occurs in an intake manifold, where the flow is constrained from the wall. For this reason, it decided to adopt the K-Omega SST model.

With the results obtained by the steady-state simulation, it is possible to build the same characteristic as it did for the experimental result: this characteristic will be used to compare the performance of each runner for two intake manifold different geometries, to assess which configuration will be the best one.

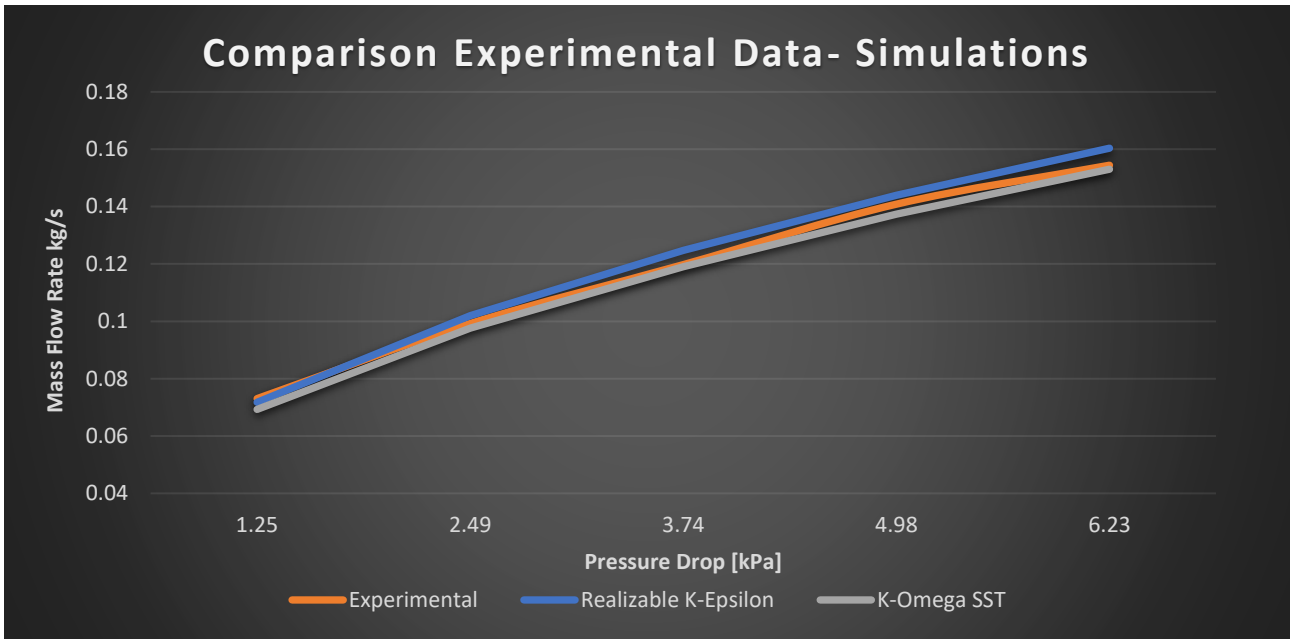


Diagram 7. Mass Flow Rate vs Pressure Drop characteristic comparison.

Once the steady-state model is validated, it can be considered accurate enough to simulate the air flow in different intake manifold geometry layouts.

3 INTAKE MANIFOLD LAYOUT DEFINITION

The intake manifold is made of two different fundamental parts: a plenum or air chamber in which the air is accumulated during the engine cycle, and a set of runners that connects the plenum to the intake ports, located on the cylinder head. The main purpose of an intake manifold is to transfer the air coming from the turbocharger group to the cylinder head. An intake manifold optimization improves the volumetric efficiency and the performances of the engine, because the higher is the mass flow going into the cylinders, the higher is the amount of fuel that can be injected and so the higher is the output power.

In the ATM V6 engine the fuel injection is performed behind the intake valve or directly inside the combustion chamber, therefore only air is present in the intake manifold. Usually racing engines are endowed by a throttle body plate per cylinder, but for the preliminaries engine developments, it decided to use only one throttle valve per bank at the intake manifold inlet.

At first, it is necessary to understand which is the best flow addressing in the plenum. Considering the engine layout, there are two possible solutions, that comes from the analysis of bulk of the engine accessories and of the pipe routings to have an efficient installation on the vehicle.

- Lateral air flow entrance.
- Central air flow entrance.

The runner length and the plenum volume has been defined in the 1D model of the ATM V6 engine, because in the 1D model the runner length can be optimized in terms of pressure waves tuning while the plenum volume can be analyzed taking in account the fast-transient response of the engine. As a result, they were fixed to following values:

Table 9. Runner and Plenum Dimensions.

Runnel Length	Plenum Volume
85 mm	2.35L

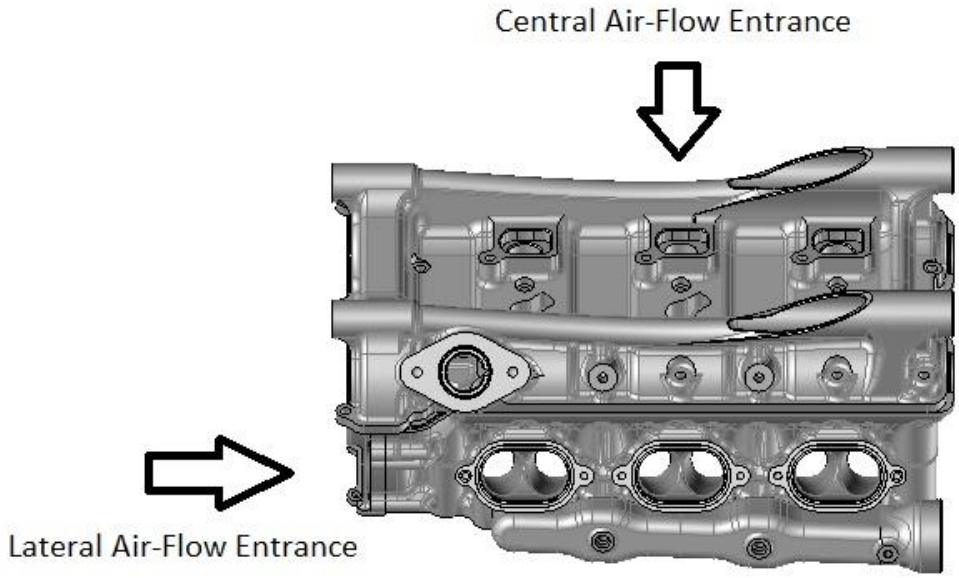


Figure 19. Possible configuration of air flow addressing.

Starting from these two possibilities, two different geometries have been modeled on the CAD software TopSolid. In the first geometry, with the central air flow entrance, the throttle valve is connected to the plenum by an important divergent part: the idea is to make the air flow to expand in such a way to recover the pressure losses that occurs when it passes through the throttle valve. The dimension of the fillet that connects the plenum to the runners is 10mm.

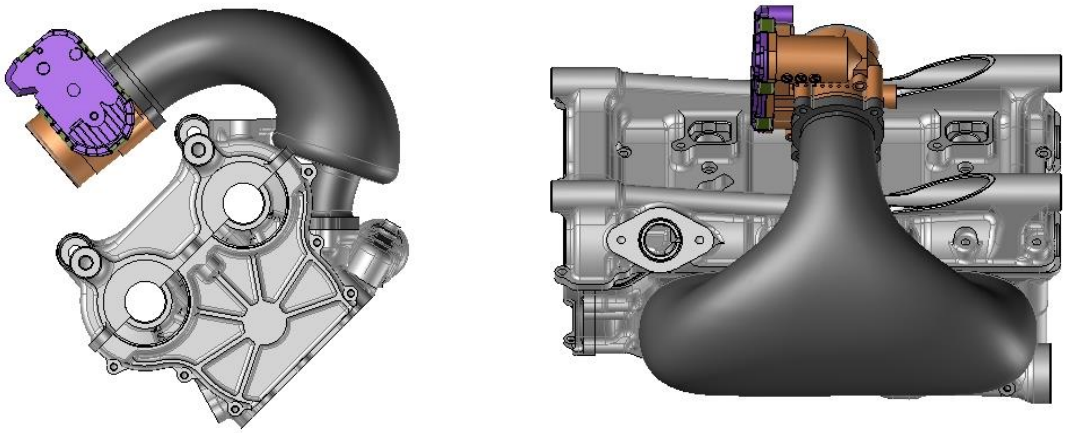


Figure 20. Central air flow entrance configuration.

The second geometry layout that has been produced has the same geometrical characteristics of the previous one, but the air enters laterally:

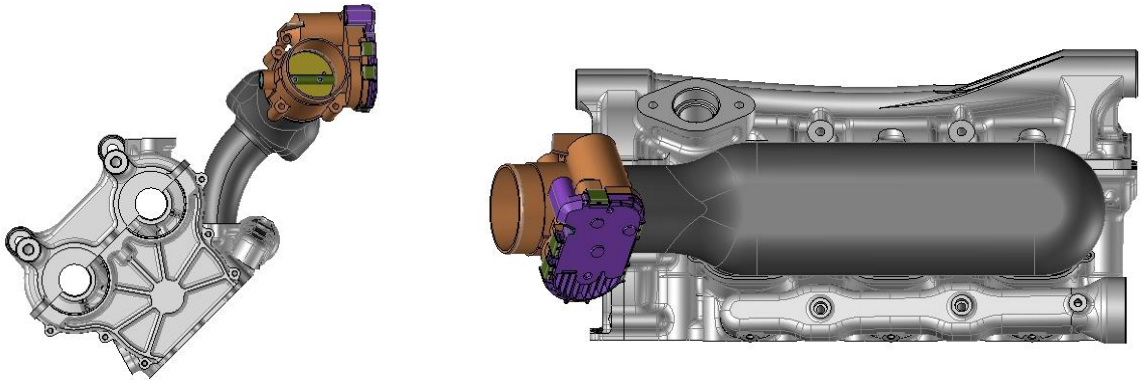


Figure 21. Lateral Air flow entrance configuration.

The table below summarizes the most important geometrical characteristic of the two intake manifolds:

Table 10. Intake manifold geometrical characteristics.

	Fillet Dimension mm	Plenum Volume L	Runner Length mm	Diameter of inlet cross section mm
LATERAL	10	2.35	85	60
CENTRAL	10	2.3	85	60

Looking at the table, the two geometry are similar in terms of geometrical characteristics. This is done because at this stage it wants to investigate which is the best configuration in terms of air flow entrance.

3.1 Simulation Set-up

Starting from the two geometries shown in the previous part, they have been edited on the opensource Software SALOME, to make the geometries ready for a steady-state analysis:

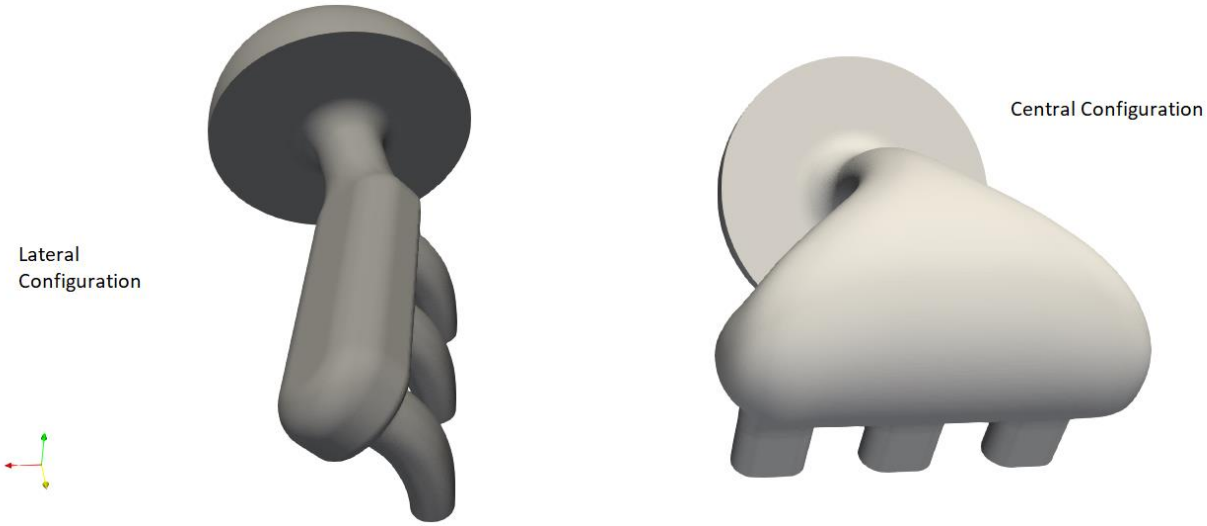


Figure 22. Central and Lateral Configurations

The runner length of the Central configuration has been increased up to 85 mm, in order to make the two geometries as similar as possible, except for the air flow entrance. The aim of the simulations is to analyze, by imposing the same boundary conditions used in the steady-state model validation, the mass flow exiting from each runner: for the same pressure drop between inlet and outlet, the higher is the mass flow, the lower are the pressure losses. The Central configuration will be analyzed only for 2 runners since it exploits the symmetry of the runner 1 and 3.

3.2 Simulation results

The histograms below show the results in terms of comparison for each runner between the two geometries:

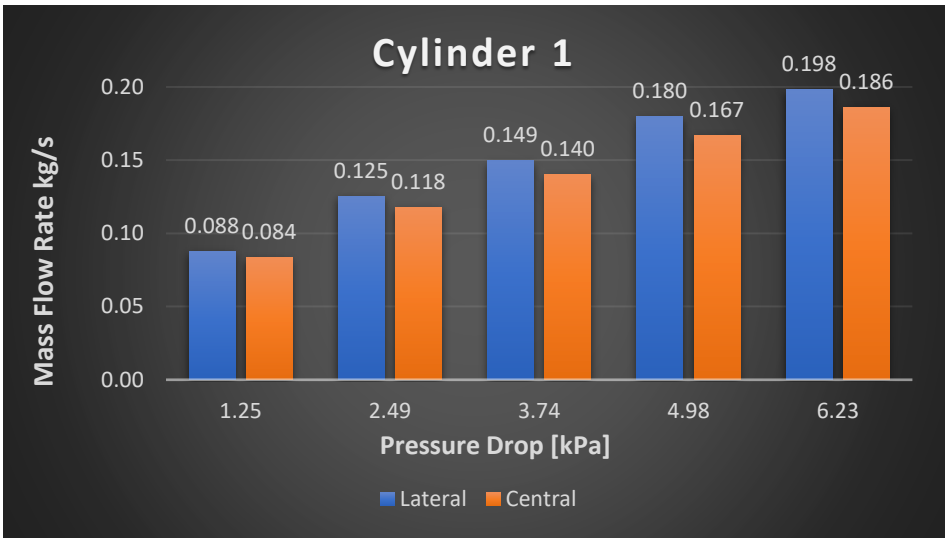


Diagram 8. Cylinder 1 Mass Flow Rate comparison for the lateral and central configuration.

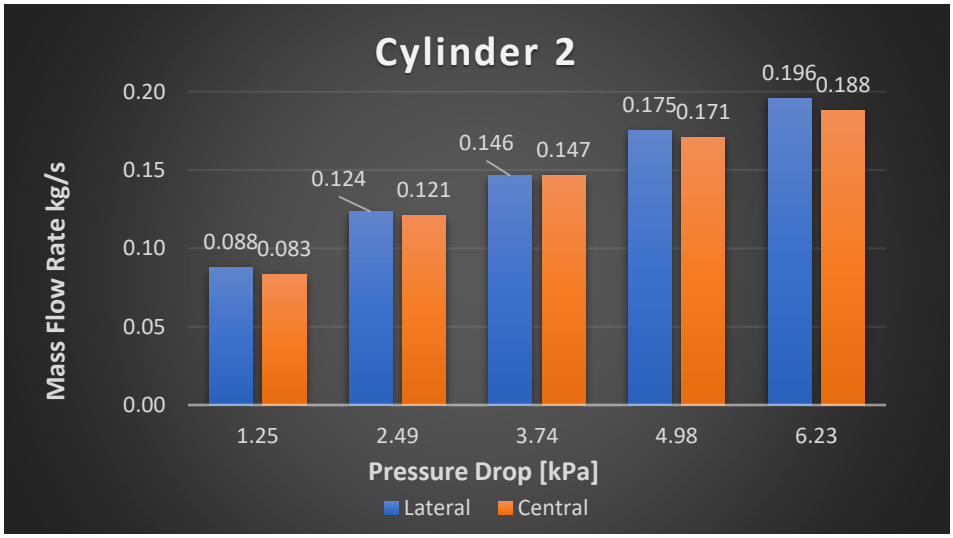


Diagram 9. Cylinder 2 Mass Flow Rate comparison for the lateral and central configuration.

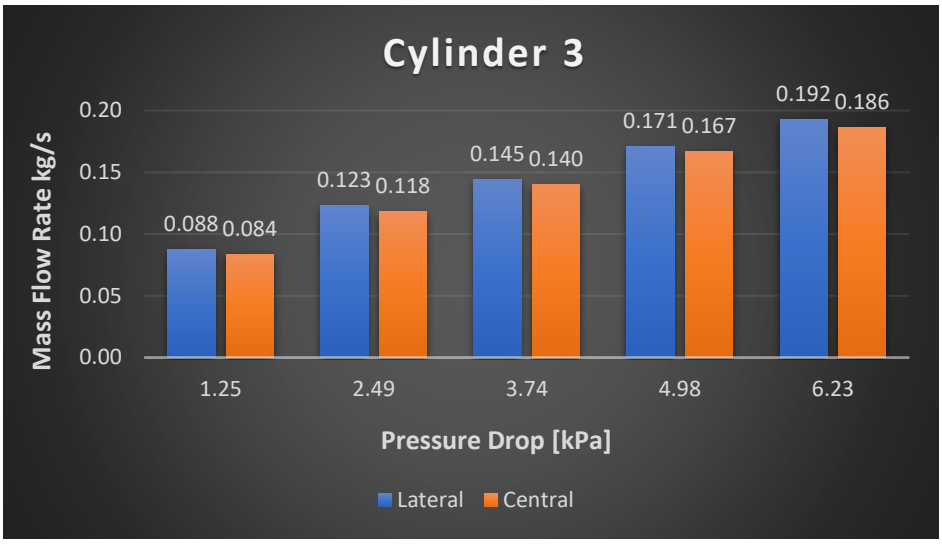


Diagram 10. Cylinder 3 Mass Flow Rate comparison for the lateral and central configuration.

The Lateral configuration shows a higher mass flow for each pressure drop tested, and the higher difference occurs when the pressure drop increases up to 6.23 kPa.

To understand the reason of this difference, it analyzes the flow fields for the worst-case condition of 6.23 kPa. The streamlines immediately underline a possible reason of this variation of mass flow between the two solutions. Considering the runner 1:

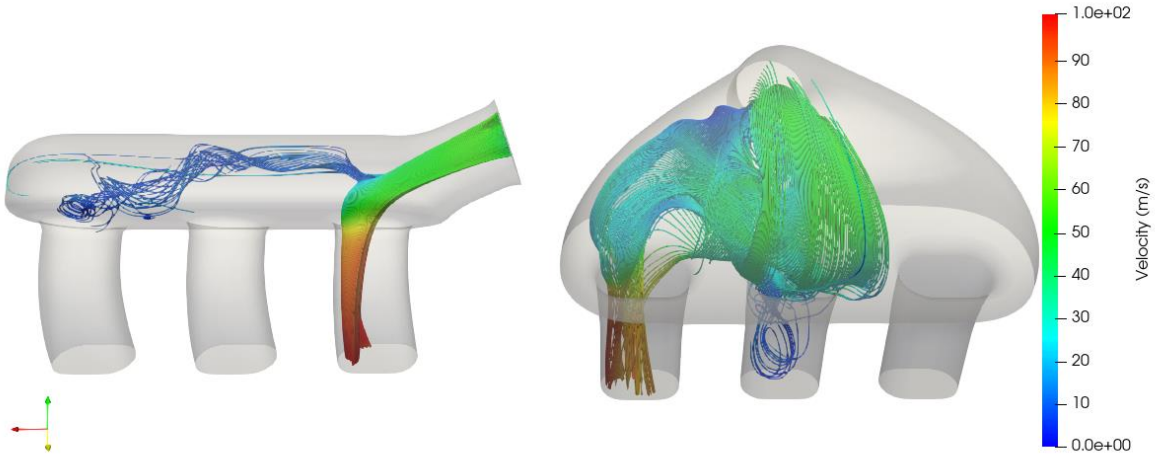


Figure 23. Streamlines.

The Lateral configuration has a well-directed and steady air flow that enter inside the runner 1 while the central configuration shows an important turbulent recirculating region in the middle of the plenum and in the divergent region towards the first runner. These considerations are confirmed considering the velocity field in the region above the runner:

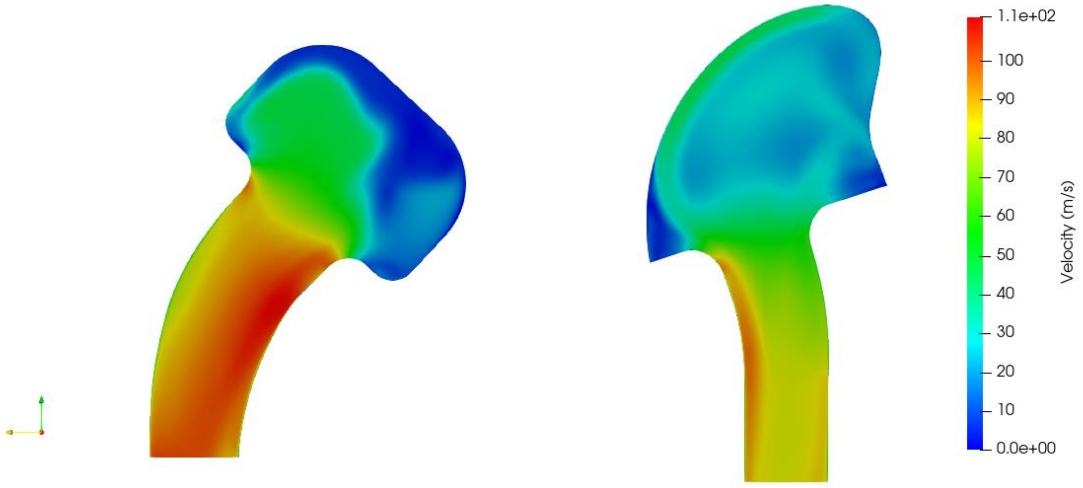


Figure 24. Velocity magnitude field.

The lateral geometry has a higher speed distribution in the region above the first runner while the central geometry, in the same position, has an important turbulent region: this is the main cause of the mass flow rate variation between the two different geometries.

It is also important to notice how the velocity distribution on the outlet is higher for the lateral configuration, except in the last portion:

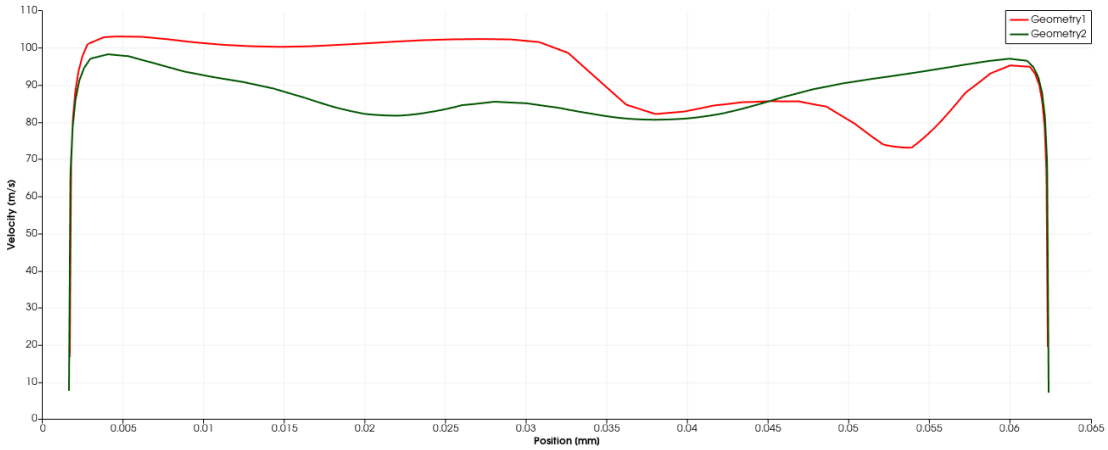


Figure 25. Velocity Magnitude along a midline passing for the outlet section.

The lateral configration shows that in more than half of the outlet section, the velocity is higher than the geometry 2, while in the second part they become equal, up to the final part in which there is a velocity drop. This velocity drop is related to the flow detachment shown below:

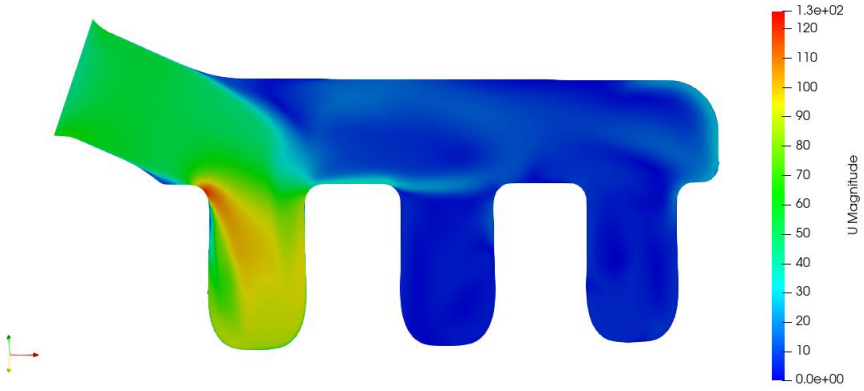


Figure 26. Flow detachment in the runner 1 – Lateral Configuration.

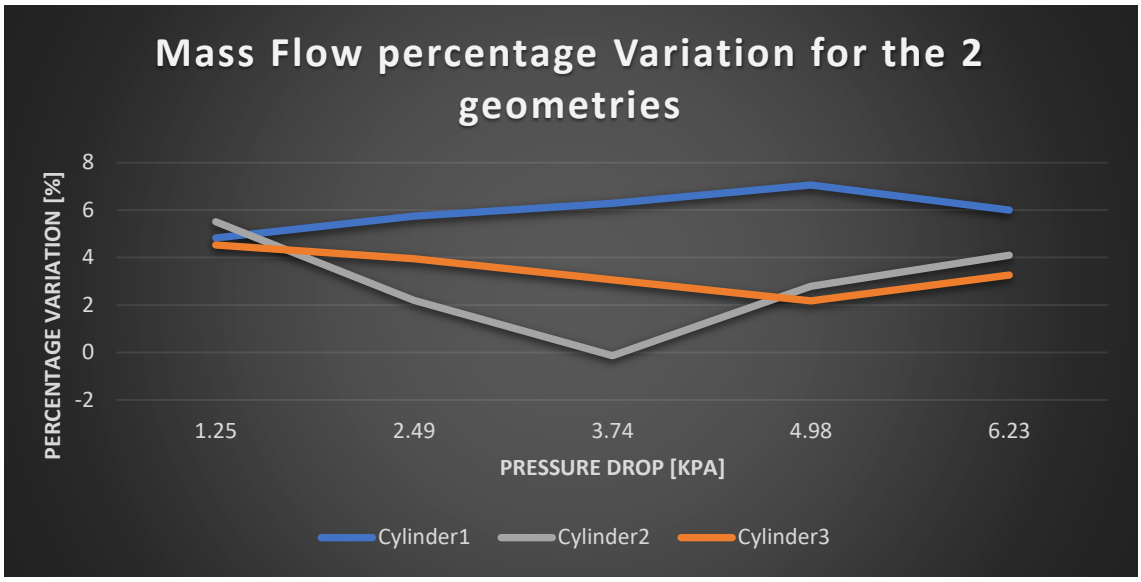


Diagram 11. Mass Flow Rate percentage Variation for each cylinder of the two configurations.

The diagram above shows the percentage of variation of mass flow of each runner of the lateral configuration with respect to the central configuration, and so it well underlines the advantage of a lateral air flow entrance if compared to the central one.

3.3 Conclusions on the intake manifold layout definition

From the analysis of the simulation results, it is clear how, for the same pressure drop, the lateral geometry has a higher mass with respect to the central geometry, and therefore it can be considered the best one. The flow addressing is along the axis of the plenum and controlling the divergent region that connects the entrance region to the plenum is much easier with respect to the central geometry because of the central entrance, since whichever is the geometry of the divergent region, it will always have a flow detachment, that means regions of high-pressure gradients and so the development of turbulent regions that will reduce the effective area of passage of the flow: this will reduce the mass flow.

The following characteristic will be used to compare different geometries modification in order to individuate the geometrical parameters that mostly affects the intake manifold performance:

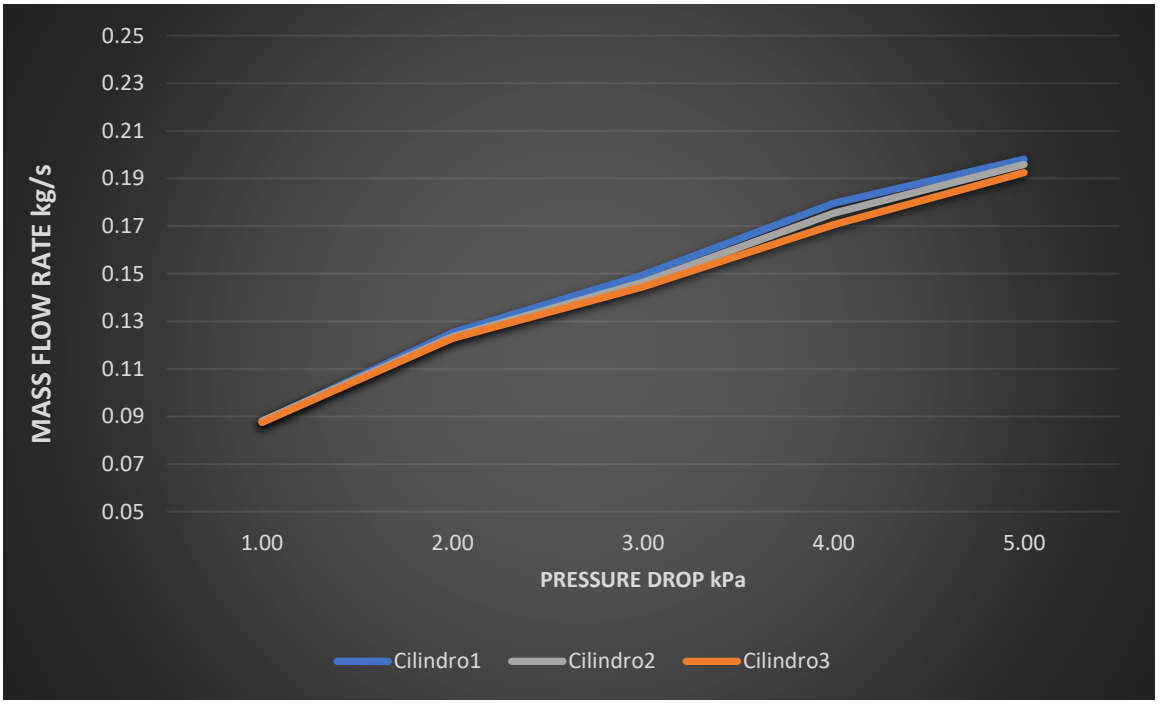


Diagram 12. Mass Flow Rate - Pressure Drop characteristic for the cylinders of the lateral configuration.

4 GEOMETRY OPTIMIZATION

Once defined the best air flow entrance, it wants to understand which are the most important geometrical characteristics of the intake manifold in terms of pressure drop reduction, increment of the mass flow and uniformity of velocity distribution at the outlet of each manifold.

It will focus on three different geometrical characteristics:

- Fillet of the mouthpiece: this geometrical characteristic has the aim of guiding the flow from the plenum to the beginning part of the runner with the lowest possible dissipation. A big fillet is not detrimental but if it is too thick, it could increase excessively the boundary layer, and so it would increase the friction forces, producing a loss of energy of the fluid. In the same way, a too large fillet can increase excessively the local velocity, with a subsequent deceleration and local detachment of the flow from the surface.

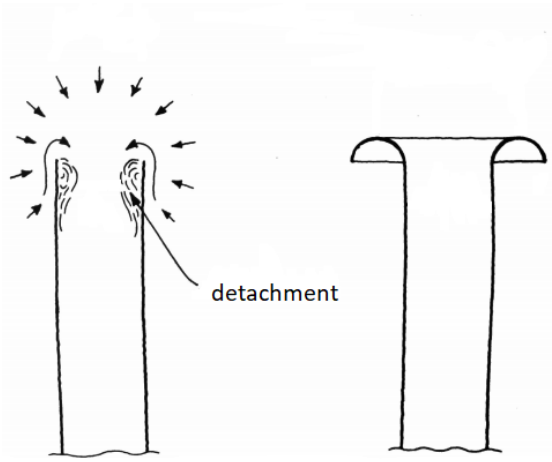


Figure 27. Fillet of the runner.

- Area of the inlet section: it corresponds to the diameter of the plate of the throttle valve. In this region the air flow velocity reaches high values, even if it is at full throttle condition: this means a transformation of pressure energy in velocity energy, that not always can be recovered. In the market throttle valve with different standardized plate diameter are present, and therefore this is not a parameter that can be arbitrarily varied: the values will be analyzed are found in the catalogue of the supplier it chose for the throttle valve.

The table below shows the values of the parameters that will be investigated by means of steady-state computational fluid dynamics simulations:

Table 11. Simulation Schedule for the Geometry Optimization.

	Simulation 1	Simulation 2	Simulation3
Mouthpiece Fillet Radius	10 mm	15 mm	20 mm
Inlet cross section diameter	60 mm	68 mm	82 mm

Each parameter will be analyzed for each runner of the intake manifold, and it will try to understand which is the most affecting geometrical characteristic on the intake manifold performance.

It will analyze the mass flow rate resulting from the variation of the following geometrical characteristics and also the velocity uniformity distribution at the outlet of each runner. The velocity uniformity index is indicated by γ and it is referred to a surface, i.e., the outlet section of the intake manifold. It permits to evaluate the uniformity of a scalar quantity, and so once computed the velocity magnitude, it can apply the following expression of γ:

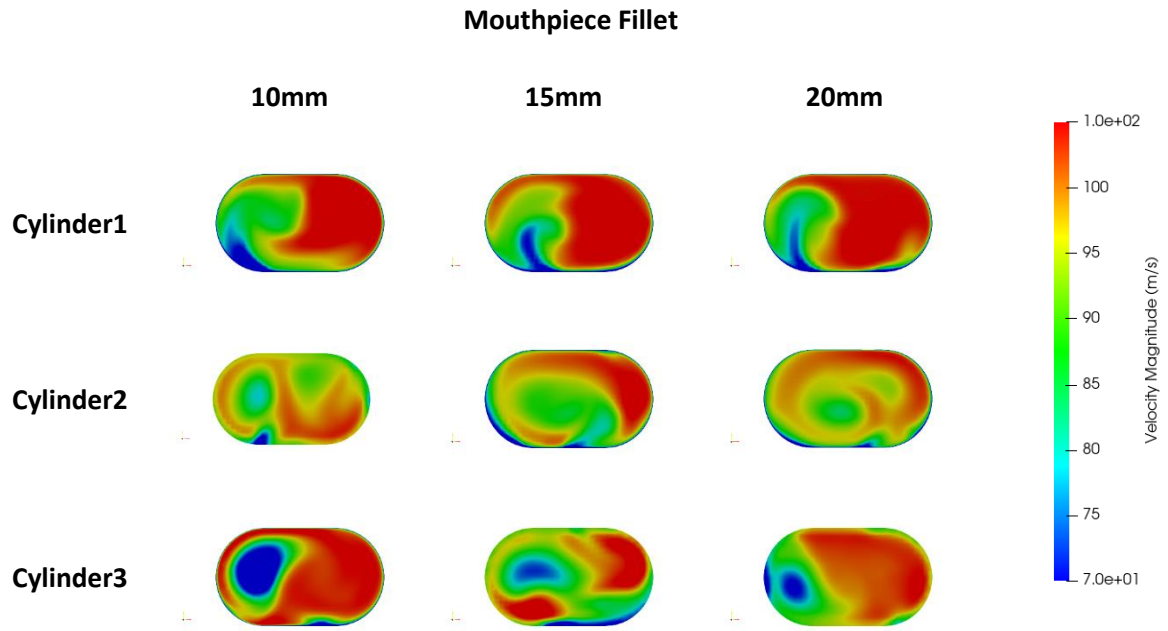
$$\gamma = 1 - \frac{\sum_f \emptyset_f - \emptyset}{2|\emptyset| \sum_f A_f}$$

Where ∅ is the surface average of the scalar, $\emptyset_f$ is the value of the scalar on the face and A_f is the area of the face.

4.1 Mouthpiece Fillet simulation results

The following table shows the velocity field distribution at the outlet of each runner for the different fillet dimension:

Table 12. Outlet velocity distribution for different fillet radius.



The velocity uniformity index has been computed for each outlet section velocity distribution:

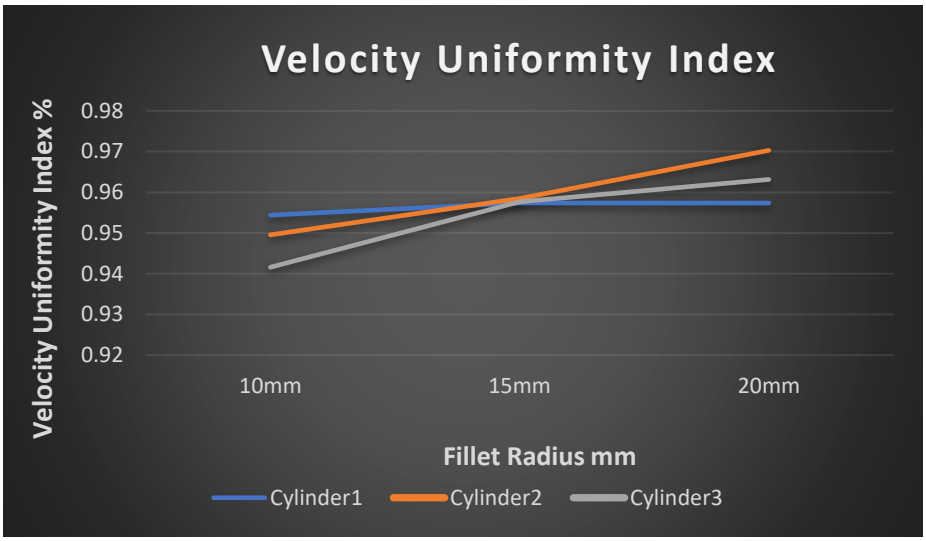


Diagram 13. Velocity uniformity index for different fillet radius dimensions.

For all the cylinders, by increasing the fillet radius, it increases the velocity uniformity at the outlet. The first runner is the one that shows the lower improvement since the airflow coming from the inlet is directed toward the first cylinder: as a result, it shows a lower sensibility to this geometrical parameter variation. Instead, both the cylinders 2 and 3 show an improvement of about two percent whit respect to the base case of 10mm.

The reason is due to a better flow detachment from the surfaces of the runner:

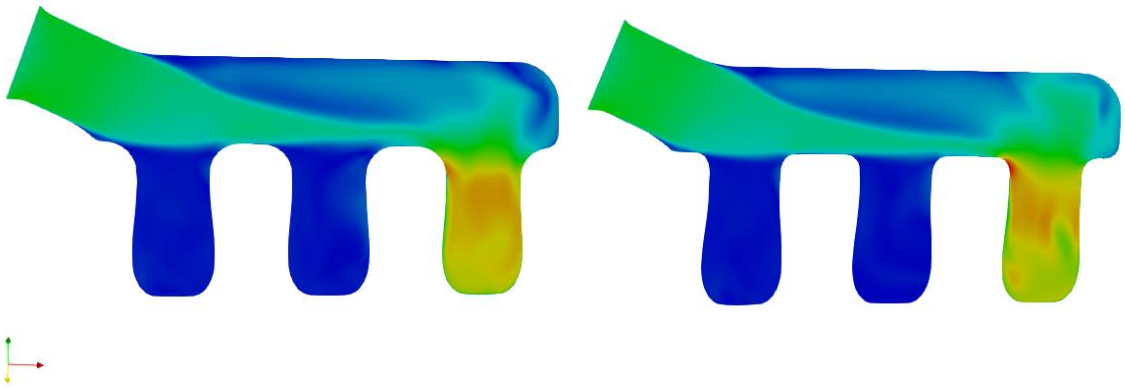


Figure 28. Flow detachment from the third cylinder for 20mm (left) and 10mm (right).

In case of 10 mm the flow detaches in an upper position with respect to the 20mm fillet radius, and it also reaches higher local velocity because a smaller fillet radius produces a reduction of the entrance section of the flow, therefore it observes a local increase of speed that produces higher velocity gradients and so a reduction in the velocity uniformity distribution at the outlet.

For what concerns the mass flow, the following diagram shows the trend against the fillet radius:

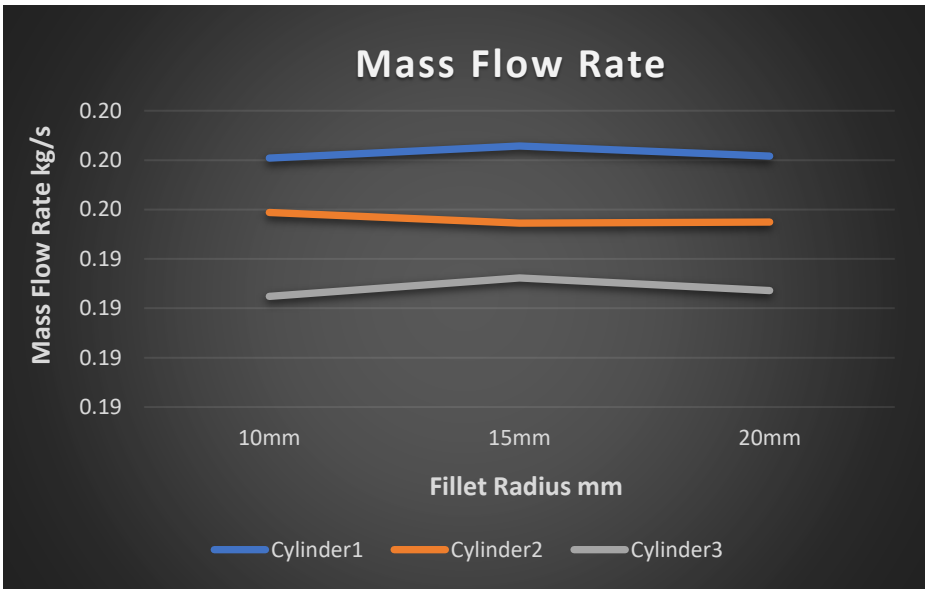


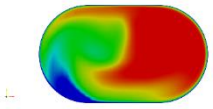
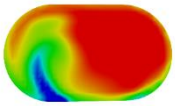
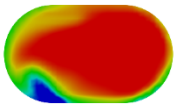
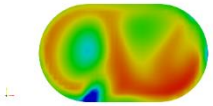
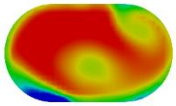
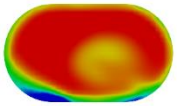
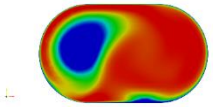
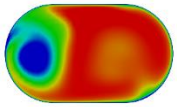
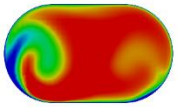
Diagram 14. Mass Flow Rate for Different fillet radius dimensions.

The fillet radius increase does not produce an improvement of the mass flow.

4.2 Inlet Section Variation simulation results

The table below shows the outlet velocity distribution passing from 60 to 82 mm:

Table 13. Outlet Velocity Distribution for different inlet section diameter.

	Inlet Section Diameter		
	60mm	**68mm**	**82mm**
Cylinder 1			
Cylinder 2			
Cylinder 3			

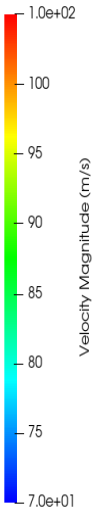


Table 14. Outlet velocity distribution for different inlet cross section diameter.

The velocity index as a function of the inlet cross section diameter is shown below:

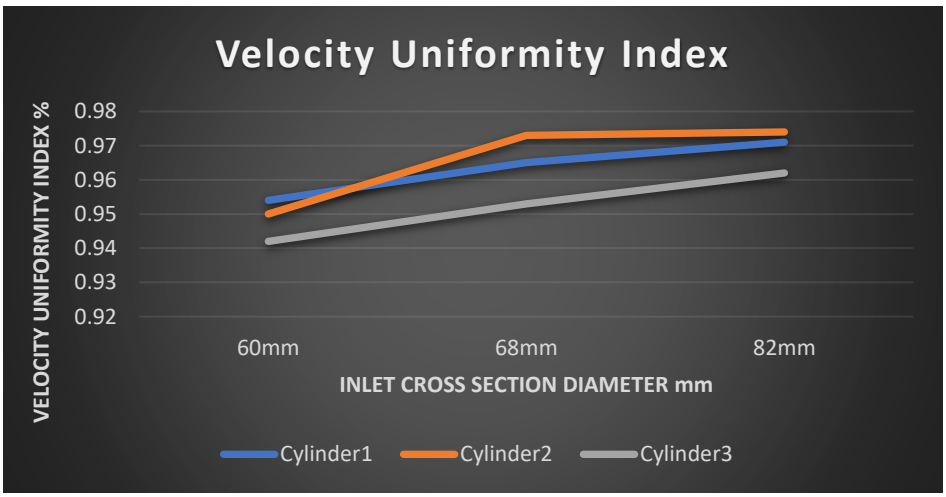


Diagram 15. Velocity Uniformity Index for different inlet cross section diameters.

Increasing the Inlet Cross Section Diameter improves the velocity uniformity index for all the runners, but the runner 2 is the more affected one due to this parameter variation.

Considering the mass flow:

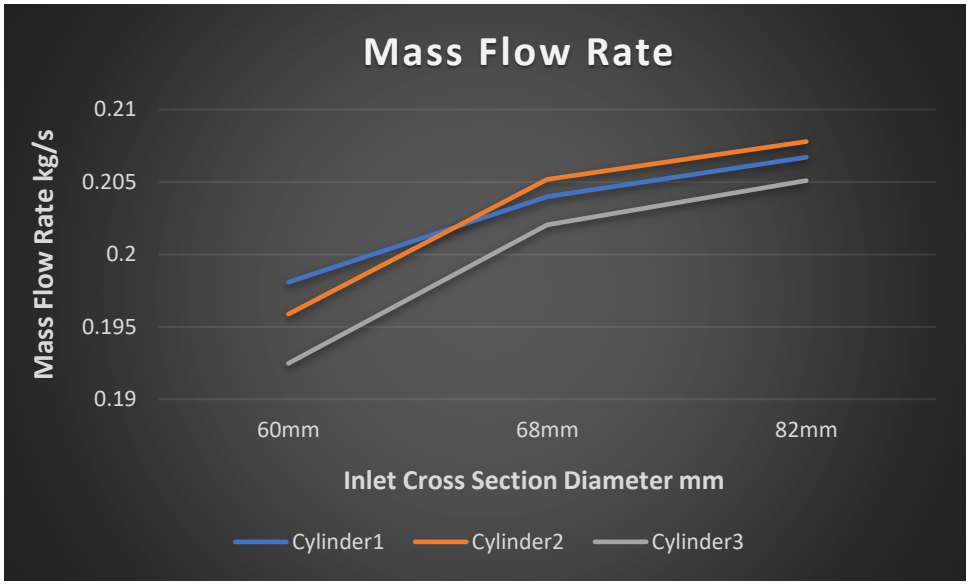


Diagram 16. Mass Flow Rate for different inlet cross section diameters.

The mass flow shows an almost linear increase for the cylinder 1 and 3. Moreover, for the 82mm diameter, the cylinder 1 and 2 have almost the same mass flow while the cylinder 3 is closer, in terms of mass flow, to the cylinder 1. Therefore, by increasing the inlet cross section diameter, the balancing of mass flow between the three cylinders, improves of 5.5% for the first cylinder and of 6% for the remaining two.

4.3 Conclusions on the geometry Optimization

The diagram below compares the velocity uniformity index of each cylinder for the different fillet radius examined:

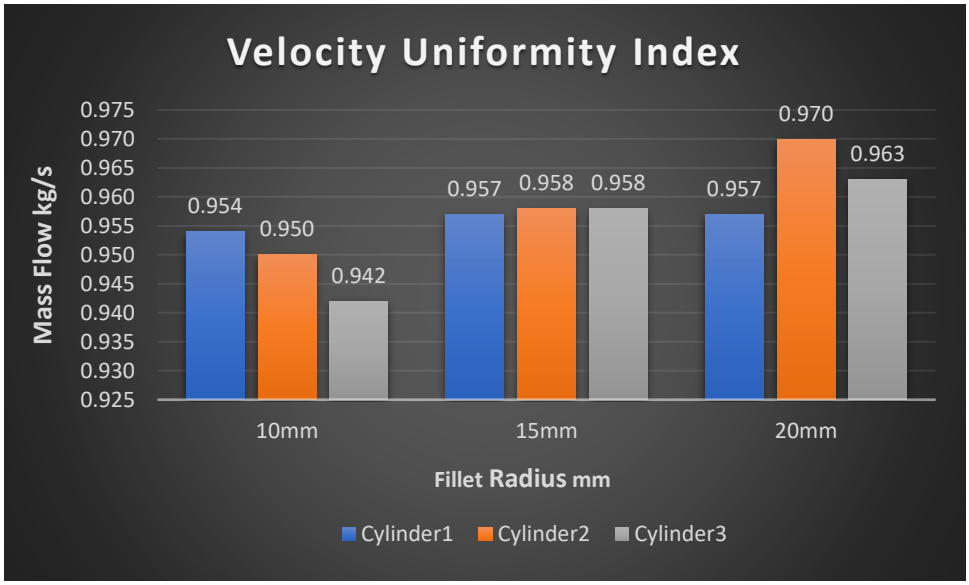


Diagram 17. Comparison between the different cylinder in terms of velocity uniformity index as a function of the fillet radius.

In terms of absolute values, the increase of the fillet radius increases the velocity uniformity index, but for 20mm fillet the velocity uniformity index of the cylinder 2 is higher with respect the remaining cylinders. Since the final goal is the achievement of a balanced air distribution among all cylinders, it is more convenient to adopt a fillet radius of 15 mm. For what concerns the mass flow, the fillet radius doesn't affect significantly its magnitude.

As far as the inlet cross section diameter is concerned:

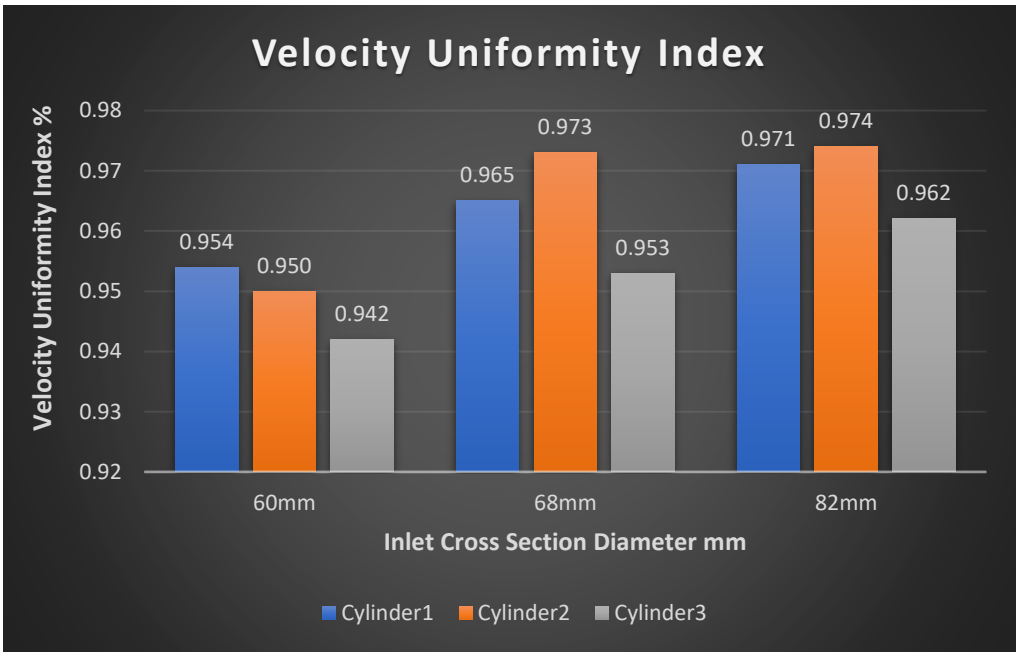


Diagram 18. Comparison between the different cylinder in terms of velocity uniformity index for different inlet cross section diameters.

The diagram shows how the bigger is the inlet cross section diameter, the higher is the uniformity index. The same is true for the mass flow:

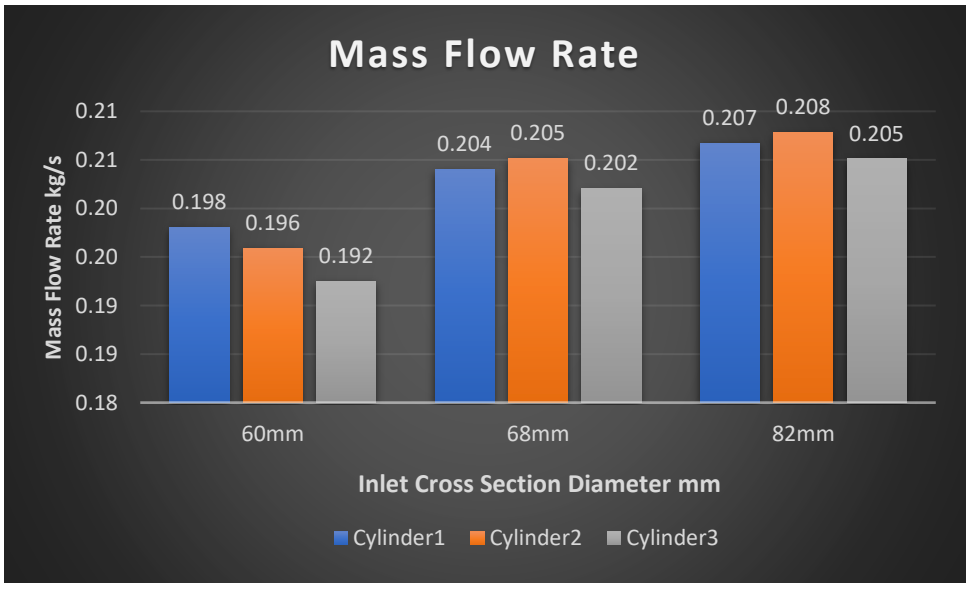


Diagram 19. Comparison between the different cylinder in terms of mass flow rate for different inlet cross section diameters.

Moreover, by increasing the inlet cross section diameter it balances the mass flow of each cylinder with respect to the base case of 60mm. Anyway, it has to consider that increasing the inlet cross section diameter means to increase the plate of the throttle valve, that has different consequences:

- this increment must be compatible with the engine layout in terms of bulk for the other engine components.

- the increase of the plate of the throttle valve increases the moment of inertia of the plate itself, and therefore its response can be slower.

- it decreases the capability of controlling the airflow, because the higher is the plate diameter, the higher is the availability of area for the motion of the flow even at the minimum throttle valve opening.

In conclusion, it has to consider as 15mm the best fillet radius and 68mm the best compromise for the air flow inlet cross section diameter. On the basis of these considerations, the final layout has been developed:

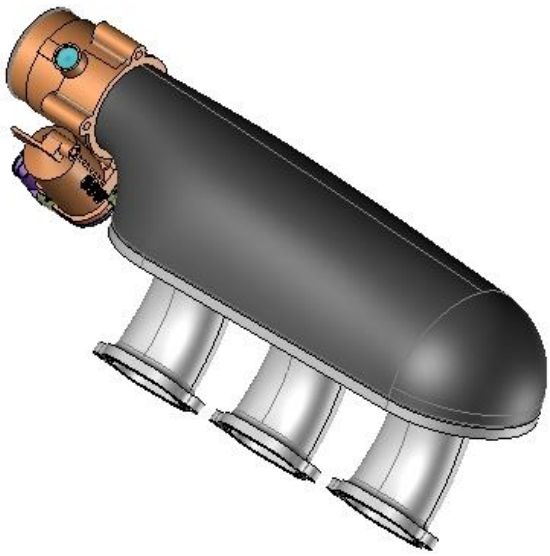


Figure 29. Final intake manifold layout.

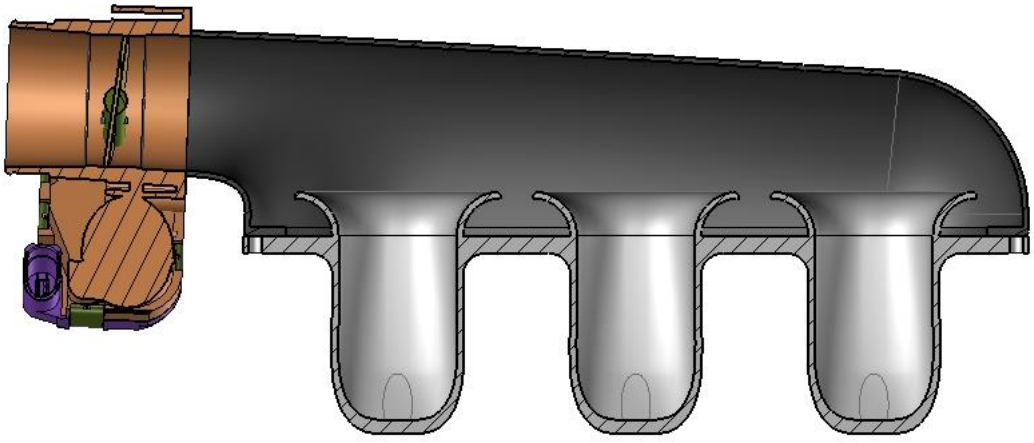


Figure 30. Section of the final layout.

5 TRANSIENT MODEL

To analyze the dynamic performance of the intake manifold in terms of percentage variation of mass flow that flows towards the cylinders, a transient CFD model has been developed on OpenFOAM. The main aim of the transient model is to evaluate the performance of the intake manifold in terms of balancing of mass flow that flows outside each runner during the engine cycle.

To perform a comparison between the transient model results and experimental data, it is necessary to acquire the following data:

- Airflow rate that flows inside the cylinders: this can be done by installing a lambda sensor on each exhaust manifold runner of the same engine in such a way to have an indirect information of the unbalance of each cylinder.

- Pressure measurements by means of high-speed pressure sensors of the pressure traces at the outlet of the intake manifold, so in the region of entrance of the engine head.

Regarding the unbalance between the different cylinders, an experimental activity has been done on the ATM-AR-F3R EVO engine, a different version of the ATM-AR-F3R engine: it has a bigger turbocharger group size to increase the boost pressure and then the power. A lambda sensor is installed on each runner of the exhaust manifold:

Figure 31. Lambda sensors on the ATM-AR-F3R EVO exhaust manifold.

During the test, to make the different cylinders of the engine to operate at the same lambda, it was necessary to make a correction on the amount of fuel injected in each cylinder. The following table shows the correction:

Table 14. Fuel Corrective Coefficient.

	Cyl1	Cyl2	Cyl3	Cyl4
3000	1.007	1	0.994	0.999
3250	1.007	0.997	0.993	1.004
3500	1.014	0.996	0.988	1.002
3750	1.008	0.996	0.989	1.006
4000	1.002	0.999	0.994	1.004
4250	1.004	0.997	0.992	1.007
4500	1.002	0.996	0.993	1.008
4750	1.002	0.996	0.992	1.01
5000	0.999	0.997	0.992	1.012
5250	0.999	0.998	0.991	1.013
5500	1	0.997	0.99	1.013
5750	0.999	1	0.992	1.009
6000	0.997	1	0.991	1.014
6250	1.003	0.999	0.987	1.011
6500	1	1.002	0.998	1.012

By injecting the same amount of fuel in each cylinder, the lambda measurement gives an estimation of the percentage variation of air that enters inside the cylinder with respect to the amount of air that corresponds to the target lambda. The corrective coefficients are, for each engine revolution speed tested, very close to the unity: the higher unbalance is of 1.2%.

Regarding the pressure, there was not the possibility to perform these measurements at the engine test bench, but it is an activity that could be executed in the future. Therefore it is necessary to

obtain the needed input for the model from an already existing validated 1D model of an engine of AutotecnicaMotori: the ATM-AR-F3R EVO engine. The following diagrams show how the 1D model well approximates the real engine behavior in terms of torque, air mass flow rate entering inside the cylinder, brake mean effective pressure and brake specific fuel consumption:

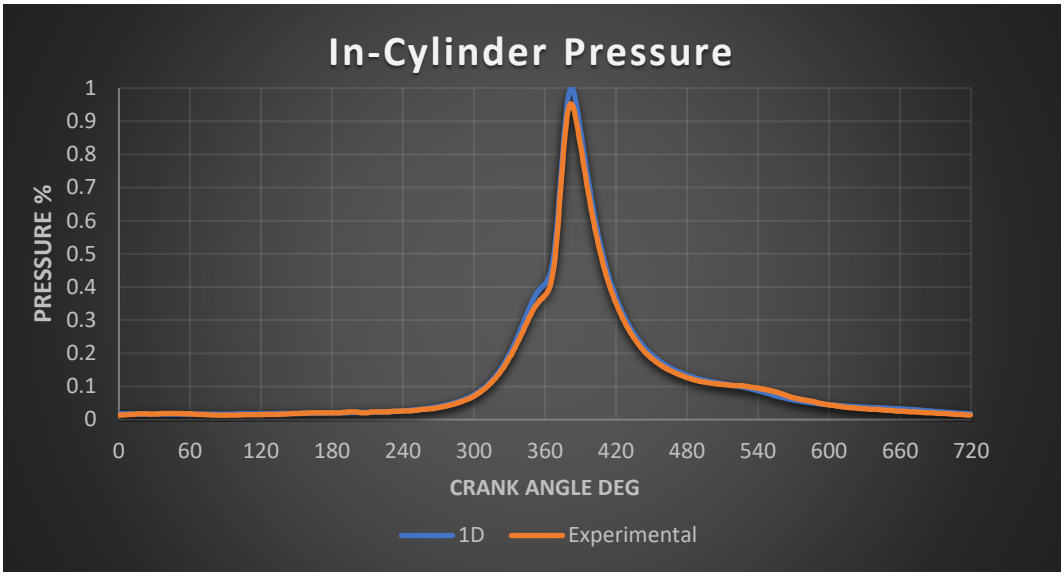


Diagram 20. In-cylinder pressure for 5000 RPM.

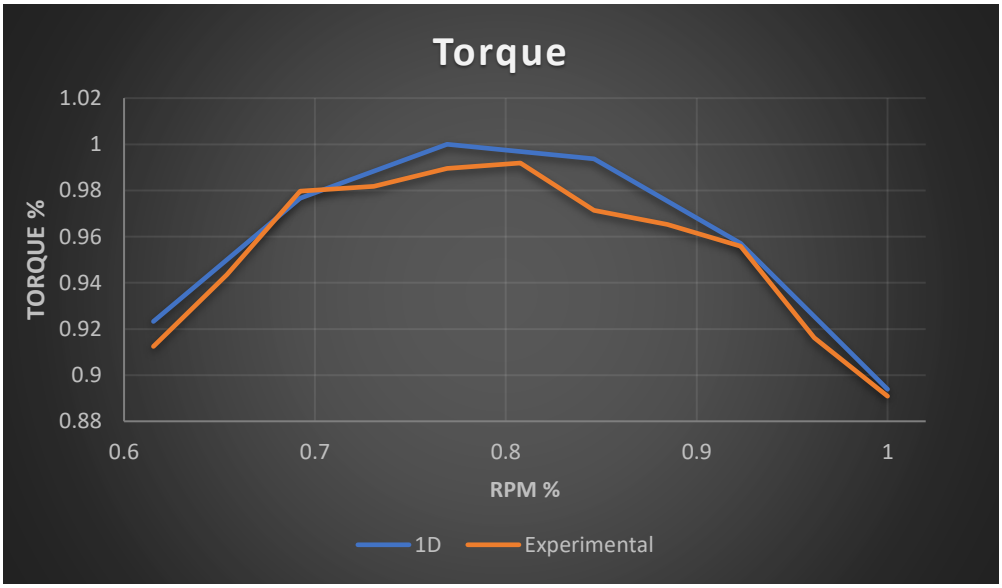


Diagram 21. Maximum Torque characteristic.

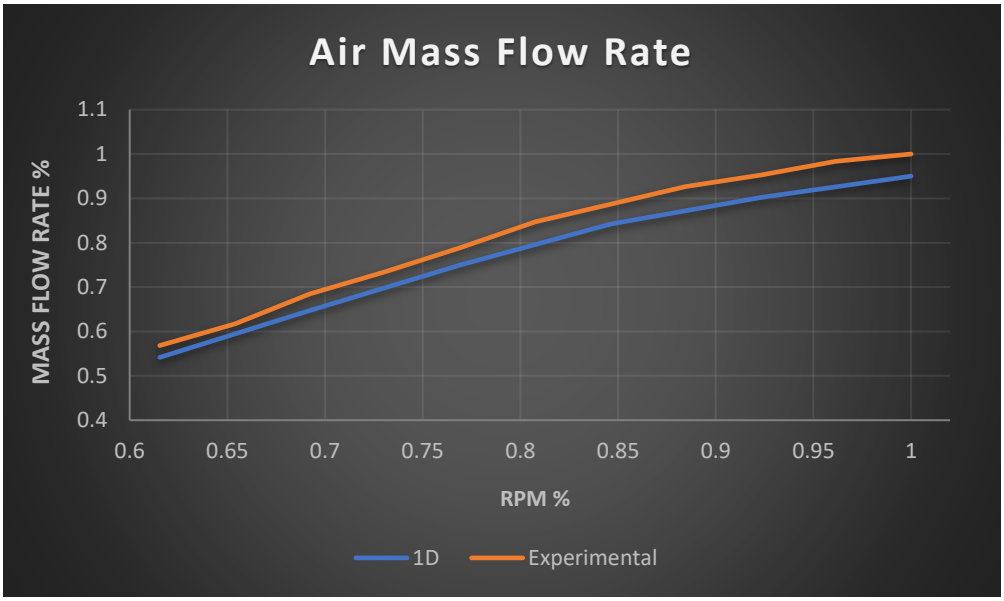


Diagram 22. Air mass Flow Rate comparison 1D - experimental measurements.

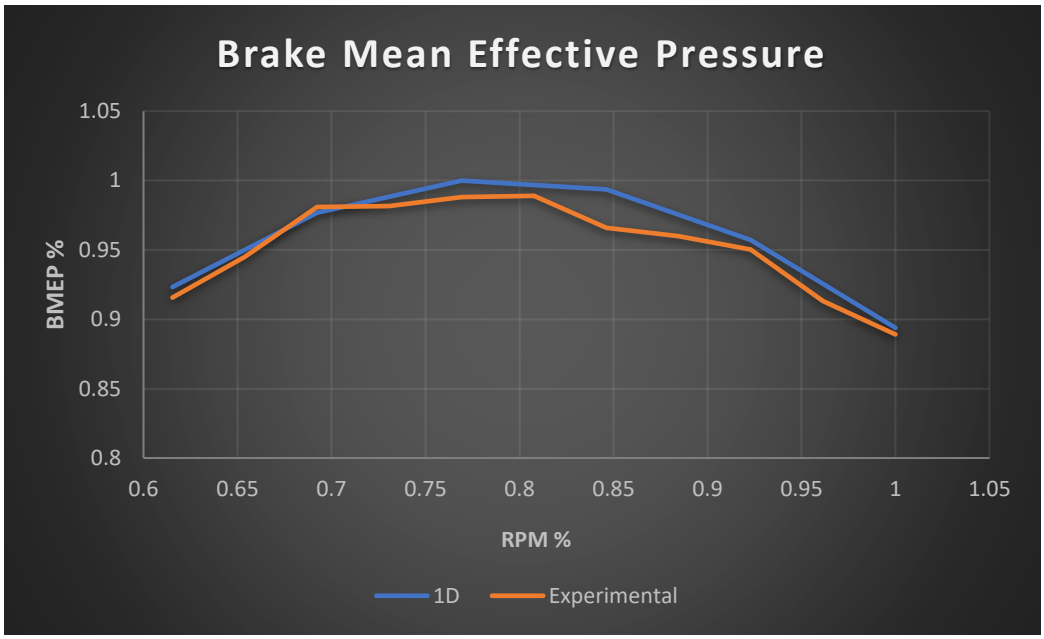


Diagram 23. Bmep comparison between 1D - experimental data.

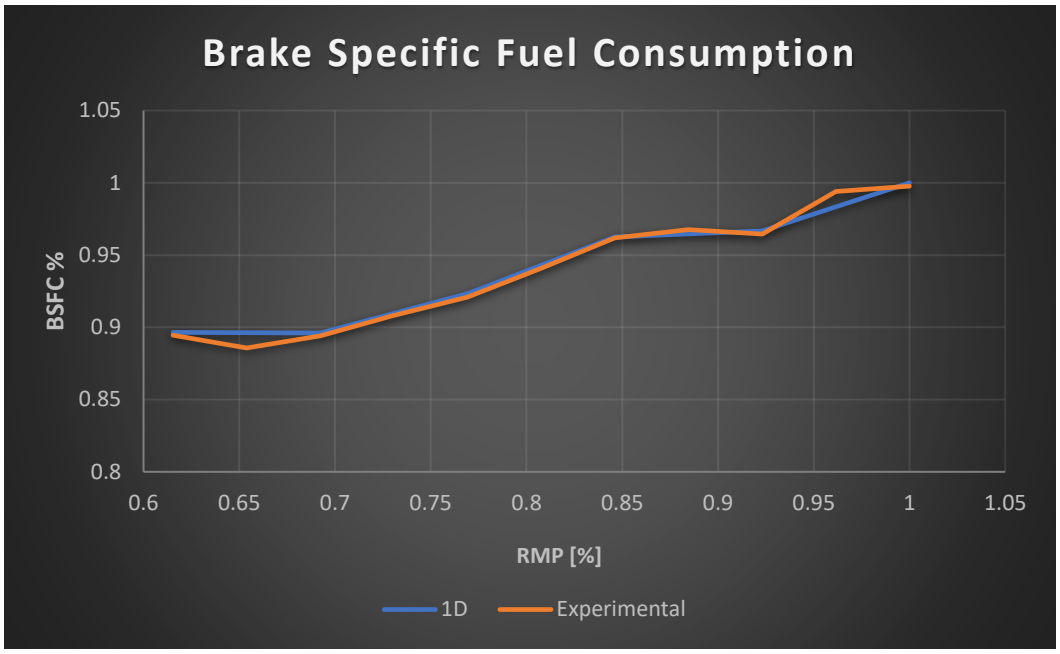


Diagram 24. Bsfc comparison between 1D - experimental data.

The goal of the transient simulation is to evaluate the mass flow percentage variation of each cylinder with respect to the target value of mass flow. The target mass flow value is obtained by the mass flow rate law coming from the 1D model. Since it is dealing with a 4 cylinders engine and since the mass flow imposed at the inlet refers to the whole engine, it has to be divided by 4: this is the target value from which it is possible to compute the percentage variation of air flow rate that goes out from the intake manifold runners.

5.1 Meshing Process

The geometry it will use to attempt the transient model validation is the same geometry used in the steady state validation, therefore no editing of the geometry has been necessary.

The mesh parameters are different with respect to the steady-state model:

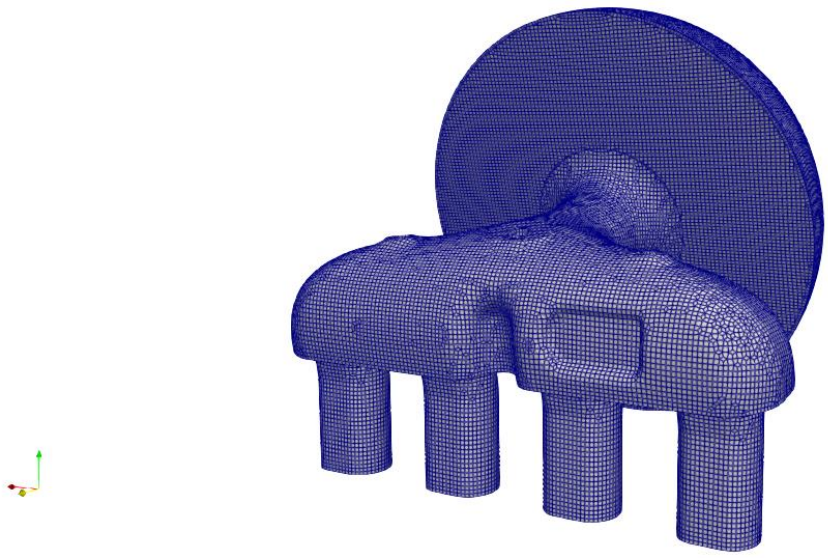


Figure 32. F3 Regional Intake manifold mesh for the transient model.

Table 15. Mesh Parameters.

	K-Omega SST
Plenum Mesh Size	3mm
Atmosphere Volume Mesh size	3mm
Number of prism layer	6
Prism layer Expansion ratio	1.3
Total Number of cells	550000

The mesh size has been strongly increased with respect to the steady-state simulation due to the necessity of having an acceptable computational time. This because in a transient simulation it is necessary to keep the maximum Courant-Friedrichs-Lewy number (CFL number) below to 1 if it does not want to lose information about the flow field evolution. Considering the CFL number expression for a n-dimensional case:

$$C = \Delta t \sum_{i=1}^{n} \frac{u_{xi}}{\Delta x_i} \leq C_{max} \tag{5.1}$$

Where u represents the velocity of the flow while Δx_i represents the spatial interval. Therefore, by decreasing the mesh size, so the Δx_i, if it wants to keep C_{max} below 1, it is necessary to decrease the time step, but if the time step is too small the computational time increases excessively, especially when the speed of the flow increases.

The time step has been set to an adaptive time step: it means the time step will adapt at each iteration considering the new flow speed, always keeping the maximum Courant Number below 1.

5.2 Boundary Conditions

The boundary conditions imposed in the transient simulations are almost the same of the ones imposed in the steady-state model validation:

Table 16. Boundary Conditions.

	Inlet	Outlet	Wall
T	Total Temperature=294,65 °K	InletOutlet	zeroGradient
V	massFlow	PressureInletOutletVelocity	uniformFixedValue (0 0 0)
P	zeroGradient	uniformFixedValue	zeroGradient
k	turbulentIntensityKineticEnergyInlet	inletOutlet	kqRWallFunction
nut	calculated	Calculated	nutkWallFunction
omega	turbulentMixingLengthFrequencyInlet	inletOutlet	omegaWallFunction
alphat	calculated	Calculated	compressible::alphatWallFunction

They are equal to the boundary conditions used in the steady-state model, with the difference of introducing a time-varying mass flow rate at the inlet and a time varying pressure at the outlet. Moreover, in this simulation, since it wants to reproduce the operation of an intake manifold, all the runners will be considered as outlet. As a result, it will have four outlets.

5.3 Solver

The solver it chooses for the transient simulation is called rhoPimpleFoam. It is a transient solver for compressible flow, and it adopts the Pimple algorithm, that is a combination of the SIMPLE algorithm and of the PISO algorithm. It is possible to exploit the rhoPimpleFoam solver in two different ways:

- PISO algorithm.
- PIMPLE algorithm.

The Pressure Implicit with Split Operator (PISO) algorithm involves one predictor step and two corrector steps. The PISO algorithm can be summarized in the following steps:

1. It sets the boundary conditions.
2. It solves the discretized momentum equation to compute an intermediate velocity field.
3. It computes the mass fluxes at the cell faces.
4. It solves the pressure equation.
5. It corrects the mass fluxes at the cell faces.
6. It corrects the velocities based on the new pressure field.
7. Update of the boundary conditions.

The last 5 steps are repeated for a prescribed number that is set by the keyword nCorrectors in in the fvSolution text file of OpenFOAM:

```
PIMPLE
{
    momentumPredictor   yes;
    transonic           no;
    nOuterCorrectors    50;
    nCorrectors         2;
    nNonOrthogonalCorrectors 4;
    consistent          no;
    SIMPLErho           yes;

    pMaxFactor          1.5;
    pMinFactor          0.9;

    residualControl
    {
        "(U|k|omega|p|h)"
        {
            relTol          0;
            tolerance       1e-5;
        }
    }
    turbOnFinalIterOnly no;
}

relaxationFactors
{
    fields
    {
        "p.*"           0.3;
        "rho.*"         1;
    }
    equations
    {
        "U.*"           0.7;
        "h.*"           0.3;
        "(k|omega).*" 0.3;
    }
```

Figure 33. fvSolution text file.

By the *nCorrectors* dictionary it sets the number of times for which the last 5 steps are repeated. It is also possible to increase the number of times in which the pressure equation and the mass fluxes correction at the cell faces is performed: this is done by the sub-dictionary *nNonOrthogoalCorrector*s. Once repeated the set number of times the steps from 3 to 8, it increases the time step and it restarts the loop from the point 1. The step 2 is called the momentum predictor step and it is performed only if the subdictionary *momentumPredictor* is set on yes.

The PIMPLE algorithm, instead, is similar to the PISO algorithm, with the difference that once it reaches to the step 8, it restarts by computing a new intermediate velocity field in the same time

step: it is therefore the same procedure done by the SIMPLE algorithm, with the only difference that it is performed inside a single time step. Therefore, the Pimple algorithm adopts the PISO algorithm to achieve the final velocity correction ad then update the boundary conditions, but it also gives the possibility to restart a new iteration in the same time step: the new velocity field computed at the step 7 is the new velocity that is used to set the boundary conditions at the step 1: this is what the SIMPLE algorithm does. For this reason, it can be considered a combination of the PISO and of the SIMPLE algorithm.

The number of times the PIMPLE algorithm performs iterations inside the time step is defined by the keywords nOuterCorrectors.

Both the algorithms will be tested for the transient model validation.

5.4 Simulation Results – PIMPLE algorithm

The pimple algorithm needs 108 hours to complete four engine cycles with 50 nCorrectors and the residuals set at 10e-04.

The diagram below shows the instantaneous mass flow rate during the engine cycle for each cylinder obtained by the PIMPLE algorithm:

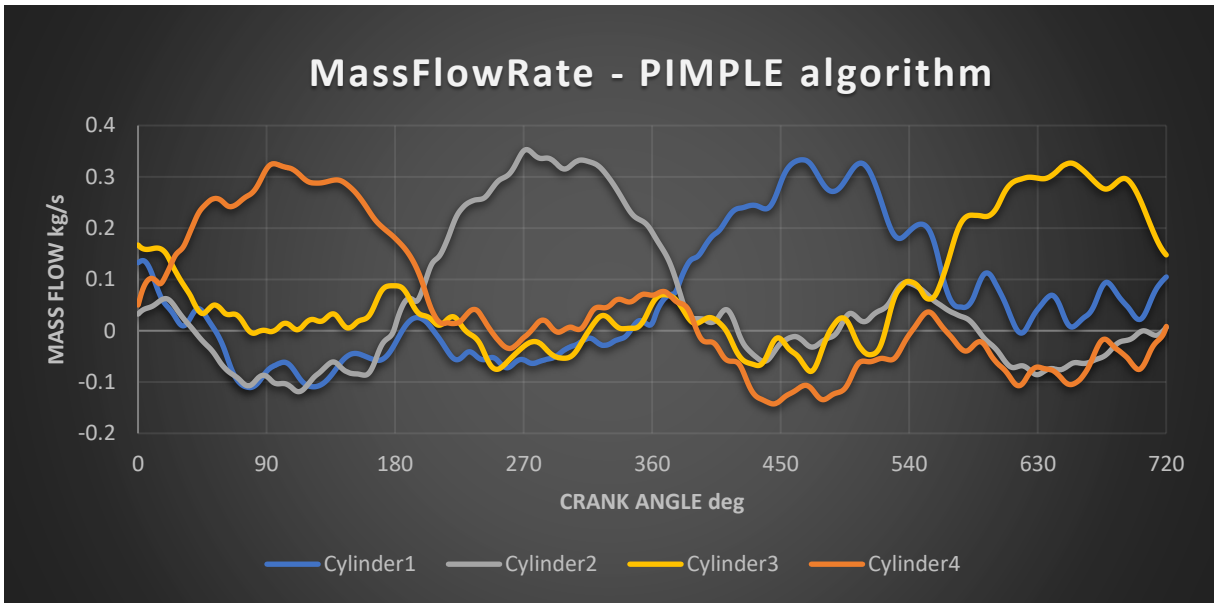


Diagram 25. Cylinder's mass flow rate traces computed by the 3D model with the PIMPLE algorithm.

Zero crank angle represents the start of combustion of the cylinder 1. Each cylinder shows a mass flow rate pick, that correspond to its intake stroke. The instantaneous mass flow can be compared with the instantaneous mass flow rate coming from the 1D model:

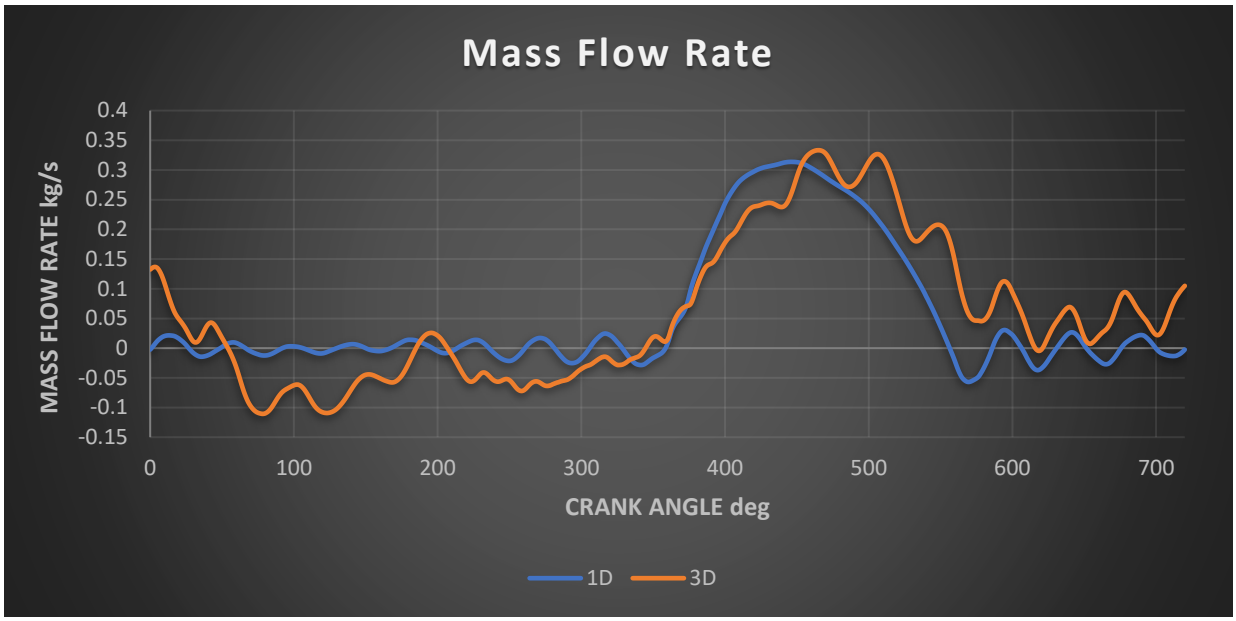


Diagram 26. Comparison 3D - 1D Mass Flow Rate.

The two mass flow traces are different, especially for what concerns the crank angle interval outside the intake stroke. This is a first important difference that limits the 3D transient model, because the presence of a positive pick around 0 crank angle degree and a negative pick around 100 crank angle degree produces an important oscillation in the stability of the results. In fact, considering two consecutive cycles for the cylinder 1:

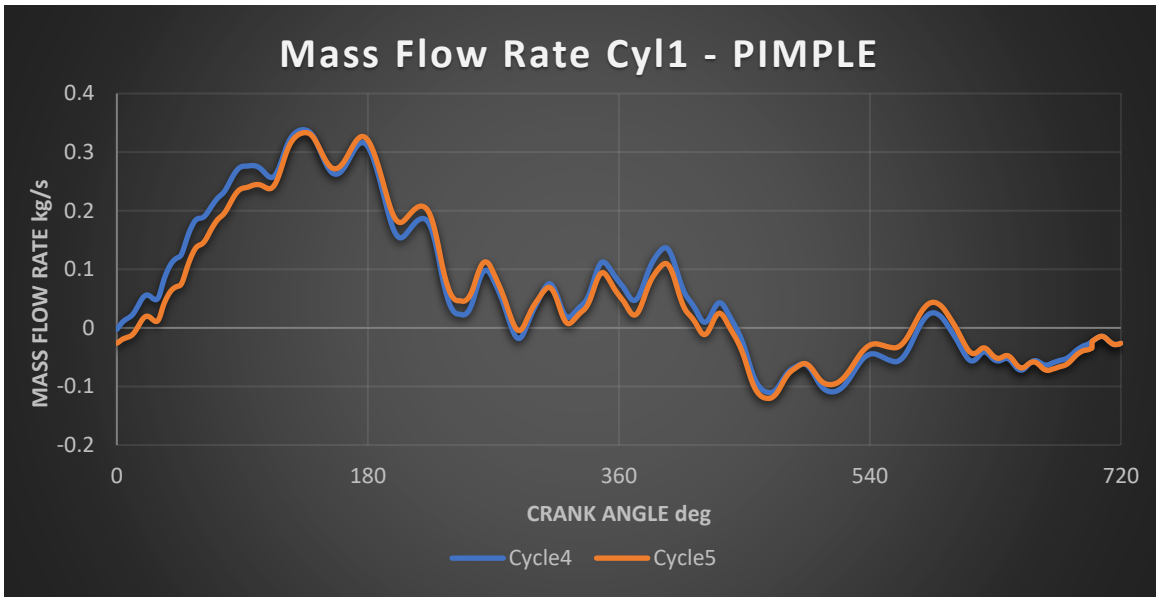


Diagram 27. Comparison Cycle 4 and 5 of the Cylinder 3 by the Pimple algorithm.

It has an important cycle to cycle oscillation. To maintain the correct mass conservation, it is necessary to evaluate the amount of mass that goes into each cylinder by an integral average of the instantaneous mass flow over the engine cycle. If the oscillation of the mass flow in the crank angle interval outside the intake phase was near to zero, it would have a lower impact on the integral average but reaching values about the 40% of the maximum instantaneous mass flow intaken from the cylinder, even a small variation strongly affects the stability of the result. Consequently, since the evaluation of the percentage difference of mass flow with respect to the target is computed on the entire instantaneous mass flow rate during the engine cycle, the PIMPLE algorithm does not produce enough stable results.

5.5 Simulation Results – PISO algorithm

The PISO algorithm takes 72 hours to perform 12 cycles: it is substantially faster than the PIMPLE algorithm for the reasons explained in the section 5.4. The diagram below shows the instantaneous mass flow rate of each cylinder during the engine cycle: zero crank angle degree corresponds to the start of combustion of the cylinder 1.

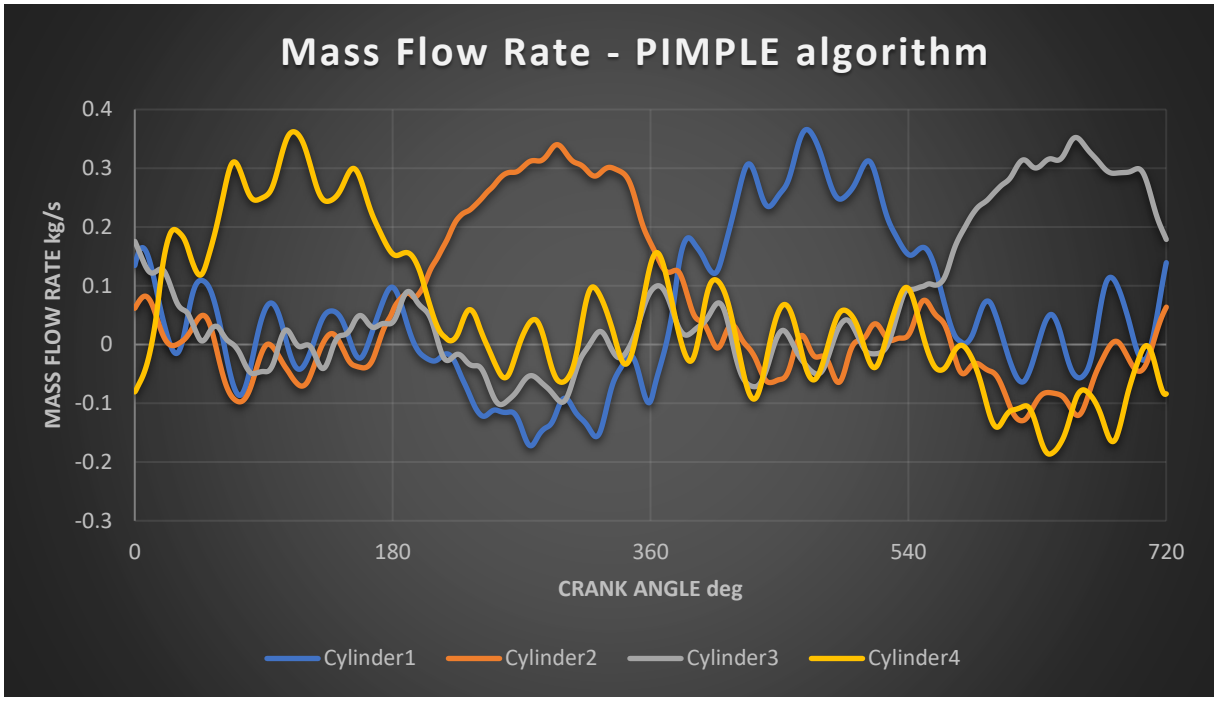


Diagram 28. Instantaneous mass flow rate for the PISO algorithm.

Considering the cycle-to-cycle variability of the cylinder 3:

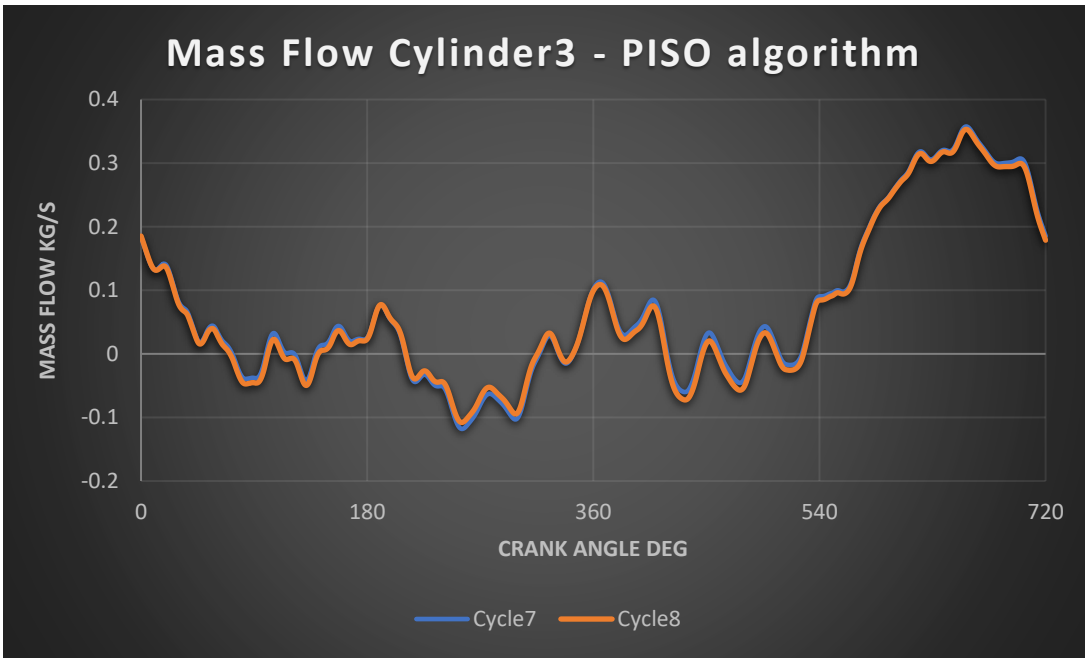
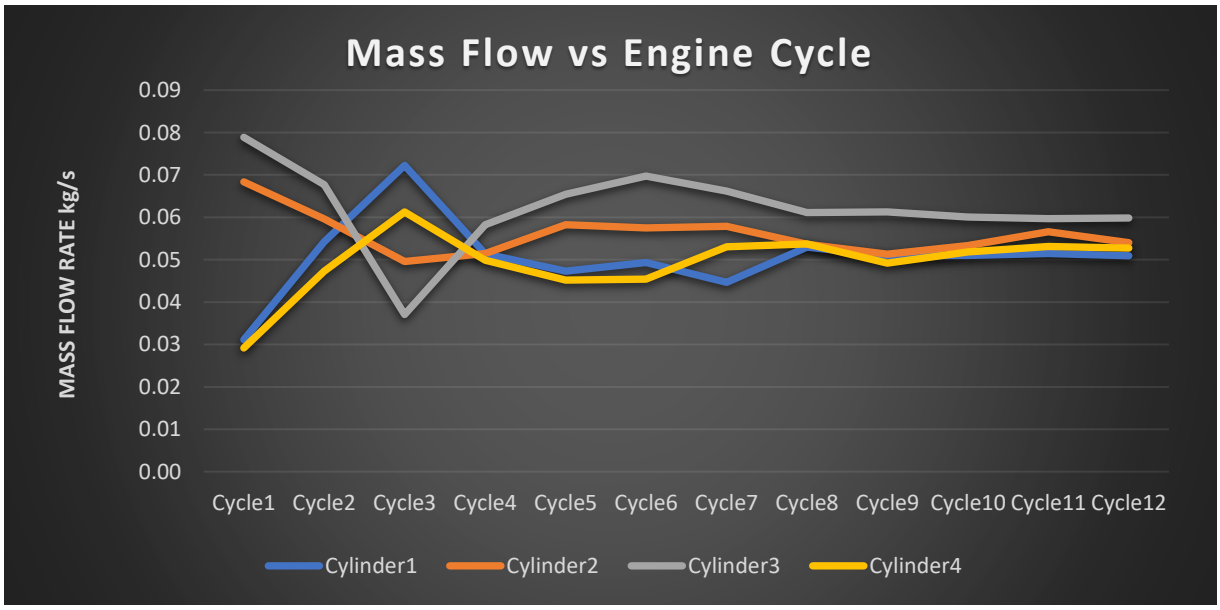


Diagram 29. Comparison Cycle 7 and 8 for the Cylinder 3 - PISO algorithm.

Considering the trend, by the PISO algorithm it achieves a good stability in the trend of the mass flow since the curves are overlapped. Considering the mass flow rate of each cylinder during the simulated engine cycles:



The first five cycles show a high instability of the mass flow, due to the fact the simulation starts from a static condition in terms of velocity and with a uniform temperature of 305°K and a uniform pressure of 2.5 Bar. After the initial oscillation, the mass flow rate stabilizes. The lower percentage variation is computed for the cylinders 2 and 4 while the cylinder 1 and 4 show a percentage variation respectively of -7.5% and 8.7%.

The histogram below shows the percentage variation trend during all the simulated cycles:

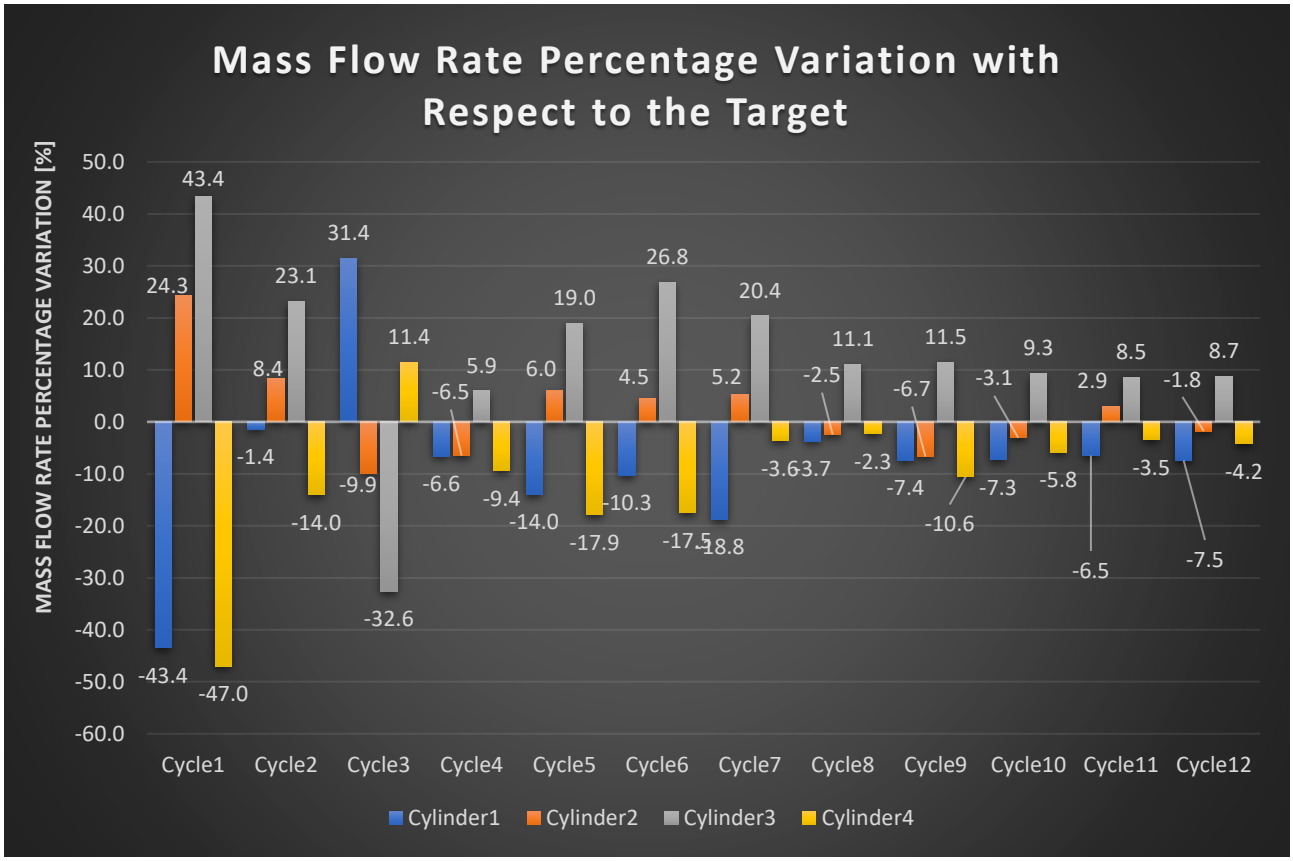


Diagram 30. Percentage Variation with respect to the target for the different simulated cycles.

By the transient simulation it perceives a percentage variation with respect to the target higher than the measured one, and therefore the transient model can not be used to make evaluations on the performance of mass flow rate balance between the different cylinders. Anyway, it has to consider that the percentage variation has been computed by the percentage of variation of the integral average of the instantaneous mass flow over the engine cycle and the target mass flow. The integral

average over the engine cycle is affected by the numerical error done by the software: this numerical error is present in the mass conservation in each time step of sampling of the engine cycle, and therefore the integral average is strongly affected by these numerical errors that alters the final percentage variation. Since it wants to compare percentages, a priori it does not accept a computed percentage variation of about 8% with respect to the percentage variation of 1% computed at the engine test bench: to do it, it would be necessary more experimental data in order to have the possibility to assess a possible correlation between the result coming from the numerical computation of the CFD code and the experimental results. Since there are data of lambda percentage variations only regarding the ATM-AR-F3R EVO engine, it is not possible to attempt any kind of correlation.

5.6 Transient Simulations on the central and lateral geometries

The transient simulation has been performed on the central and lateral geometry to evaluate if with a different kind of geometry, showing similar results to the one done on the ATM-AR-F3R Intake manifold.

The boundary conditions for these simulations come from the in development 1D model of the ATM V6 engine since these intake manifold layouts are conceived for this engine. The diagram below shows the trend of the mass flow of each cylinder during the simulated engine cycles for the maximum torque engine working point:

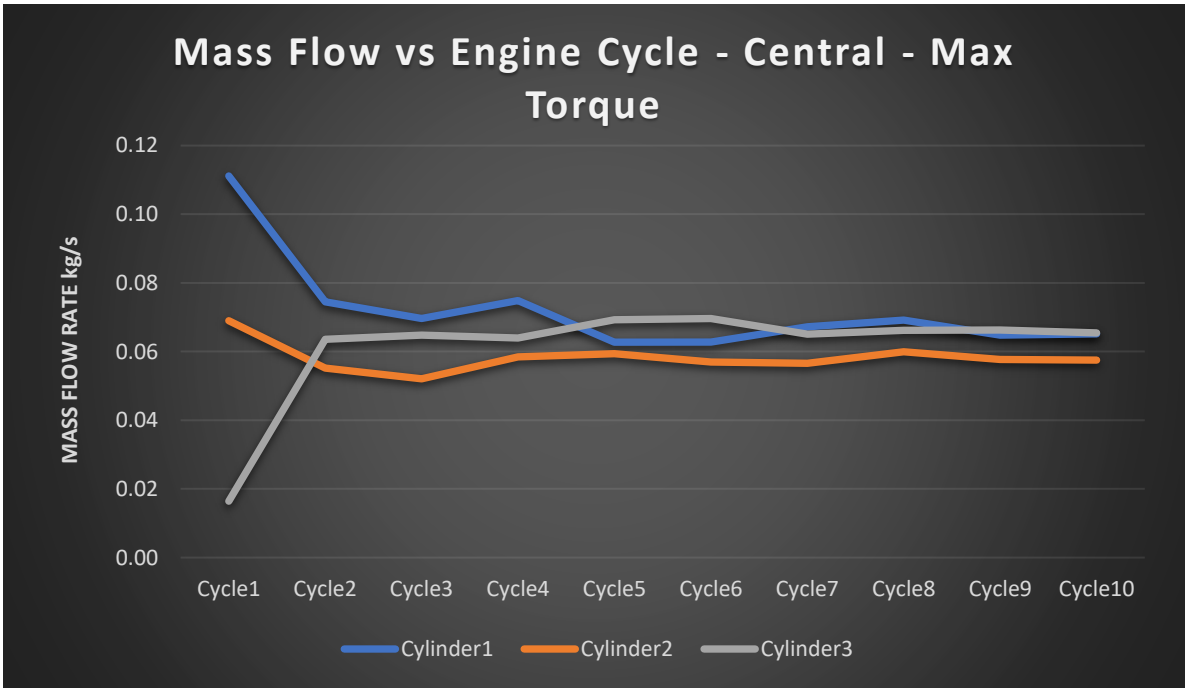


Diagram 31. Cylinder's Mass Flow Rate trend for maximum torque engine point.

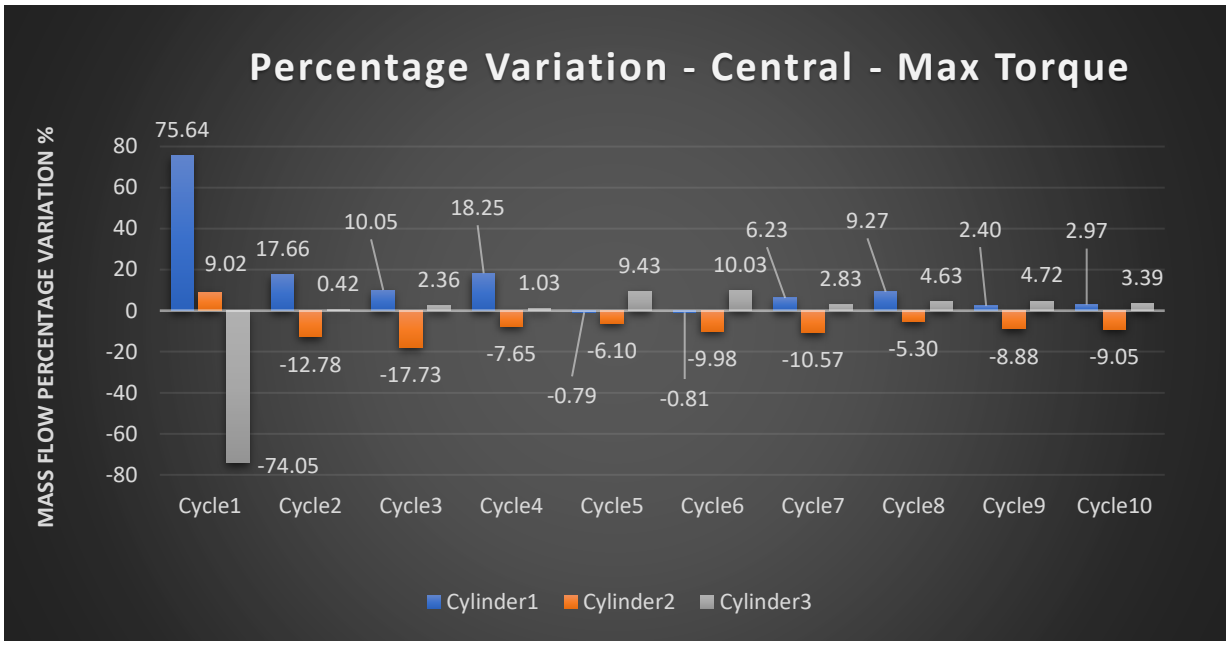


Diagram 32. Mass Flow Rate percentage variations for the lateral geometry at maximum torque conditions.

Instead, for the lateral configuration it gets:

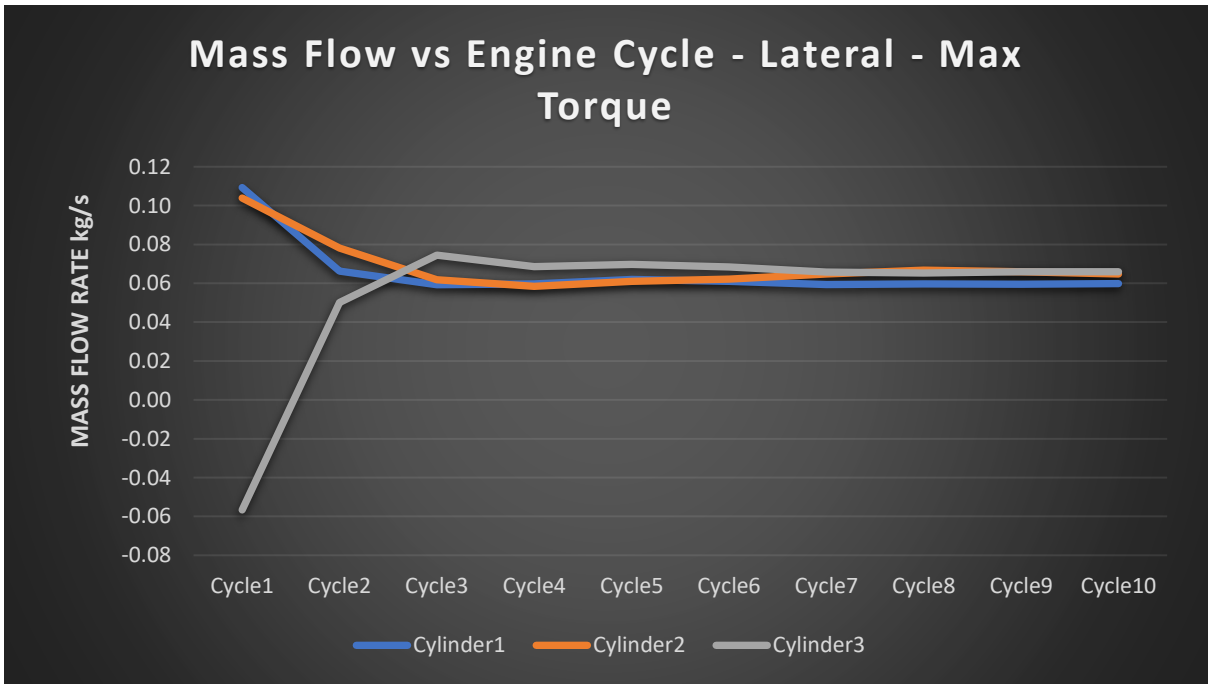


Diagram 33. Mass Flow Rate trend for the lateral geometry at maximum torque conditions.

With the following percentage variation:

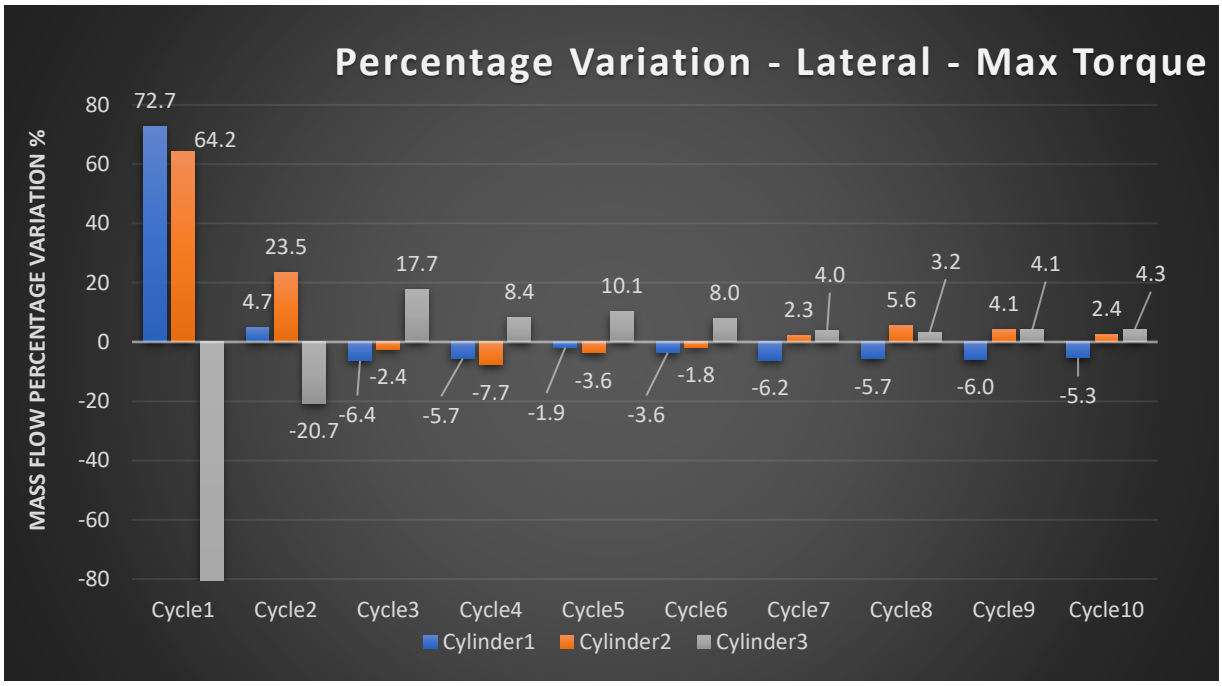


Diagram 34. Cylinder's Mass Flow Rate percentage variation with respect to the target at maximum torque conditions.

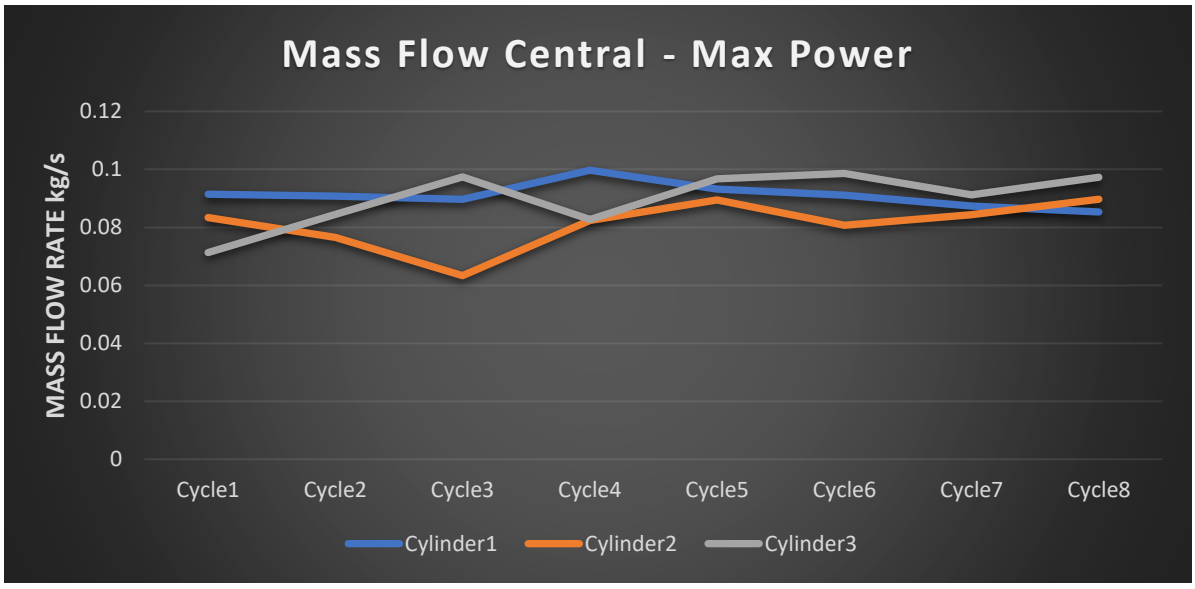


Diagram 35. Mass Flow Rate trend at maximum power contitions for the central configuration.

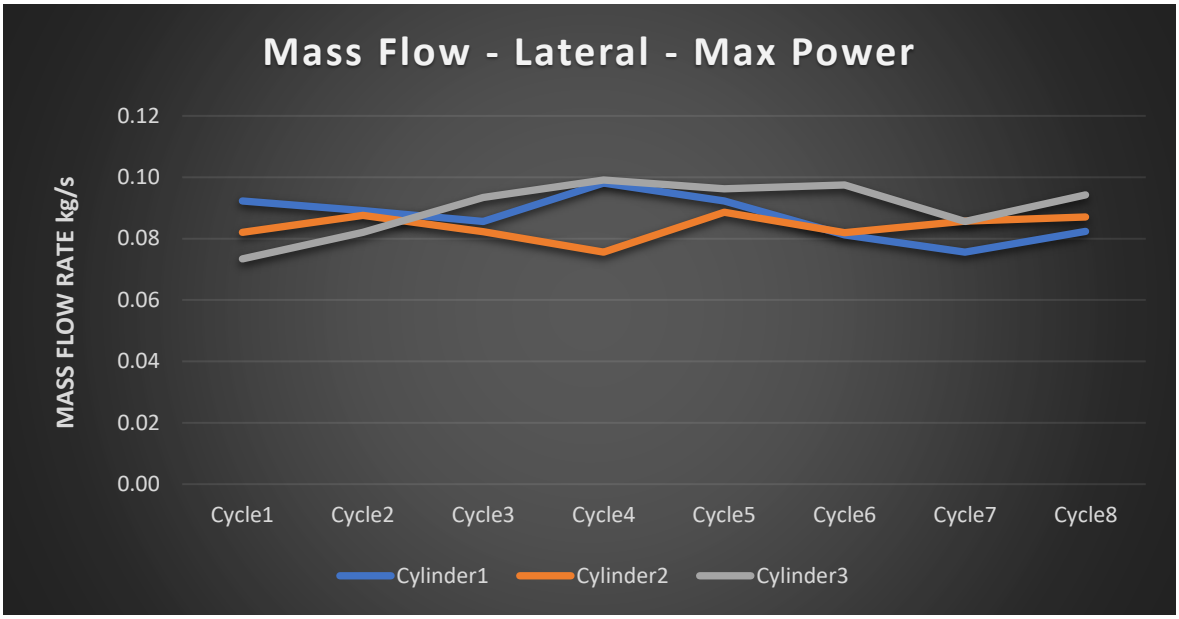


Diagram 36. Mass Flow Rate trend at maximum power for the lateral configuration.

In case of max power conditions, the percentage variation perceived by the simulation is higher than the 10% with respect to the target:

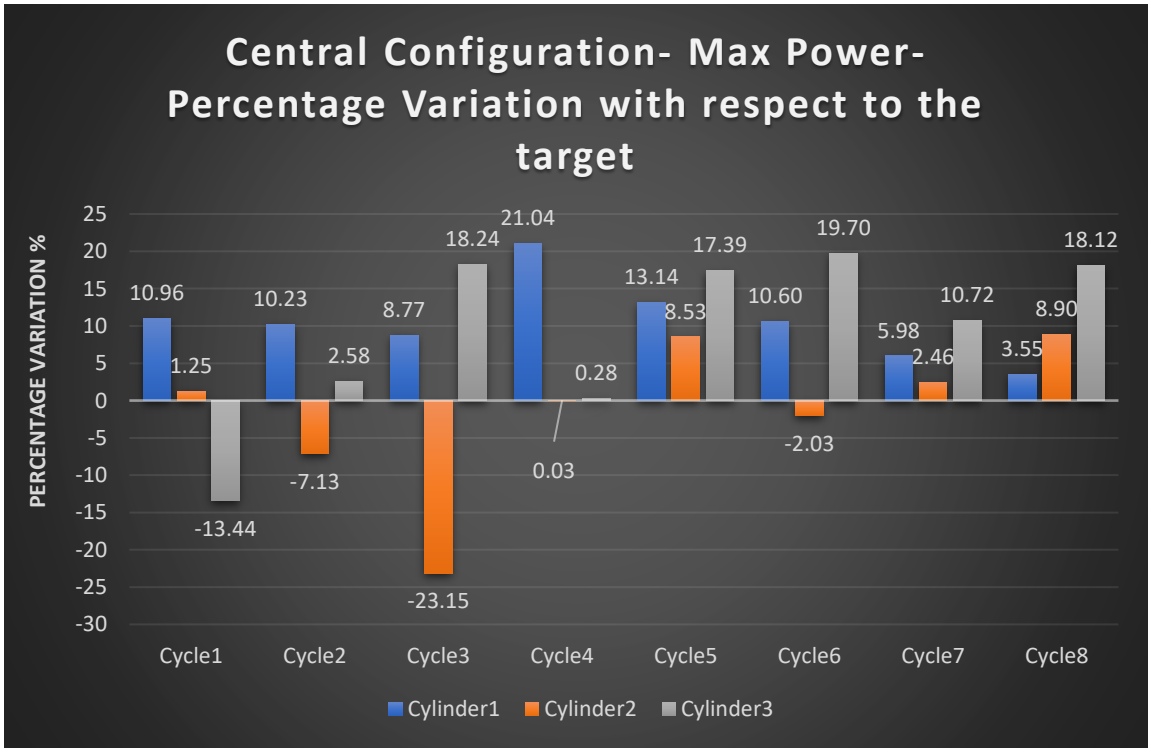


Diagram 37. Mass Flow Rate Percentage Variation with respect to the target for the central configuration at maximum power conditions.

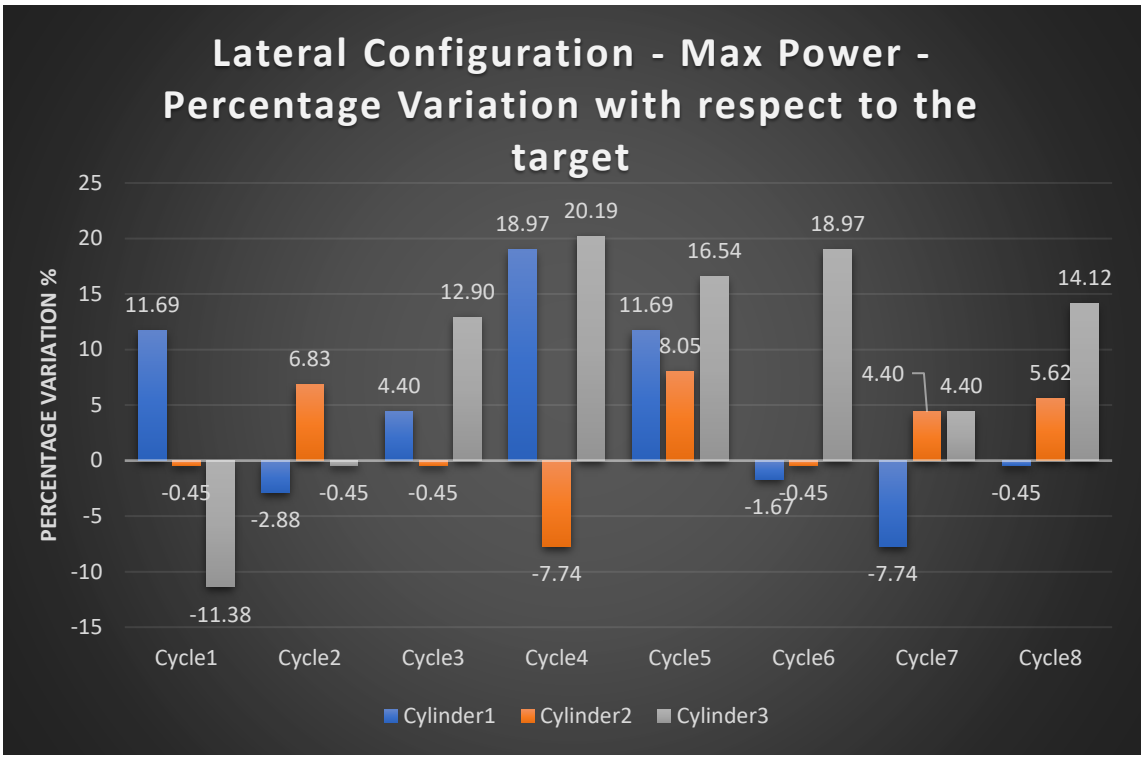


Diagram 38. Mass Flow Rate percentage variation with respect to the target for the lateral configuration.

The percentage variation is higher with respect to the condition of maximum torque: this is a further confirmation of how the transient model, in this moment, can not give attenable predictions on the performance of air mass flow balancing between the cylinders.

5.7 Transient Model Final Considerations

The results obtained by the transient simulation on the ATM-AR-F3R intake manifold does not permit, in this moment, to use the transient model to make evaluations on the percentage variation of air that flows inside each cylinder of different intake manifold layout configurations.

Moreover, there are other reasons that produce the computed unbalance:

- Representation of the engine as a pressure pulsating source: at the outlet of the intake manifold it imposed a pressure wave coming from a validated 1D model. This boundary condition approach is not sufficient to reproduce the exact flow condition that occurs inside the intake manifold since the pressure waves coming from the 1D code are modeled not taking in account the real intake manifold geometry and the intake duct geometry, that strongly affects the pressure traces.

- Limitations of the RANS turbulence models: the RANS equations are Navier-Stokes equation where the independent variables are not instantaneous but averaged on a certain period of time. This approach is performed to reduce the computational time: a turbulent flow field is composed by different vortexes with different scales that transfer kinetic energy from the largest scales to the smaller ones, up to arriving to the Kolmogorov scale, where it has the smaller vortexes. The RANS approach does not solve each scale but uses a statistical approach where all the scales are modeled by means of the turbulence models, that provide the closure equations. Therefore, all the turbulent scales are modeled by an average turbulent value that represents the mean turbulence in a certain period of time. For some practical application, the only knowledge of the mean quantities is sufficient to correctly solve the problem: this is particularly true when the scales of the mean motion are much bigger than the ones of the turbulent fluctuations, and so when the time step over which it is performing the average is longer than the turbulence time scales. If it does not occur, it performs an average operation on an interval that does not contains all the turbulence scales since they are bigger than the interval itself. This condition can occur in a transient computational fluid dynamic simulation, where the time-step is reduced to respect the Courant-Friedrich-Lewis condition. For this reason, the assumption of turbulence modeling that lies behind the RANS approach produces, in a transient simulation, a bigger error with respect to what occurs in a steady-state simulation.

- Sensibility of the quantitative analysis to the time-step error: to perform a quantitative analysis, the percentage variation with respect to the target is computed by an integral average over the engine cycle. The integral average over the engine cycle is affected by the mass conservation error that can be present in some time interval in which the engine cycle is divided by the software. This produces a significant error in the integral average, and therefore the results, at this stage, are not reliable.

6 CONCLUSIONS AND FUTURE DEVELOPMENTS

The CFD analysis of the intake manifold for motorsport application points out how the steady-state model is, now, the best computational fluid dynamic model to assess the intake manifold performance in terms of pressure losses and velocity uniformity distribution. The reliability of the steady-state model has been validated by two different turbulence models. The K-Omega model has been indicated as the most effective one in perceiving pressure gradients and flow detachment from the walls. By means of the steady-state model it is possible to locate the best air flow entrance and evaluate the effects of different geometrical parameters on the pressure losses and on the velocity distribution at the engine head cross section.

The transient model is, potentially, a model that is capable to take in account the dynamic conditions of operation of an intake manifold for motorsport application, but it has some intrinsic limitations due to the discretization of a single component that is part of a complex system as the engine: it is difficult to correctly reproduce, by means of boundary conditions coming from a 1D model, the effects of the different engine components that affects the intake manifold performance. Moreover, to perform a quantitative evaluation of the balancing of air distribution between the different cylinders it has to adopt an integral average over the whole cycle, and this produces a significant numerical error on the results. For these reasons, in the future developments of this project, realistic boundary conditions should be applied: a 1D model produces the same pressure trace for all the cylinder, but the pressure waves are strongly affected by the geometry, that is not taken in account by the 1D model. The measurement of the pressure in the outlet section of the intake manifold would introduce realistic boundary conditions. Additionally, it should consider gathering a larger amount of experimental data concerning the unbalance between the cylinders of different engines, in order to have the possibility to develop a different key lecture in such a way to have the possibility to create a correlation between the numerical error present in the simulation and the percentage variation measured at the engine test bench.

Additionally, the RANS approach used up to now is not perfectly indicated for a fully transient simulation, and therefore a deepening on large eddy simulation (LES) field can be taken in account, always considering the trade-off between the increase in accuracy and the increase of computational time.

REFERENCES

[1]. Giacomo Augusto Pignone, Ugo Romolo Vercelli, “Motori ad alta potenza Specifica”, GIORGIO NADA EDITORE, June 2003;

[2]. www.OpenFOAM.com;

[3]. Menter, F.R. “Two-Equations Eddy-Viscosity Turbulence Models for Engineering Applications”, AIAA Journal, August 1994;