



**Politecnico  
di Torino**

**Politecnico di Torino**

Master Degree in Aerospace Engineering  
A.a. 2023/2024

# **Multi-physics simulation for a pressure relief valve**

Supervisors:

Prof. Ferrero Andrea  
Prof. Rundo Massimo  
Ing. Catelani Daniele  
Ing. Cipolato Elia

Candidate:

Bosco Giada



## **Abstract**

This thesis aims to investigate the interaction between the multibody dynamics and fluid dynamics of a pressure relief valve for hydraulic applications.

The performance of a hydraulic valve is significantly influenced by flow forces generated due to changes in fluid momentum within the valve. These flow forces consist of three components: a steady-state factor, directly proportional to the flow rate and fluid velocity, and two unsteady factors, dependent on the flow rate derivative and spool acceleration, respectively. The first contribution, which is the most influential, consistently tends to decrease the flow area. It has been proven that, under a constant pressure drop across the valve, the steady-state force is directly proportional to the spool displacement, resembling the behavior of a virtual spring.

In the present work, the behavior of the valve was investigated by means of the co-simulation tool developed by MSC Software, now incorporated into Hexagon MI (Manufacturing Intelligence) - Design & Engineering, which allows to couple multibody simulations with fluid dynamic simulations. The main goal of this study is to validate the model through co-simulation and propose variations for further exploration. The outcomes will provide useful data to deliver practical insights in the understanding of system-fluid dynamics interactions.



# Acknowledgements

Ringrazio i miei due relatori: il Prof. Andrea Ferrero, per la sua precisa e costante supervisione, e il Prof. Massimo Rundo, per avermi permesso di ispirarmi ai suoi precedenti lavori per svolgere il mio e per i suoi suggerimenti fondamentali.

Ringrazio Hexagon per la possibilità che mi ha concesso di svolgere il mio lavoro di tesi presso la loro azienda, per le licenze e l'hardware messi a disposizione e per il tempo dedicato al training. In particolare, ringrazio i miei due tutor: l'Ing. Daniele Catelani, per aver mostrato continuo interesse per il mio lavoro e per la sua inestimabile esperienza che ha messo a mia disposizione; l'Ing. Elia Cipolato, per avermi guidata per tutto il lavoro, per la sua presenza e per il suo sostegno. Ringrazio inoltre tutto l'ufficio Hexagon di Torino, per avermi accolta nel migliore dei modi; in particolare, ringrazio l'Ing. Angelo Casolo e l'Ing. Jordi Urcola Peya, per la loro disponibilità e per i loro preziosi consigli.

In ultimo, ma non per importanza, grazie ai miei amici. Grazie Luca, per avermi sempre spinto ad inseguire i miei sogni. Grazie alla mia famiglia, per avermi sempre sostenuta e per essere stata l'appoggio dei momenti difficili.

# Contents

<b>List of Figures</b>	V
<b>List of Tables</b>	VIII
<b>1 Introduction</b>	1
1.1 Safety Valves . . . . .	2
1.2 Safety Relief Valves . . . . .	3
1.3 Vacuum Relief Valves . . . . .	4
1.4 Liquid Relief Valve . . . . .	5
1.5 Force/Lift Diagrams . . . . .	6
1.6 Present work . . . . .	8
<b>2 Fundamentals of multibody simulation</b>	9
2.1 Introduction . . . . .	9
2.2 Adams Generalized Coordinates . . . . .	10
2.3 Joints in Adams . . . . .	11
2.4 Dynamic analysis . . . . .	12
2.5 Numerical Solution and GSTIFF integrator . . . . .	13
2.5.1 GSTIFF I3 Integrator . . . . .	14
2.6 GForces . . . . .	14
2.7 Multi-component force . . . . .	15
2.8 Bistop function . . . . .	16
2.9 Translational Spring Damper . . . . .	17
<b>3 Fundamentals of fluid-dynamic simulation</b>	19
3.1 Introduction . . . . .	19
3.2 Conservation equations . . . . .	19
3.3 Turbulence modeling . . . . .	22
3.3.1 RANS . . . . .	22
3.4 Boussinesq closure model . . . . .	23
3.4.1 Standard $k - \varepsilon$ model . . . . .	24
3.4.2 SST $k - \omega$ model . . . . .	25
3.5 Numerical methods for fluid dynamics . . . . .	26
3.5.1 Finite volume method . . . . .	27
3.5.2 The matrix solver . . . . .	30
3.6 Pressure-based solver . . . . .	31

<b>4</b>	<b>Fundamentals of cosimulation</b>	<b>32</b>
<b>5</b>	<b>Adams model</b>	<b>35</b>
5.1	Joints . . . . .	36
5.2	Translational Spring Damper definition . . . . .	38
5.3	Setting of Shutter end position . . . . .	39
5.3.1	Gforce and Dynamic simulation . . . . .	40
<b>6</b>	<b>scFLOW model</b>	<b>42</b>
6.1	Model set up and definition of the fluid domain . . . . .	42
6.2	Overset mesh . . . . .	43
6.3	Analysis model . . . . .	45
6.3.1	Part material . . . . .	45
6.3.2	Regions . . . . .	45
6.4	Conditions . . . . .	47
6.4.1	Flow boundary conditions . . . . .	47
6.4.2	Wall boundary conditions . . . . .	48
6.4.3	Moving elements . . . . .	49
6.5	Octree and Mesh . . . . .	50
6.5.1	Background octree and mesh . . . . .	52
6.5.2	Overset octree and mesh . . . . .	54
<b>7</b>	<b>Co-simulation model</b>	<b>57</b>
<b>8</b>	<b>Grid independence analysis</b>	<b>59</b>
8.1	Convergence order evaluation . . . . .	68
8.1.1	Results . . . . .	69
8.1.2	Model verification . . . . .	70
<b>9</b>	<b>Results</b>	<b>72</b>
9.1	First analysis . . . . .	72
9.2	More accurate analysis . . . . .	75
9.3	Effect of damping . . . . .	86
9.4	Effect of initial lift . . . . .	88
9.5	Effect of preload . . . . .	89
9.6	Effect of outlet pressure . . . . .	92
<b>10</b>	<b>Conclusion</b>	<b>96</b>

# List of Figures

1.1	A representation of the vessel-valve system using mechanical modeling, incorporating state variables and parameters [8]	2
1.2	Safety valve [20]	3
1.3	Safety relief valve with a bellows for balance [20]	4
1.4	Breather valve [20]	5
1.5	Relief Valve [20]	5
1.6	Proportional Relief Valve [20]	6
1.7	Influence of pressure loss and back pressure on Force/Lift diagram	7
1.8	Mode of Blowdown Adjustment [20]	7
2.1	Example of the action and reaction force movement [21]	15
2.2	Example of the Bistop function [21]	16
2.3	Example of a Translational Spring Damper [21]	18
3.1	Control volume	20
3.2	Data and element [22]	27
3.3	Data and element in 2D view [22]	28
3.4	Data involved in calculating the numerical flux on the red surface [22]	29
4.1	Representation of exchange of physical quantities [23]	32
4.2	Representation of co-simulation workflow [23]	33
5.1	Valve main parts	35
5.2	Fixed joint for Shutter Support	36
5.3	Fixed joint for Spring Upper Support	37
5.4	Fixed joint for Spring Lower Support and Shutter	37
5.5	Translational joint for Shutter	38
5.6	Spring parameters	38
5.7	Bistop function parameters	39
5.8	Gforce applied to Shutter center of gravity	40
5.9	Performed dynamic simulation parameters	41
6.1	Unification of parts Spring Lower Support and Shutter	42
6.2	Fluid domain	43
6.3	Fluid domain	43
6.4	Setting for overset mesh	44
6.5	Fluid domain for component mesh	44
6.6	Background and component meshing units	45



6.7	Defined regions . . . . .	46
6.8	Numerical region . . . . .	47
6.9	Inlet and outlet conditions . . . . .	48
6.10	Setting of moving condition . . . . .	49
6.11	Orientation of the reference system . . . . .	49
6.12	Quadratic interpolation . . . . .	50
6.13	Refinement levels for octree generation [22] . . . . .	50
6.14	Prism growing on wall [22] . . . . .	52
6.15	Background octree definition . . . . .	53
6.16	Background mesh definition . . . . .	53
6.17	Background mesh details . . . . .	54
6.18	Background mesh . . . . .	54
6.19	Overset octree definition . . . . .	55
6.20	Overset mesh definition . . . . .	55
6.21	Overset mesh details . . . . .	56
6.22	Overset mesh . . . . .	56
7.1	Adams and scFLOW configurations . . . . .	57
7.2	Coupling pairs of Adams GForce marker and scFLOW Moving condition . . . . .	58
7.3	Cosimulation settings . . . . .	58
8.1	Pressure fields . . . . .	62
8.2	Velocity fields . . . . .	63
8.3	Grid independence study with $k - \varepsilon$ model . . . . .	64
8.4	Pressure fields . . . . .	65
8.5	Velocity fields . . . . .	67
8.6	Grid independence study with $k - \omega$ model . . . . .	67
9.1	Cartridge assembly modifications . . . . .	73
9.2	Shutter displacement and Shutter speed in x, z and y-direction . . . . .	73
9.3	Spring deformation and Spring deformation velocity . . . . .	74
9.4	Spring force . . . . .	74
9.5	Shutter Limit Switch force . . . . .	74
9.6	Gforce in y-direction and Gforces in x and z directions . . . . .	75
9.7	Cartridge assembly modifications . . . . .	76
9.8	Shutter displacement in y-direction . . . . .	76
9.9	Shutter displacement in x and z-direction . . . . .	77
9.10	Shutter speed in y-direction . . . . .	77
9.11	Shutter speed in x and z-direction . . . . .	77
9.12	Spring deformation . . . . .	78
9.13	Spring deformation velocity . . . . .	78
9.14	Spring force . . . . .	78
9.15	spring displacement and the spring force linear trend with respect to the displacement of the shutter . . . . .	79
9.16	Lower Spring Support displacement in y, x and z-directions . . . . .	79
9.17	Lower Spring Support speed in y, x and z-directions . . . . .	80
9.18	Shutter Limit Switch force . . . . .	80
9.19	Translational joint Force in x and y-directions . . . . .	80

9.20	Translational joint Force in z-direction and magnitude . . . . .	81
9.21	Translational joint Torque in x and y-directions . . . . .	81
9.22	Translational joint Torque in z-direction and magnitude . . . . .	81
9.23	Gforce in x and y-directions . . . . .	82
9.24	Gforce in z-direction and magnitude . . . . .	82
9.25	Gforce, torque components in x and y-directions . . . . .	82
9.26	Gforce, torque components in z-direction and magnitude . . . . .	83
9.27	Pressure field at end of simulation . . . . .	83
9.28	Velocity field at end of simulation . . . . .	84
9.29	Velocity vector field at end of simulation . . . . .	84
9.30	Pressure field in steady-state simulation . . . . .	85
9.31	Velocity field in steady-state simulation . . . . .	85
9.32	Velocity vector field in steady-state simulation . . . . .	86
9.33	Shutter displacement and Shutter speed in x, z and y-direction . . . . .	87
9.34	Spring deformation and Spring deformation velocity . . . . .	87
9.35	Spring force . . . . .	87
9.36	Shutter Limit Switch force . . . . .	88
9.37	Shutter displacement and Shutter speed in y-direction . . . . .	88
9.38	Shutter displacement and Shutter speed in y-direction . . . . .	89
9.39	Gforce in y-direction and Gforces in x and z-directions . . . . .	89
9.40	Pressure field at maximum shutter lift . . . . .	90
9.41	Velocity field at maximum shutter lift . . . . .	90
9.42	Shutter displacement and Shutter speed in y-direction . . . . .	91
9.43	Gforce in y-direction and Gforces in x and z-directions . . . . .	91
9.44	Pressure field at maximum shutter lift . . . . .	91
9.45	Velocity field at maximum shutter lift . . . . .	92
9.46	Shutter displacement and Shutter speed in y-direction . . . . .	92
9.47	Shutter displacement in x and z-direction . . . . .	93
9.48	Shutter speed in y-direction . . . . .	93
9.49	Shutter Limit Switch force . . . . .	93
9.50	Pressure field at end of simulation . . . . .	94
9.51	Velocity field at end of simulation . . . . .	94
9.52	Velocity vector field at end of simulation . . . . .	95

# List of Tables

3.1	Standard $k - \varepsilon$ model constant parameters values . . . . .	25
5.1	Allowed DOF for used joints . . . . .	36
5.2	Main geometric parameters of the valve . . . . .	39
6.1	Fluid property . . . . .	45
6.2	Boundary conditions . . . . .	48
6.3	$y^+$ values with reference to used models . . . . .	51
8.1	Coarse, mid, fine and extra fine grids . . . . .	60
8.2	Discretization of the boundary layer . . . . .	60
8.3	Convergence order evaluation for $k - \varepsilon$ model . . . . .	69
8.4	Convergence order evaluation for $k - \omega$ model . . . . .	70
9.1	First analysis configuration . . . . .	73



# Chapter 1

## Introduction

Pressure relief devices provide the essential function of purging a system from overpressure conditions. Particularly, a Self-Actuated Pressure Relief Valve (SRV) is designed to protect life and equipment through controlled fluid discharge at a predetermined pressure [8]. It acts as the final control device to prevent accidents or explosions caused by overpressure. SRVs must comply with international codes and close at a predetermined pressure when the system pressure returns to a safe level.

SRVs must be designed with materials compatible with a wide range of process fluids, including corrosive and toxic media. They should operate smoothly on various fluids and fluid phases. These design parameters result in a diverse range of SRV products in the market, all adhering to internationally recognized codes [17].

Pressure relief valves fall into two major groups: direct-loaded pressure relief valves and piloted pressure relief valves. Direct-acting valves respond directly to system fluid pressure, while pilot-operated valves use a pilot to control the main valve based on system pressure. Direct-loaded valves may have an auxiliary actuator for lift assistance or supplementary closing force. Pilot-operated valves can have a pilot directly controlling the main valve or indirectly through discharged fluid.

Pressure relief valves are critical safety elements in high-pressure hydraulic and pneumatic systems. A direct spring-operated relief valve consists of a precompressed helical spring and a valve disc (Fig. 1.1). During normal operation, the valve disc presses against the valve seat, closing the pressurized fluid space. The set pressure, adjustable with spring precompression, determines the valve opening. External viscous damping, optional but restricted by some standards, aims to prevent valve vibrations. Vibrations, indicated by harsh noise, can lead to mechanical damage or insufficient discharging, emphasizing the importance of proper damping considerations [12].

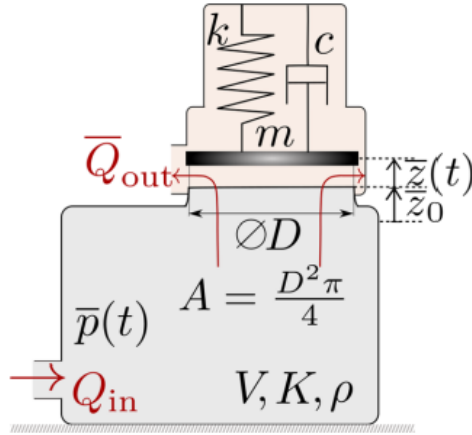


Figure 1.1: A representation of the vessel-valve system using mechanical modeling, incorporating state variables and parameters [8]

Early pressure relief valves were initially loaded with a weight, leading to minimal disc lift within allowable overpressure. Charles Ritchie's 1848 innovation enhanced lift using a peripheral flow deflector, creating an annular chamber for sudden valve opening. William Naylor's 1863 design improved lift by turning discharged fluid through 180°. Modern designs combine Ritchie and Naylor principles, incorporating a lip around the disc to form an annular chamber with a secondary orifice. Liquid relief valves developed based on this design open fully within a 10% overpressure.

Early valves used a weight, due to challenges in producing suitable springs to regulate the disc's lifting force. They were later replaced by springs due to practicality and controllability. Consequently, the majority of direct-loaded pressure relief valves now employ spring loading, with exceptions for low-pressure applications as per industry standards. The following sections outline a standard range of direct-loaded pressure relief valves available in the industry [3] [10].

## 1.1 Safety Valves

Safety valves, exemplified in Figure 1.2, serve primarily for relieving steam in industrial boiler plants and other steam systems. Their design focuses on safeguarding the spring from excessive temperature increases that might lead to spring setting drift and potential relaxation over time. To ensure this protection, safety valves often feature an open bonnet, allowing steam entering the bonnet to directly escape into the surrounding atmosphere [6] [17].

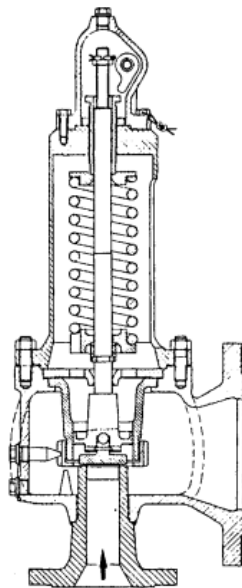


Figure 1.2: Safety valve [20]

The disc of safety valves typically moves within a guided sleeve, equipped with a threaded ring for altering the deflection direction of the escaping fluid. This, in turn, adjusts the reactive force on the underside of the disc, allowing for sensitive blowdown adjustment. Adjusting the guide ring affects blowdown length, with lowering extending it and raising shortening it.

The nozzle ring, located below the nozzle seat, regulates the difference between set pressure and popping pressure. Adjusting the nozzle ring influences valve popping timing and blowdown duration.

Safety valves with open bonnets find partial balance through the stem guide, permitting operation against a built-up back pressure of around 20%, subject to manufacturer consultation. They are unsuitable for superimposed back pressure due to potential leakage around the stem to the bonnet, and thus, manifold use is discouraged [20].

## 1.2 Safety Relief Valves

Safety relief valves are versatile in gas or liquid services within the process industry. They prevent fluid leakage with vented or bellows-sealed bonnets. In gas service, valves pop open, while in liquid service, they modulate open, reaching full open position at 25% overpressure (or 10% in newer designs).

Two types exist: conventional and balanced safety relief valves. Conventional ones have a closed bonnet vented to the outlet, suitable for short pipelines limiting back pressure to 10%. Superimposed back pressure affects the set pressure. Some can handle constant back pressure up to 50%. Open bonnet conversion allows up to 20% built-up back pressure.

Balanced safety relief valves (Figure 1.3) minimize back pressure limitations with balanced bellows between the disc and vented bonnet. They maintain rated capacity up to 40% back pressure, closing gradually beyond. Blowdown adjustment, critical for valve stability, is achieved through a nozzle ring or external screw [20].

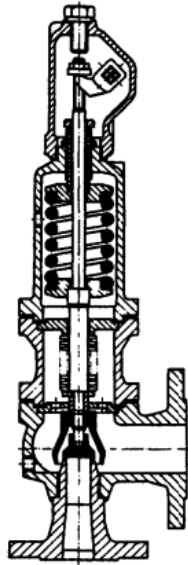


Figure 1.3: Safety relief valve with a bellows for balance [20]

### 1.3 Vacuum Relief Valves

A vacuum relief valve operates through a mechanism that permits the ingress of air into a system to mitigate the buildup of excessive vacuum pressure. When the internal pressure within the system diminishes beyond a predetermined threshold, the valve activates, facilitating the intake of air and thereby equalizing the pressure. This action serves to safeguard the integrity of the system by preventing potential damage resulting from the imposition of excessive vacuum pressures.

There are different configurations of vacuum relief valves, either individually or in conjunction with positive over-pressure relief.

In Figure 1.4, a breather valve integrates a direct-loaded vacuum relief valve and an overpressure relief valve. Unlike other vacuum relief valves, it employs a disc that swings open on a point contact hinge. Soft diaphragm seals allow the valve to reseal close to the set pressure, making it suitable for low-pressure storage tanks [20].



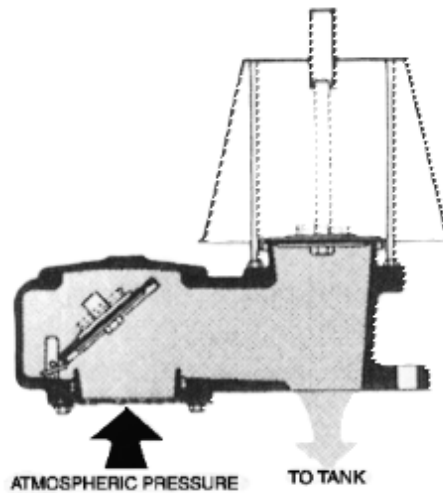


Figure 1.4: Breather valve [20]

## 1.4 Liquid Relief Valve

Typically, liquid relief valves only slightly modify the disc's geometry compared to traditional safety relief valves to achieve specific performance in liquid service.

The liquid relief valve in Figure 1.5 opens fully under a 10% overpressure, ensuring stable operation across various conditions. Blowdown adjustment is possible through the nozzle ring.

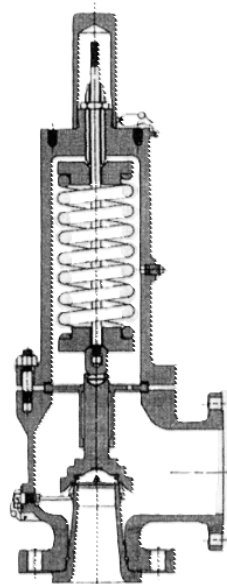


Figure 1.5: Relief Valve [20]

Figure 1.6 displays a liquid relief valve with a linear proportional opening characteristic, suitable for wide load range operations. The flared nozzle outlet minimizes disc movements in response to flow rate changes. Tests confirm effective internal friction to prevent valve oscillations, even with rapidly accelerating mass flow. Figure 5-16 illustrates the opening and closing characteristics of this valve [6] [20].

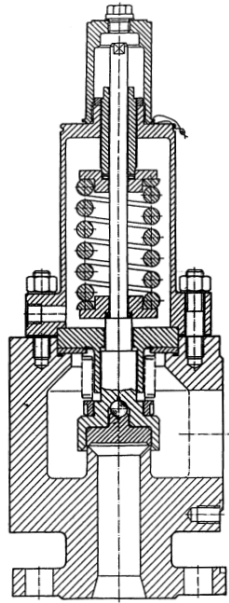


Figure 1.6: Proportional Relief Valve [20]

Incompressible fluids, like liquids, impact valve behavior differently. Their nearly incompressible nature and high density cause significant pressure changes with small alterations in inlet flow velocity. Liquid relief valves, more susceptible to chatter than gas valves, can be equipped with friction dampers to eliminate valve chatter<sup>1</sup> [20].

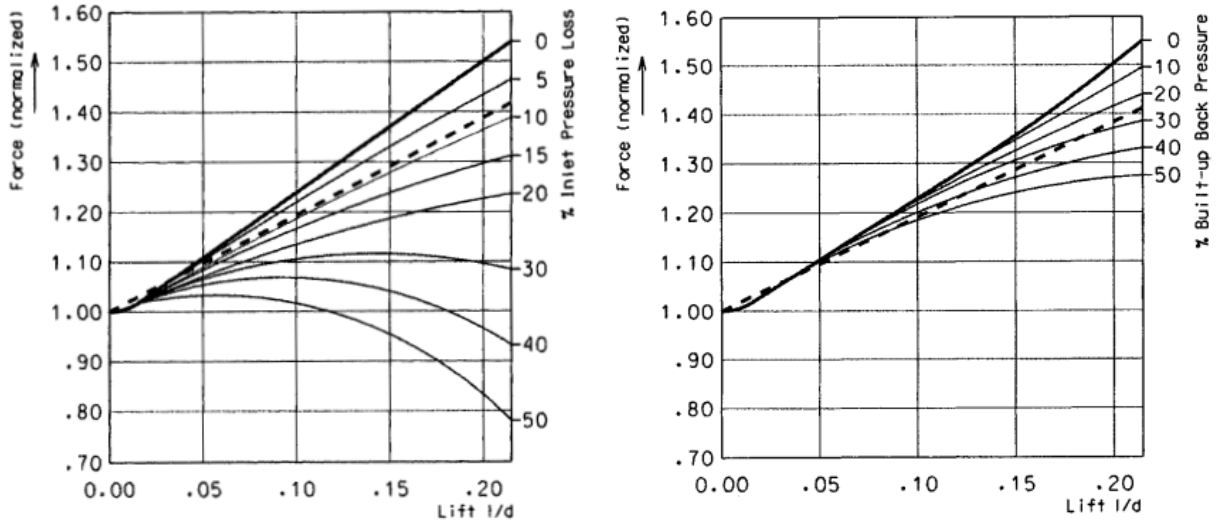
## 1.5 Force/Lift Diagrams

In order to understand how pressure relief valves behave in operation, diagrams depicting force and lift, as illustrated in Figures 5-29 to 5-33, serve as valuable tools. These diagrams are instrumental in analyzing the impact of factors like inlet pipe pressure loss and back pressure on the lifting force of the valve, as well as in examining the blowdown adjustment using specific devices.

The diagrams feature a dashed line symbolizing the closing force applied by the spring on the valve disc, while S-shaped curves represent the net opening forces from the fluid acting on the disc. These curves, determined experimentally under constant inlet pressure, vary in shape depending on the valve design.

---

<sup>1</sup>Irregular, back-and-forth movement of the components within the pressure relief valve, involving direct contact between the disc and the nozzle.



(a) Effect of pressure loss [20] (b) Effect of back pressure [20]

Figure 1.7: Influence of pressure loss and back pressure on Force/Lift diagram

The force/lift diagram (Figure 1.7a) illustrates the diminishing effect of increasing inlet pipe pressure loss on lifting force, considering zero back pressure. This specific valve type restricts lift to  $l/d_0 = 0.22$ . The manufacturer has chosen a spring characteristic resulting in full valve opening with a 3% inlet pressure loss.

Examining the force/lift diagram (Figure 1.7b) reveals the diminishing effect of built-up back pressure on the lifting force for an unbalanced conventional pressure relief valve under zero inlet pipe pressure loss conditions. If the built-up back pressure at 3% inlet pipe pressure loss is sufficiently high, the valve may lose its ability to fully open in one stroke.

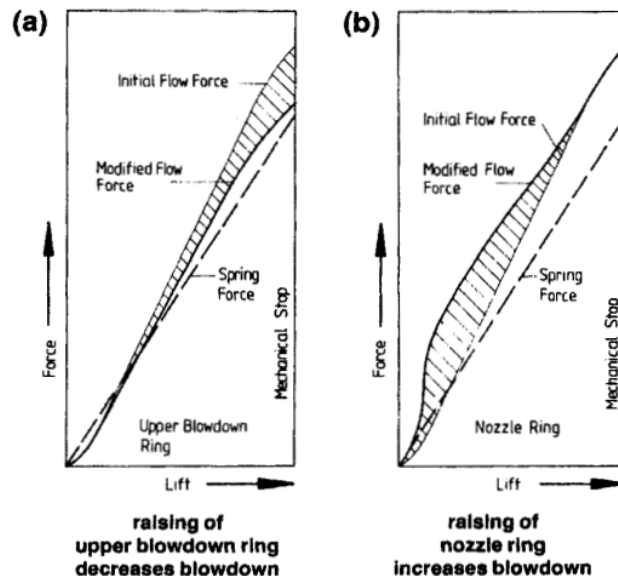


Figure 1.8: Mode of Blowdown Adjustment [20]

The force/lift diagrams in Figures 1.8 (a) and (b) demonstrate blowdown adjustment modes for different pressure relief valve types. (a) applies to valves with a guide sleeve and a screwed ring, adjusting blowdown by altering the sleeve length; (b) pertains to valves with a screwed ring on

the nozzle, adjusting blowdown by reducing the width of the secondary orifice.

Different valves may exhibit varied resistance, and manufacturers offer methods like adjusting spring characteristics or selecting larger orifices to enhance back pressure tolerance. This approach involves balancing shorter decaying lift force curves with larger inlet and discharge piping to reduce pressure loss [6] [20].

## 1.6 Present work

In the present thesis work, we aim to reproduce the behavior of the opening transient of a Pressure Relief Valve, until an equilibrium position is reached. The motion of the valve is analyzed by coupling two software programs, *scFLOW* for fluid dynamics and *Adams* for multibody dynamics, using *MSC Cosim* co-simulation software. Comparison is then made with the results obtained in [1], to validate the correctness of the analysis performed and the margins for error and improvement.

# Chapter 2

# Fundamentals of multibody simulation

## 2.1 Introduction

The simulation of Multibody systems involves analyzing the motion of mechanical systems under the influence of external forces. The multibody systems approach involves utilizing a finite set of elements, including rigid bodies and flexible bodies, springs, dampers, joints, supports and actuators. The accepted assumptions are outlined as follows:

A multibody system comprises rigid bodies and flexible bodies, ideal joints and elastic connections.

Ideal joints include rigid joints, joints with completely specified motion (rheonomic constraint), and vanishing joints (free motion). Elastic connections include bushings, beams, fields, general forces and more advanced elements.

The topology of the multibody system can be arbitrary, allowing for chains, trees, and closed loops.

Joints and actuators are consolidated in open libraries of standard elements.

Subsystems can be added to existing components of the multibody system.

*Adams*, or Automatic Dynamic Analysis of Mechanical Systems, is a sophisticated software system designed to assist analysts in conducting three-dimensional analyses of mechanical systems. It covers aspects such as kinematics, statics, quasi-statics, and dynamics and is versatile enough to handle systems with various interconnected rigid or flexible bodies [18]. The software allows for extensive rotational and translational movements, accommodating different internal or external forces and predefined motions. Notably, it is adaptable to diverse topological configurations, treating chain, tree, cluster, closed-loop, and multiple closed-loop setups uniformly. *Adams* efficiently identifies and eliminates redundant constraints.

The program's inputs include part geometry, mass properties, reference frames, body types, compliance descriptions, constraints, actuator and sensor models, restraints, connectors, control laws, and graphic entities. Outputs provide time-dependent positions, velocities, accelerations, forces, and user-defined variable values in various formats like tables, plots, static configurations, and dynamic animations. The *Adams* software system comprises ten integrated programs, with

the *Adams* program serving as the primary engine for general kinematic, static, and dynamic analyses.

Tanks to its flexibility, *Adams* is suitable for various analyses and problems boasting, as in [25]:

- Rigid and flexible multibody systems;
- Sensitivity analysis;
- Vibration analysis;
- Vehicle design & testing;
- Coupled control/mechanical system analysis;
- Kinematics and kinetics;
- Contact and friction;
- Loads and displacement;
- Durability and life-cycle analysis;
- Fracture or fatigue calculations;
- Kinetic, static, and dissipative energy distribution;
- Vehicular cornering, steering, quasi-static, and straight-line analysis;
- Control system analysis.

In the following sections, we will discuss the main equations and methods used by the software to perform a multibody simulation. All mathematical expressions and formulations provided here are based on or guided by the content found in the cited reference [13].

It should be noted that what follows is only a small part of the capabilities of the *Adams* software. This choice was made in order not to excessively make the discussion heavy while at the same time providing the details necessary for understanding the model described later on.

## 2.2 Adams Generalized Coordinates

In *Adams*, the way we express the position and orientation of rigid bodies involves specific mathematical principles.

The location of a rigid body is defined using three Cartesian coordinates:  $x$ ,  $y$ , and  $z$ . These coordinates represent the body's position in three-dimensional space.

The orientation of a rigid body is determined by a trio of Euler angles: yaw ( $\psi$ ), pitch ( $\theta$ ), and roll ( $\phi$ ). These angles describe the body's orientation through a 3-1-3 rotation sequence.

These three angles and the previous Cartesian coordinates are saved in two different arrays, not vectors, using the following structure:

$$p = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad \varepsilon = \begin{bmatrix} \psi \\ \phi \\ \theta \end{bmatrix}$$

The combination of position and orientation is captured in a set of generalized coordinates for a specific rigid body. These coordinates serve to uniquely define the configuration of the body:

$$q_i = \begin{bmatrix} p_i \\ \varepsilon_i \end{bmatrix}$$

The body's longitudinal velocity ( $u$ ) and angular velocity ( $\omega$ ) are calculated based on the time derivatives of the chosen generalized coordinates. A matrix "B" relates the time derivative of Euler angles to the body's angular velocity. This matrix establishes a link between the body's intrinsic properties and the selected generalized coordinates.

$$u = \dot{p}$$

$$\bar{\omega} = B\dot{\varepsilon} \equiv B\zeta$$

$$B = \begin{bmatrix} \sin\phi\sin\theta & 0 & \cos\phi \\ \cos\phi\sin\theta & 0 & -\sin\phi \\ \cos\theta & 1 & 0 \end{bmatrix}$$

The link between angular velocity ( $\bar{\omega}$ ) and the body orientation matrix (A) time derivative is expressed through a skew-symmetric operator.

$$\dot{A} = A\tilde{\bar{\omega}}$$

In a mechanical system comprising  $n_b$  bodies, the array:

$$q = [q_1^T, q_2^T, \dots, q_{n_b}^T]^T = [q_1, q_2, \dots, q_n]^T \quad (2.1)$$

where  $n = 6 \cdot n_b$ , provides a comprehensive representation of the positions and orientations of all bodies within the system at a specific time.

In summary, *Adams* employs a combination of Cartesian coordinates and Euler angles to precisely define both the position and orientation of rigid bodies in a mechanical system, enabling detailed simulation and analysis.

## 2.3 Joints in Adams

In *Adams* simulation environment, joints are considered limitations influencing specific coordinates within the system ( $q_1$  through  $q_n$  of Eq. (2.1)). Mathematically, a joint constraint is represented by:

$$\Phi(q) = 0 \quad (2.2)$$

where  $\Phi$  contains all constraints imposed by the joints in the model:

$$\Phi(q) = [\Phi_1^T(q), \Phi_2^T(q), \dots, \Phi_{n_j}^T(q)]^T = [\Phi_1(q), \Phi_2(q), \dots, \Phi_m(q)]^T \quad (2.3)$$

with  $n_j$  the number of joints, and  $m$  is the total number of constraints. For instance, a revolute joint between two bodies introduces a set of constraints (as in Eq. (2.2)) to permit a specific degree of freedom between the connected bodies.

The kinematic constraint equations for position, velocity, and acceleration are expressed by taking one-time and two-time derivatives of Eq. (2.3):

$$\Phi_q \dot{q} = 0 \quad (2.4)$$

$$\Phi_q \ddot{q} = -(\Phi_{qq})_q \dot{q} \equiv \tau \quad (2.5)$$

Equations Eq. (2.4) to Eq. (2.5) ensure that the generalized coordinates, along with their derivatives, satisfy the necessary conditions for the coherent evolution of the mechanical system, guaranteeing that the constraints imposed by joints are adhered to throughout the system's motion.

## 2.4 Dynamic analysis

The formulation of rigid body equations of motion involves several quantities:

- $M$ : Generalized mass matrix.
- $J$ : Generalized inertia matrix expressed about the principal local reference frame.
- $K$ : Kinetic energy, defined as a combination of translational and angular kinetic energy

$$K = \frac{1}{2}u^T M u + \frac{1}{2}\bar{\omega}^T \bar{J} \bar{\omega}$$

- $\lambda$ : Array of Lagrange multipliers representing the number of constraint equations induced by joints.
- $F(q, \dot{q}, t) = \begin{bmatrix} f \\ \bar{n} \end{bmatrix}$ : Vector of applied forces.
- $Q(q, \dot{q}, t) = \begin{bmatrix} (\Pi^P)^T f \\ (\Pi^R)^T \bar{n} \end{bmatrix}$ : Generalized force acting on the body, obtained by projecting applied force  $F$  onto the generalized coordinates.

Considering  $P_v$  as the velocity at the point  $P$  where the external force  $F$  is applied, the calculation of projection operators is performed as follows:

$$\Pi^P = \frac{\partial v^P}{\partial u}$$

$$\Pi^R = \frac{\partial \bar{\omega}}{\partial \zeta}$$

In addition, we can define the angular momenta as:

$$\Gamma = \frac{\partial K}{\partial \zeta} = B^T \bar{J} B \zeta$$

The equations of motion, derived through Lagrange's formulation, result in second-order differential equations. The choice of generalized coordinates in *Adams* influences these equations, ensuring the representation of the mechanical system.

The numerical solution for dynamic analysis involves solving a set of 15 equations, including kinetic and kinematic differential equations, along with the constraints induced by joints, as stated:

$$M\dot{u} + \Phi_p^T \lambda - (\Pi^P)^T f = 0 \quad (2.6)$$

$$\Gamma - B^T \bar{J} B \zeta = 0 \quad (2.7)$$

$$\dot{\Gamma} - \frac{\partial K}{\partial \varepsilon} + \Phi_\varepsilon^T \lambda - (\Pi^R)^T \bar{n} = 0 \quad (2.8)$$

$$\dot{p} - u = 0 \quad (2.9)$$

$$\dot{\varepsilon} - \zeta = 0 \quad (2.10)$$



The solution provides information about the translational and angular momenta, satisfying both the dynamic equations and kinematic constraints.

The numerical solution process utilizes an implicit integration formula, such as the backward Euler formula, to discretize the first-order time derivatives. The resulting system of nonlinear algebraic equations is solved using a Newton-Raphson iterative algorithm. The complexity of solving this system arises due to the inclusion of kinematic constraint equations, making dynamic analysis in mechanical systems a challenging simulation.

*Adams* employs reliable methods for solving these equations, such as direct index 3 DAE<sup>1</sup> solvers or more refined algorithms that reduce the problem to an analytically but yet numerically different index 2 DAE problem. The solution process involves iterative steps, employing Newton-Raphson method, and the solver may adjust the step size to achieve convergence. Refactorization may be required during long simulations to address issues of singularity in the Jacobian matrix. It's important to note that the presented backward Euler formula is conceptually representative, and *Adams* typically uses higher-order integrators for improved performance based on the characteristics of the problem being solved.

## 2.5 Numerical Solution and GSTIFF integrator

Three distinct formulations of Differential Algebraic Equation (DAE) systems are implemented in *Adams*: Index 3 (I3), Stabilized Index 2 (SI2), and Stabilized Index 1 (SI1). Each formulation involves different equations that differentiate in terms of their optimization equations calculated for each state of the dynamic system. While the I3 formulation offers less accurate evaluations of system velocities and accelerations compared to SI2 and SI1, it stands out for its lower computational demand.

Geometrical nonlinearities involve phenomena like large deformations and strain, while physical nonlinearities include material characteristics and contact effects. Nonlinear problems, involving both geometric and physical nonlinearities, represent a significant challenge due to their diverse nature. Achieving convergence, minimal computational times, and accurate solutions depends heavily on the proper configuration of mathematical models and solvers, which are essential components in addressing these issues [9].

Simulating nonlinear dynamic systems demands careful consideration of integrator selection and settings to avoid divergence, computational instability, or inaccuracies. Stiff differential equations, particularly challenging to integrate stably, require specialized integrators termed stiffly stable integrators. *Adams* offers three such integrators — GSTIFF, WSTIFF, and HASTIFF — all robust, multi-step<sup>2</sup>, implicit integrators featuring variable-order and step size. The distinction

---

<sup>1</sup>Differential-Algebraic Equations. A DAE is characterized by an associated index, and the general principle is that as the index increases, the numerical solution of the DAE becomes more difficult.

<sup>2</sup>Multistep methods enhance efficiency by utilizing and considering data from previous steps rather than only referring to the current state.

among these lies primarily in their predictor<sup>3</sup> methods: GSTIFF employs a Nordsiek<sup>4</sup> vector. The correction step of the STIFF integrators adopts the implicit formulation of the Backward Differentiation Formula (BDF), represented as:

$$\bar{y}_{n+1} = \sum_{i=1}^k \alpha_i \bar{y}_{n-i+1} + h \beta_0 \dot{\bar{y}}_{n+1}$$

where  $\alpha$  and  $\beta$  are constants that vary based on the order of the corrector  $k$ .

### 2.5.1 GSTIFF I3 Integrator

The GSTIFF I3 integrator in *Adams* is acknowledged for its high-speed capabilities, ensuring efficient simulations while ensuring accuracy in system displacements [7].

Despite its strengths, potential errors in velocities and accelerations should be noticed. To address this, controlling the *Hmax* parameter is recommended, enabling the integrator to operate at a constant step size while maintaining a high order, preferably three or more.

The GSTIFF I3 integrator is particularly well-suited for numerically stiff models, commonly found in mechanical systems with a broad range of frequency content. It also proves beneficial for models featuring velocity inputs.

## 2.6 GForces

The GForce in *Adams* represents general forces with three force components and three moment components, all mutually orthogonal. These components are actually vacant, awaiting input from the co-simulation software. For each part engaged in the co-simulation, it is crucial to establish a corresponding GForce.

GForce format is:

"GFORCE/id\I = id, JFLOAT = id, RM = id"

- *I = id*: Defines the location where *Adams* Solver exerts forces and torques. It's crucial to ensure that the "I" marker is fixed and on a separate part from the "JFLOAT" marker. This ensures consistent force application at a fixed point on the part associated with the "I" marker.
- *JFLOAT = id*: Specifies the point where *Adams* Solver applies reaction forces and torques. *Adams* Solver adjusts the "JFLOAT" marker position to align with the "I" marker, allowing movement in the point of reaction force application relative to its part.
- *RM = id*: Designates the marker and, consequently, the coordinate system where force and torque components are defined.

---

<sup>3</sup>Numerical integration involves an initial evaluation, known as a predictor, to estimate the state of a system at a future time step based on its current state and the derivatives of the state variables.

<sup>4</sup>The Nordsieck method [16] [14], derived by stabilizing a Taylor series approach, is a reliable and efficient approach for integration of systems of ordinary differential equations. It utilizes current values of higher derivatives of a polynomial approximation of the solution adjusting interval sizes for specified accuracy, with minimal computation of derivatives per step.

By considering two markers, "I" and "JFLOAT," the creation of a GForce initiates an action on the part associated with marker "I" and an equal and opposite reaction on the part linked to marker "JFLOAT." The position of the JFLOAT marker is automatically adjusted by the software during system movement, ensuring a constant overlap with the I marker. As a result, the reaction force is exerted on the part to which the JFLOAT marker belongs, precisely at the position of the I marker at any given moment.

The vectors derived from the three force components and three moment components determine the direction of the action force and torque, respectively.

$$F_a = FX\hat{x}_{rm} + FY\hat{y}_{rm} + FZ\hat{z}_{rm}$$

$$T_a = TX\hat{x}_{rm} + TY\hat{y}_{rm} + TZ\hat{z}_{rm}$$

$F_a$  represents the applied translational force on the "I" marker.  $FX$ ,  $FY$ , and  $FZ$  are functions defined by the user for the magnitudes of force components along the x, y, and z axes, respectively.

$T_a$  represents the applied rotational force on the I marker.  $TX$ ,  $TY$ , and  $TZ$  are user-defined functions for the magnitudes of the rotational force components along the x, y, and z axes, respectively.

$\hat{x}_{rm}$ ,  $\hat{y}_{rm}$  and  $\hat{z}_{rm}$  represent unit vectors in the positive x, y and z-direction of the RM marker, respectively.

The reaction force mirrors the action force but acts in the opposite direction.

$$F_r = -F_a$$

$$T_r = -T_a$$

$F_r$  represents the translational reaction applied at the JFLOAT marker, while  $T_r$  represents the rotational reaction.

## 2.7 Multi-component force

A force with three or more components, known as a multi-component force or torque, is employed to apply translational and/or rotational force between two components in a model. The force is directed from the first part, referred to as the action body, to the second part, called the reaction body, automatically applied by *Adams*.

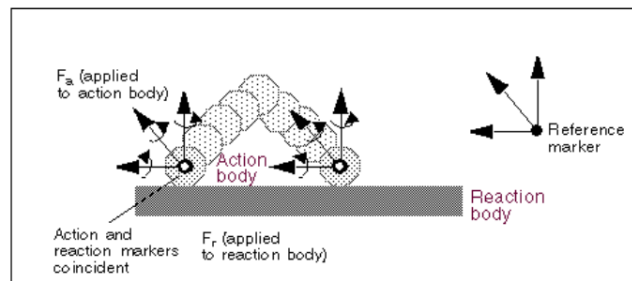


Figure 2.1: Example of the action and reaction force movement [21]

To identify the points of application for the multi-component force, *Adams* establishes distinctive markers for each part. The action marker is associated with the action body, while the

reaction marker corresponds to the reaction body. Throughout the simulation, *Adams* ensures the coincidence of the reaction marker with the action marker. The floating nature of the reaction marker, not fixed to the body to which it belongs, makes it commonly referred to as a floating marker. These markers are also termed I and J markers.

Moreover, *Adams* generates a third marker known as the reference (R) marker, providing information about the force direction. The user has the flexibility to define the orientation of the reference marker during the creation of a multi-component force.

In the case of a six-component general force and a three-component force, the total force provided by *Adams* Solver results from the vector sum of the specified force components. The magnitude of this force is the square root of the sum of the squares of the three mutually-orthogonal force components:

$$\bar{F}_a = FX\hat{x}_{rm} + FY\hat{y}_{rm} + FZ\hat{z}_{rm}$$

Where:

- $\bar{F}_a$  is the action force;
- $FX$  is the user-defined function determining the magnitude and sign of the x-component.
- $FY$  is the user-defined function determining the magnitude and sign of the y-component.
- $FZ$  is the user-defined function determining the magnitude and sign of the z-component.
- $\hat{x}_{rm}$ ,  $\hat{y}_{rm}$ , and  $\hat{z}_{rm}$  are unit vectors along the +x, +y, and +z directions of the reference marker.

Reaction forces is:

$$\bar{F}_r = -\bar{F}_a$$

## 2.8 Bistop function

As illustrated in *Figure 5.7* below, the BISTOP function is employed to simulate a gap element. This gap element is characterized by a slot that establishes the motion domain for a component denoted as Part I. When Part I remains within the slot without interference at its ends, it can freely move without external forces exerted upon it.

However, as Part I attempts to surpass the physical limits of the slot, the BISTOP function comes into play, generating impact forces that simulate contact. These forces are strategically designed to guide Part I back into the slot, maintaining adherence to the specified constraints.

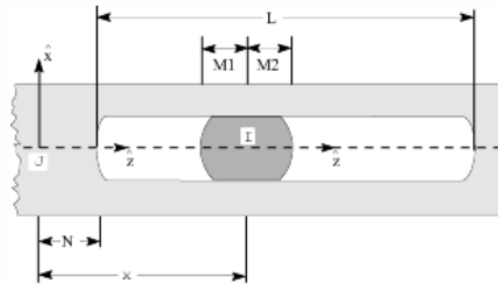


Figure 2.2: Example of the Bistop function [21]

The BISTOP function has the following definition:

$$BISTOP(x, \dot{x}, x_1, x_2, k, e, c_{max}, d)$$

Where:

- $x$ : variable representing the distance that you intend to utilize for force computation (N+M1 for the left end of the slot in *Figure 5.7*).
- $\dot{x}$ : time derivative of  $x$ .
- $x_1$ : lower bound of  $x$  (N+M1 in *Figure 5.7*) - if the variable  $x_1$  is greater than  $x$ , the *Adams* Solver computes a positive force value.
- $x_2$ : upper bound of  $x$  (N+L-M2 in *Figure 5.7*) - if the variable  $x_2$  is lower than  $x$ , the *Adams* Solver computes a negative force value.
- $k$ : stiffness.
- $e$ : exponent of the force deformation characteristic.
- $c_{max}$ : maximum damping coefficient.
- $d$ : distance (penetration) at which we decide to apply full damping coefficient.

The BISTOP force is comprised of two components. The first involves a stiffness factor contingent on the degree of penetration of Part I into the constraining Part J. The second component introduces a damping or viscous element, providing a means to simulate energy dissipation within the system. This comprehensive model not only captures the dynamic interactions within the simulated environment but also allows for a nuanced representation of mechanisms involving energy loss.

To avoid a sudden discontinuity in the damping force at zero penetration, we define the damping coefficient as a cubic step function of the penetration. This means that the damping coefficient is consistently zero at zero penetration. It gradually reaches its maximum, denoted as  $c_{max}$ , at a penetration value  $d$ , set by user.

## 2.9 Translational Spring Damper

A translational spring damper characterizes the forces between two components, which locations are defined by user, acting over a specified distance in a particular direction.

Upon selection, *Adams* applies an action force to the chosen initial component, termed the action body, and exerts an equal and opposite reaction force on the second selected component, known as the reaction body. These forces are directed along the line connecting the endpoints of the spring-damper, commonly referred to as the line of sight. A positive action force tends to drive the action body away from the reaction body, while a negative action force tends to draw the action body toward the reaction body.

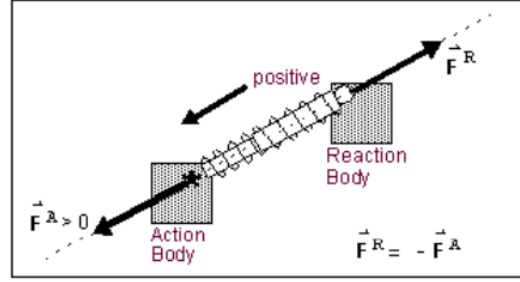


Figure 2.3: Example of a Translational Spring Damper [21]

Flexibility is provided in defining the damping and stiffness values. Users can express these values as coefficients or employ splines to establish relationships between damping and velocity or stiffness and displacement. Furthermore, the system allows setting the stiffness value to 0 for a pure damper or the damping value to 0 for a pure spring.

Additional customization includes specifying a preload force to further tailor the system's behavior to their simulation requirements.

The translational force exerted by a spring damper is directly proportional to the relative displacement and velocity of the two defined locations that establish the endpoints of the spring damper. The mathematical representation of the action force is given by the linear relation:

$$force = -C \left( \frac{dr}{dt} \right) - K(r - length) + preload$$

Where:

- $r$ : distance between the two designed markers along the line-of-sight.
- $\frac{dr}{dt}$ : relative velocity of the markers along the line-of-sight.
- $C$ : viscous damping coefficient.
- $K$ : spring stiffness coefficient.
- $preload$ : spring reference force.
- $length$ : reference length, therefore when  $r = length$ , the force attains the specified preload force.

This equation captures the linear relationship governing the translational force behavior of the spring damper, considering displacement, velocity, damping, stiffness, preload force, and reference length.

# Chapter 3

## Fundamentals of fluid-dynamic simulation

### 3.1 Introduction

Fluid dynamics operates within the framework of classical physics, adhering to the fundamental principles of conserving mass, momentum, and energy. It is a common practice to express conservation laws under the hypothesis that the fluid behaves as a continuous medium, known as the continuum hypothesis [26].

Selecting a fluid dynamics model involves considering factors such as the flow type (whether laminar or turbulent), boundary conditions (like solid walls or inlets), the complexity of the problem, available computational resources, and the presence of experimental data for validation. Ultimately, the choice is a balance between precision and resource availability, with a keen understanding of each model's limitations and a need for validation to ensure accuracy.

This section is focused on the theoretical foundations of the physical phenomena investigated through fluid dynamic analysis, concentrating on introducing the equations employed by the *scFLOW* software in its computational process.

### 3.2 Conservation equations

The continuum assumption allows us to discuss the characteristics of a fluid at a specific spatial point and the physical attributes of an extremely small volume element within the fluid, which we refer to as a material particle.

A fluid's behavior can be fully characterized by specifying the physical attributes of each material particle over time, through the Lagrangian and Eulerian formulations.

The Lagrangian approach involves tracking individual material particles over time, while the Eulerian method focuses on the evolution of flow properties at fixed points within the domain. While the Eulerian approach is generally more convenient for analysis, there are cases where the Lagrangian formulation is preferred, particularly in tracking fluid interfaces. In most situations, flow properties at fixed locations, such as pressure on a wall, are easily obtained through the Eulerian formulation [26].

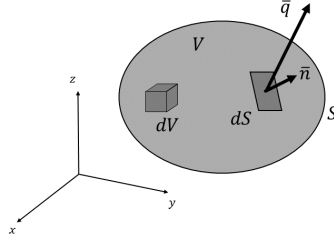


Figure 3.1: Control volume

Given a control volume, mass, momentum, and energy are exchanged through its boundaries. Taking these exchanges into account, the conservation equations for the mentioned quantities are formulated. The integral form of the mass conservation equation is written as:

$$\frac{\partial}{\partial t} \int_V \rho dV = - \int_S \rho (\bar{q} \cdot \bar{n} dS) \quad (3.1)$$

Where:

- $V$  is the control volume and  $S$  is its surface;
- $\rho$  is the fluid density;
- $\bar{q}$  is the fluid velocity;
- $\bar{n}$  is the normal to the control volume surface, conventionally assumed positive when outward. In this case, there are no source terms as mass injections into the considered volume are assumed to be absent.

By applying the Gauss theorem, we arrive at the mass equation written in the form of divergence or conservative differential:

$$\frac{\partial \rho}{\partial t} = \nabla \cdot (\rho \bar{q}) \quad (3.2)$$

The conservation equation for momentum is developed based on Newton's second law: the sum of inertia forces and momentum fluxes is equal to the resultant of external forces acting on the system.

$$\frac{\partial}{\partial t} \int_V \rho \bar{q} dV = - \int_S \rho \bar{q} (\bar{q} \cdot \bar{n} dS) + \int_S \bar{\sigma} \cdot \bar{n} dS + \int_V \rho \bar{f}_c dV \quad (3.3)$$

Where:

- $\bar{f}_c$  is the force (gravity, centrifugal etc.) per unit mass;
- $\sigma$  is the stress tensor, with:
  - $\sigma_n = -p \bar{n}$ , if the fluid is ideal;
  - $\sigma_n = -p \bar{n} + \bar{\tau}_n$ , if the fluid is real (viscous), where  $\tau_n$  is the viscous friction stress.



By applying the Gauss theorem:

$$\frac{\partial \rho \bar{q}}{\partial t} = -\nabla \cdot (\rho \bar{q} \bar{q}) + \nabla \cdot \bar{\sigma} + \rho \bar{f}_c \quad (3.4)$$

The energy balance law equates the change in total energy within the control volume to the power of the forces applied to the system and the thermal flux.

$$\frac{\partial}{\partial t} \int_V \rho E dV = - \int_S \rho E (\bar{q} \cdot \bar{n} dS) + \int_S (\bar{\sigma} \cdot \bar{n}) \cdot \bar{q} dS + \int_V \rho \bar{f}_c \cdot \bar{q} dV - \int_S \bar{q}_T \cdot \bar{n} dS \quad (3.5)$$

Where  $\bar{q}_T$  represents the thermal flux and  $E$  is the total energy per unit mass, defined as:

$$E = e + \frac{q^2}{2}$$

where  $e$  is the internal energy and  $\frac{q^2}{2}$  is the kinetic energy. By applying the Gauss theorem:

$$\frac{\partial \rho E}{\partial t} + \nabla \cdot (\rho E \bar{q}) = \nabla \cdot \bar{\sigma} \cdot \bar{q} + \rho \bar{f}_c \cdot \bar{q} - \nabla \cdot \bar{q}_T \quad (3.6)$$

In conclusion, a system of 5 equations in 14 unknowns ( $\rho, \bar{q}, \bar{\sigma}, e, \bar{q}_T$ ) is obtained. To complete the system, the following equations are added:

- For a Newtonian fluid, we have:

$$\tau_{ij} = \mu \left( \frac{\partial q_i}{\partial x_j} + \frac{\partial q_j}{\partial x_i} \right) - \frac{2}{3} \mu \nabla \cdot \bar{q} \delta_{ij}$$

where  $\delta_{ij}$  is the Kronecker delta and is defined as:

$$\delta_{ij} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$$

- Fourier's law for thermal conduction:

$$q_T = -k \nabla T$$

where  $k$  is the thermal conductivity of the fluid and  $T$  is the temperature;

- The equation of state for perfect gases:

$$p = \rho R T$$

- The internal energy equation for a calorically perfect gas:

$$e = C_v T$$

where  $C_v$  is the specific heat at constant volume.

The equations mentioned before, known as the Navier-Stokes equations, form a nonlinear system of partial differential equations that cannot be solved exactly analytically.

### 3.3 Turbulence modeling

Turbulence, as defined by von Karman and refined by Hinze, is characterized by irregular fluid motion with random variations in quantities over time and space coordinates, as in [27]. It results as an instability in laminar flow, driven by the nonlinear inertial and viscous terms of the Navier-Stokes equations.

The continuum nature of turbulence, described by three-dimensional equations, presents challenges in numerical simulation due to the vast range of scales. In particular, the phenomenon of vortex stretching<sup>1</sup> and three-dimensionality emphasize the inadequacy of two-dimensional approximations. Turbulence shows a spectrum of scales, involving a cascade process transferring energy from larger to smaller eddies.

Enhanced diffusivity in turbulent flows significantly impacts mass, momentum, and energy transfer, with apparent stresses being orders of magnitude larger than in laminar flows. Because large swirling motions endure for significant distances, the diffusivity and stresses rely on flow history and cannot be solely defined as functions of local flow properties. Moreover, the dissipation of turbulence energy by small eddies is influenced by the rate at which they receive energy from the larger eddies [27].

#### 3.3.1 RANS

The RANS approach involves statistically averaging the instantaneous Navier-Stokes equations through a process that considers a timescale larger than turbulence time scale but smaller than the the mean flow evolution time. This results in averaged mass and momentum conservation equations of the flow [5].

This methodology allows obtaining a system of equations capable of describing the mean field of a generic signal, foregoing a detailed description of turbulent fluctuations.

A Reynolds averaging operator is introduced, so that the generic signal  $u(t)$  can be decomposed as:

$$u_i(t) = \bar{U}_i + u'_i(t)$$

Where:

$\bar{U}_i$  is the mean value of the signal;

$u'_i(t)$  is its time fluctuation.

Depending on the nature of the problem, the Reynolds average can be defined in various ways (statistically stationary<sup>2</sup> problems, problems with homogeneous<sup>3</sup> turbulence, etc.)

In the general case, an ensemble average is used. Given a non-stationary and non-homogeneous flow in space, the mean value is calculated over the number of experiments  $N$ , at a point within the domain  $\bar{x}$  at the time instant  $t$ .; perturbations in each experiment are introduced by different initial conditions. It can be expressed as:

$$\bar{u}_i(\bar{x}, t) = \lim_{N \rightarrow \infty} \frac{1}{N} \sum_{m=1}^N u_i^m(\bar{x}, t)$$

---

<sup>1</sup>Vortex stretching refers to the process in fluid dynamics where vortices, or rotating regions of fluid, experience extension connected to a proportional rise in the vorticity component along the stretching axis. Vortex stretching plays a crucial role in the dynamics of turbulence, contributing to the transfer of energy between different scales of motion in a turbulent flow.

<sup>2</sup>Problems where time is the direction of homogeneity; the mean field is independent of time.

<sup>3</sup>In this case, the average does not depend on the space used for averaging

In the case of compressible flows, the Favre average is used, so we have:

$$\tilde{u}_i(x) = \frac{1}{\bar{\rho}T} \lim_{T \rightarrow \infty} \int_{t+T}^T \rho(x, t) u_i(\bar{x}, t) dt$$

Where  $T$  is the chosen period of time for integration,  $\rho(x, t)$  is the instantaneous density, and  $\bar{\rho}$  is the Reynolds-averaged density.

We start from the Navier-Stokes equations for incompressible flows in differential form. Let's consider the components of the mass and momentum equations along the  $x_i$  axis (the same procedure can be repeated for the other different axes):

$$\begin{cases} \frac{\partial q_i}{\partial x_i} = 0 \\ \rho \frac{\partial q_i}{\partial t} + \rho q_j \frac{\partial q_i}{\partial x_j} = -\frac{\partial p}{\partial x} + \frac{\partial \tau_{ij}}{\partial x_j} \end{cases} \quad (3.7)$$

The Reynolds decomposition is introduced, and an ensemble averaging operator is applied to the entire equation. The following properties of the averaging operator are considered:

- $\overline{q'_i} = 0$ ;
- $\overline{\bar{q}_i} = \bar{q}_i$ ;
- $\overline{\frac{\partial q_i}{\partial x_i}} = \frac{\partial \bar{q}_i}{\partial x_i}$ ;
- $\overline{\bar{q}_i + q'_i} = \bar{q}_i + \overline{q'_i}$  (distributive property).

This leads to the following equations:

$$\begin{cases} \frac{\partial \bar{q}_i}{\partial x_i} = 0 \\ \rho \frac{\partial \bar{q}_i}{\partial t} + \rho \bar{q}_j \frac{\partial \bar{q}_i}{\partial x_j} = -\frac{\partial p}{\partial x} + \frac{\partial}{\partial x_j} (\overline{\tau_{ij}} + \tau_{ij}^R) \end{cases} \quad (3.8)$$

Where:

$$\tau_{ij}^R = -\rho \overline{q'_i q'_j}$$

is the Reynolds stress tensor. The latter is added to the viscous stress tensor and describes, through a diffusive term, the effect of turbulent fluctuations.

### 3.4 Boussinesq closure model

The closure of the RANS equations is achieved by modeling the Reynolds stress tensor, taking into account certain assumptions.

One possible approach involves introducing an eddy viscosity with a linear Boussinesq [4] model. In the Navier-Stokes equations, microscopic molecular fluctuations can be interpreted at a macroscopic level as a diffusive term through the introduction of a viscosity. Following this procedure, the Reynolds stress is described through viscosity as a diffusive term, as follows:

$$\tau_{ij}^R = 2\mu_t \bar{S}_{ij} - \frac{2}{3}\mu_t \frac{\partial \bar{q}_k}{\partial x_k} \delta_{ij} - \frac{2}{3}\rho K \delta_{ij} \quad (3.9)$$

where:

- $\mu_t$  is the turbulent viscosity;

- $\bar{S}_{ij} = \frac{1}{2} \left( \frac{\partial \bar{q}_i}{\partial x_j} \frac{\partial \bar{q}_j}{\partial x_i} \right)$  is the velocity strain tensor calculated on averaged quantities;
- $K = \frac{1}{2} \overline{q'_i q'_i} = -\frac{1}{2\rho} \tau_{ii}^R$  is the turbulent kinetic energy;
- $\delta_{ij}$  is the Kronecker delta.

The last term in the equation ensures that the trace of the tensor is representative of the kinetic energy of fluctuations. The anisotropic part of the equation is proportional to  $\bar{S}_{ij}$ .

### 3.4.1 Standard $k - \varepsilon$ model

As the eddy viscosity ( $\mu_t$ ) is not constant and it varies based on the flow conditions and its spatial location, it requires a distinct definition for each unique problem or scenario.

To address this variability and enhance precision, a strategy involves identifying a '*fundamental quantity of turbulence*'. This involves solving a set of equations that account for advection, diffusion, formation, and consumption associated with this fundamental quantity and subsequently the eddy viscosity is determined based on these ones. In this context, the identified '*fundamental quantity of turbulence*' comprises two key parameters: turbulence energy ( $k$ ) and turbulence dissipation rate ( $\varepsilon$ ). By anchoring the description of eddy viscosity to these fundamental turbulence parameters, the approach aims to offer a more accurate representation, taking into account the dynamic nature of flow conditions. We define:

$$k = \frac{1}{2} \overline{u'_i u'_i} \quad (3.10)$$

$$\varepsilon = \nu \overline{\frac{\partial u'_i}{\partial x_j} \frac{\partial u'_i}{\partial x_j}} \quad (3.11)$$

The  $k - \varepsilon$  equations, which include the advection, diffusion, formation, and consumption of turbulence energy ( $k$ ) and turbulence dissipation rate ( $\varepsilon$ ), are conventionally described for compressible fluids through empirical equations as follows:

$$\frac{\partial \bar{\rho} k}{\partial t} + \frac{\partial \bar{u}_i \bar{\rho} k}{\partial x_i} = \frac{\partial}{\partial x_i} \left( \frac{\mu_t}{\sigma_k} \frac{\partial k}{\partial x_i} \right) + G_S - G_{S1} - G_{S2} - G_{S3} - \bar{\rho} \varepsilon \quad (3.12)$$

$$\frac{\partial \bar{\rho} \varepsilon}{\partial t} + \frac{\partial \bar{u}_i \bar{\rho} \varepsilon}{\partial x_i} = \frac{\partial}{\partial x_i} \left( \frac{\mu_t}{\sigma_\varepsilon} \frac{\partial \varepsilon}{\partial x_i} \right) + C_1 \frac{\varepsilon}{k} (G_S - G_{S1} - G_{S2} - G_{S3}) - C_2 \frac{\bar{\rho} \varepsilon^2}{k} \quad (3.13)$$

Where:

$$G_S = \mu_t \left( \frac{\partial \bar{u}_i}{\partial x_j} + \frac{\partial \bar{u}_j}{\partial x_i} \right) \frac{\partial \bar{u}_i}{\partial x_j}$$

$$G_{S1} = \frac{2}{3} \bar{\rho} k D$$

$$G_{S2} = \frac{2}{3} \mu_t D^2$$

$$G_{S3} = \frac{\mu_t}{\sigma_t \bar{\rho}^2} \frac{\partial \bar{\rho}}{\partial x_i} \frac{\partial \bar{P}}{\partial x_i}$$

$$D = \frac{\partial \bar{u}_i}{\partial x_i}$$

These equations include numerous empirical parameters, provided below (Table 3.1). Employing these parameters significantly enhances precision when compared to the method of individually assessing the eddy viscosity for each specific problem.

$\sigma_k$	$\sigma_\varepsilon$	$C_1$	$C_2$	$C_3$	$C_\mu$	$\sigma_t$
1	1.3	1.44	1.92	0	0.09	0.9

 Table 3.1: Standard  $k - \varepsilon$  model constant parameters values

From  $k$  and  $\varepsilon$  values we can obtain:

$$\mu_t = C_\mu \rho \frac{k^2}{\varepsilon} \quad (3.14)$$

### 3.4.2 SST $k - \omega$ model

Introduced by Wilcox et al.[27], the  $k - \omega$  model stands as a two-equation turbulence model like  $k - \varepsilon$  models. Rather than directly addressing turbulence dissipation, it treats the dissipation rate per unit turbulence energy, denoted as  $\omega \sim \varepsilon/k$ , carrying the dimension of frequency [1/s]. This modeling approach boasts advantages in replicating near-wall turbulence behavior, eliminating the need for damping functions to acquire a near-wall velocity profile, a necessity in the low-Reynolds-number  $k - \varepsilon$  model. Furthermore, it offers an enhanced estimation of boundary layer separation under adverse pressure gradients. However, a recognized challenge lies in its pronounced reliance on boundary conditions, such as inflow or free-stream turbulence values, limiting its reliability in the outer free-stream layer.

The SST model, an innovation by Menter[11], solves the equations for  $k$  and  $\omega$  through a zonal treatment. In this method, the conventional k-omega equations developed by Wilcox find resolution in the near-wall regions, shifting towards outer regions to align with the  $k - \varepsilon$  model. This adjustment promises a computation that is both accurate and robust. Additionally, the application of the Shear-Stress Transport concept in the SST model prevents the overestimation of eddy viscosity under adverse pressure gradients. Notably, it faithfully reproduces intricate separation phenomena that conventional eddy viscosity models may struggle to capture.

In the k-omega model, the expression for eddy viscosity is articulated as follows:

$$\mu_t|_{k-\omega} = \rho \frac{k}{\omega} \quad (3.15)$$

The transport equation for  $k$  follows the same structure as those found in other low-Reynolds number  $k - \varepsilon$  models, albeit with a different expression for energy dissipation:  $\varepsilon = C_\mu k \omega$ . The equation for  $\omega$  is given by:

$$\frac{\partial}{\partial t} \rho \omega + \frac{\partial}{\partial x_j} u_j \rho \omega = \frac{\partial}{\partial x_j} \left[ \left( \mu + \frac{\mu_t}{\sigma_\omega} \right) \frac{\partial \omega}{\partial x_j} \right] + \frac{\gamma \rho}{\mu_t} G_s - \beta \rho \omega^2$$

By incorporating the cross-diffusion term,  $CD_{k\omega}$ , into the above equation for  $\omega$ , the k-omega equations are analytically equivalent to the  $k - \varepsilon$  model:

$$CD_{k\omega} = 2 \frac{\rho}{\sigma_\omega \omega} \frac{\partial k}{\partial x_j} \frac{\partial \omega}{\partial x_j}$$

In the SST model, a gradual transition from the k-omega model to the  $k - \varepsilon$  model is achieved through the introduction of a blending function, which considers wall distance and turbulence quantities. This blending function is multiplied by the cross-diffusion term mentioned above. Additionally, model constants for both inner and outer layers are interpolated using the same blending function. These constants are detailed below, with subscripts indicating the inner (1) and outer (2) layers:

$$\begin{aligned}
 C_\mu &= 0.09 \\
 \sigma_{k1} &= 1.18, \quad \sigma_{k2} = 1.0 \\
 \sigma_{\omega1} &= 2.0, \quad \sigma_{\omega2} = 1.17 \\
 \beta_1 &= 0.075, \quad \beta_2 = 0.0828 \\
 \gamma &= \frac{\beta}{C_\mu} - \frac{\kappa^2}{\sigma_\omega \sqrt{C_\mu}}, \quad \kappa = 0.41
 \end{aligned}$$

The eddy viscosity is:

$$\mu_t|_{SST} = \rho \frac{a_1 k}{\Omega}$$

Where  $a_1 = 0.31$  and  $\Omega$  represents the magnitude of mean vorticity. Both expressions (1) and (5) are interpolated using another blending function that incorporates wall distance and turbulence quantities as parameters.

When applying a low-Reynolds-number correction for turbulence production as proposed by Wilcox, the parameters  $a_1$ ,  $\gamma$ , and  $C_\mu$  in the dissipation term of the  $k$  equation ( $C_\mu \rho \omega k$ ) are modified as follows:

$$\begin{aligned}
 a'_1 &= \frac{\beta_1/3 + Re_\tau/R_k}{1 + Re_\tau/R_k} \\
 \gamma' &= \frac{\gamma_0 + Re_\tau/R_\omega}{1 + Re_\tau/R_\omega} \gamma \\
 C'_\mu &= \frac{4/15 + (Re_\tau/R_\beta)^4}{1 + (Re_\tau/R_\beta)^4} C_\mu
 \end{aligned}$$

Where:

$$\begin{aligned}
 \gamma_0 &= \frac{1}{9} \\
 R_\beta &= 8 \\
 R_k &= 6 \\
 R_\omega &= 2.95
 \end{aligned}$$

### 3.5 Numerical methods for fluid dynamics

The mathematical models just described result in a system of partial differential equations having no exact analytical solutions. Therefore, it is necessary to discretize the spatial and temporal domains by transforming the continuous problem into a discrete problem that can be solved numerically. There are three mainly used discretization methods:

1. *Finite differences*: only structured grids are used here. The spatial derivatives of variables are approximated by difference between the values they take at discrete points of neighboring cells. The number of cells involved in the discretization depends on the degree of the derivative being approximated and the precision required in the problem.

2. *Finite elements*: discretization using finite elements consists of approximating solutions by means of shape functions, defined in each element. Shape functions are polynomials or other locally defined functions and approximate the value of a given physical quantity in each element. Through this discretization, the problem is reformulated in variational form.
3. *Finite volumes*: the domain is discretized by cells, on each of which a conservation principle is applied. The momentum field is then solved locally, and then it is integrated over the entire computational domain

### 3.5.1 Finite volume method

The finite volume method involves dividing the flow or thermal field into discrete elements, where fluid or heat can flow between neighboring elements through what is termed as "Numerical flux". This flux is computed based on the governing equations with discretized data on each element. By determining numerical flux for each element face, variations in data over time and space within each element can be tracked and updated. This method, fundamental to *scFLOW*'s calculation workflow, ensures conservation laws are accurately represented.

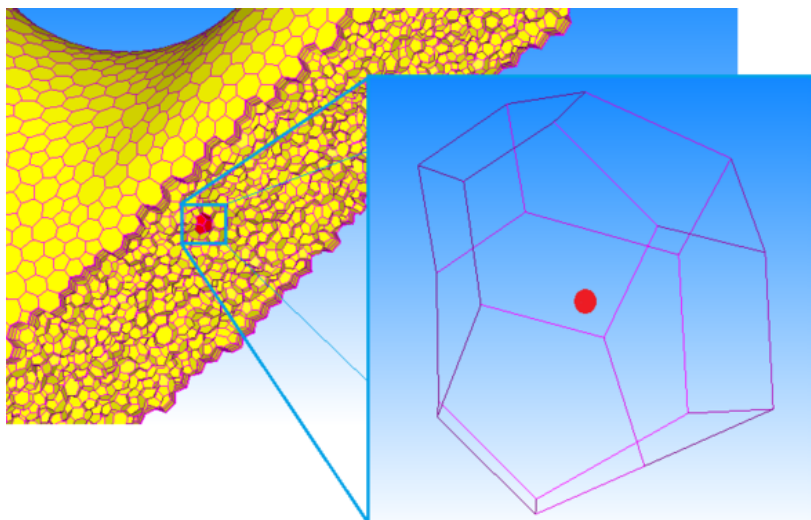


Figure 3.2: Data and element [22]

In *scFLOW*, elements are shaped as arbitrary polyhedrons, with data only available at their centroids. The governing equations are expressed with these discretized data, considering elements in a two-dimensional field. The time variations of variables at a given element are influenced by flow across its surface (advection), gradient differences between neighboring elements (diffusion), and any arbitrary generation or depletion of quantities within the element (source).

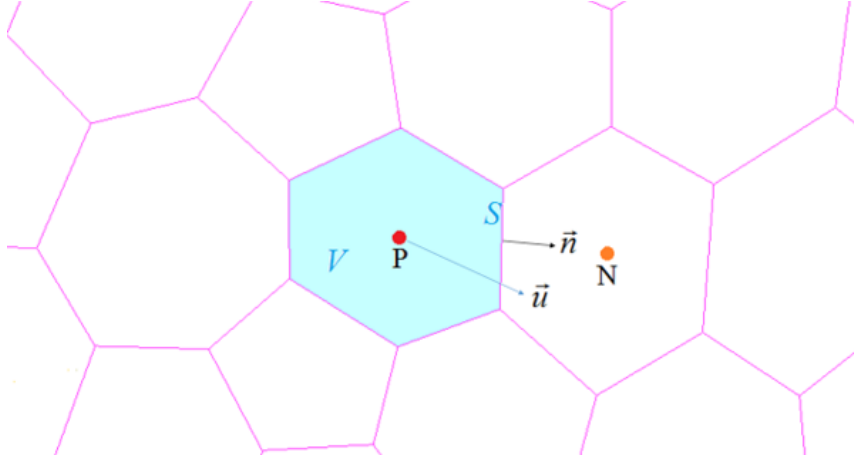


Figure 3.3: Data and element in 2D view [22]

The Navier-Stokes equation system can be write in the following form:

$$\frac{\partial \phi}{\partial t} + \frac{\partial(\phi u_j)}{\partial x_j} = \frac{\partial}{\partial x_j} \left( \alpha \frac{\partial \phi}{\partial x_j} \right) + S \quad (3.16)$$

where, terms from the left-hand side represent 'Unsteady (unknown)', 'Advection', 'Diffusion', and 'Source'. The function  $\phi(t, x_i)$ , representing the distribution in space and time, is determined by solving this equation. For instance, in the case of mass conservation where  $\phi = \rho$  and  $\alpha$  is the diffusion coefficient for the species, the solution yields the rate of mass variation per unit time per unit volume.

Naming with P the element centroid and with N the centroid of its neighbor element, considering also element volume ( $V$ ), surface area ( $S$ ), unit vector perpendicular to the element surface ( $\vec{n}$ ), and velocity vector ( $\vec{u}$ ), the Equation 3.16 is reformulated into an integral form (Eq. 3.17) using volume and surface integrals with Gauss's divergence theorem.

This allows consideration of equations on each element individually, with surface integral terms representing numerical fluxes.

$$\frac{\partial}{\partial t} \int_V \phi dV + \int_S \phi \vec{u} \cdot \vec{n} dS = \int_S \alpha (\nabla \phi) \cdot \vec{n} dS + \int_V S dV \quad (3.17)$$

### Unsteady term

The unsteady term in equation 3.17 represents the time variation of  $\phi$ , approximated using a linear method (Eq. 3.18), although higher-order accuracy approximations are also possible.

$$\frac{\partial \phi}{\partial t} \approx \frac{(\phi^{n+1} - \phi^n) \Delta V}{\Delta t} \quad (3.18)$$

This term only focuses on time variation and does not involve numerical fluxes.

### Advection term

In the computation of the advection term, the first-order partial derivative in the governing equation highlights the inflow and outflow of a quantity within each element, influenced by the local flow velocity ( $u_j$ ). This term just focuses on the spatial transport of the function  $\phi$ , and the gradient of quantity should not be changed by this term. Numerical fluxes play a crucial role in



modeling these effects within the finite volume method, determined on the element surfaces despite data being available only at the element centroids. Therefore, an approximation technique like interpolation or extrapolation becomes necessary to compute numerical fluxes accurately with discretized data. Various interpolation methods can be employed, with higher accuracy achieved by increasing the number of reference data points. Considering the nature of flow, employing more data points on the upstream side is preferable. This strategy, known as "upwind differencing", involves both first-order and second-order approximation methods depending on the selection of data points. Figure 3.4 illustrates the concept in a one-dimensional flow scenario,

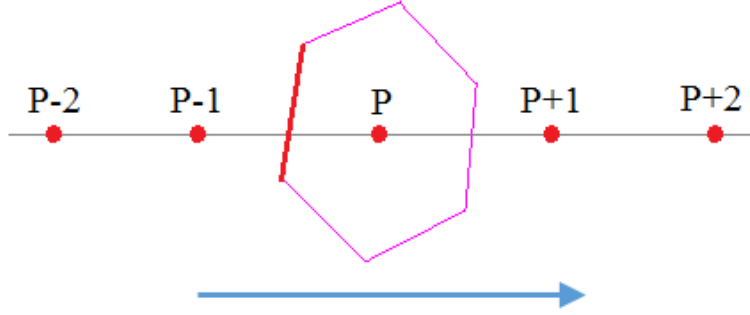


Figure 3.4: Data involved in calculating the numerical flux on the red surface [22]

where the numerical flux computation on a red-colored surface varies based on the number of employed data points. If only data from P-1 is considered, it results in a first-order upwind differencing. Including data from both P-1 and P+1 leads to central differencing, while incorporating data from P-2, P-1, and P results in second-order upwind differencing. As previously mentioned, increasing the number of data points improves the accuracy of the numerical flux approximation. However, using higher-order approximations can reduce computational robustness. In such cases, a combination of lower-order approximations or the application of limiter functions may be necessary to maintain accuracy.

The numerical flux on the surface  $S_j$  is represented by  $F_j$ . By summing the numerical fluxes from all surfaces of an element, the total numerical flux of the element is obtained. This relationship is approximated as:

$$\frac{\partial \phi u}{\partial x_i} \approx \sum_j F_j$$

It should be noted that  $F_j$  is calculated using data from multiple neighboring elements. For instance, when employing first-order upwind differencing as depicted in Figure 3.4,  $F$  can be computed using the following equation:

$$F = (\phi u)_{p-1}$$

### Diffusion term

On the other hand, the diffusion term in equation 3.16 is represented by the spatial cross-derivative; In equation 3.17, the gradient of  $\phi$  varies in space, and the function  $\phi$  smoothes by the coefficient  $\alpha$ . This term accounts for phenomena like viscosity and thermal conduction, facilitating homogeneous diffusion or dissipation of  $\phi$  without restricting spatial direction as advection does. The calculation of numerical flux for diffusion involves calculating gradients at

element centroids, which are then interpolated to determine gradients on element surfaces. Now, consider the numerical flux of diffusion on the surface  $S_j$ , which we'll represent as  $D_j$ . To find the total numerical flux of the element, we sum up all the  $D_j$  values.

$$\int_S \alpha(\nabla\phi) \cdot \vec{n} dS$$

is approximately equal to  $\sum_j D_j$ . In this context,  $D_j$  is computed using data from multiple elements, and the transport coefficient  $\alpha$  is factored into the calculation.

### Source term

Finally, the source term, directly influences the production or disappearance of  $\phi$  regardless of flow or thermal field. Its calculation involves multiplying the given production rate per unit volume by the element volume. It accounts for phenomena such as local heating, mass production, or disappearance due to chemical reactions.

$$\int_V S dV \approx S\Delta V \tag{3.19}$$

In summary, through the discretization process outlined above, the governing equation can be approximately solved given the time step and element volume. The resulting equation, termed the "Discretized equation", involves arithmetic calculations and incorporates terms representing advection, diffusion, and source effects, allowing for computational modeling of complex fluid or thermal dynamics. The Discretized equation is:

$$\frac{(\phi^{n+1} - \phi^n)\Delta V}{\Delta t} + \sum_j F_j(\phi, u) + \sum_j D_j(\phi, u, \alpha) = S\Delta V \tag{3.20}$$

### 3.5.2 The matrix solver

In numerical simulations, there are two approaches to calculate numerical fluxes and source terms: explicit and implicit time integration. Explicit integration uses properties at time  $n$ , while implicit integration uses properties at time  $n+1$ . Implicit integration solves the discretized equations for all elements simultaneously, often employed for steady-state analyses due to its computational efficiency.

The system of equations is represented in matrix form:

$$\begin{bmatrix} A \end{bmatrix} \begin{bmatrix} \vdots \\ \phi_i \\ \vdots \end{bmatrix} = \begin{bmatrix} \vdots \\ S_i \\ \vdots \end{bmatrix} \tag{3.21}$$

where  $\phi_i$  and  $S_i$  are column vectors, and  $[A]$  is a coefficient matrix containing numerical fluxes. Equation 3.21 is solved using matrix inversion in equation 3.22, known as the matrix solver.

$$\begin{bmatrix} \vdots \\ \phi_i \\ \vdots \end{bmatrix} = \begin{bmatrix} A \end{bmatrix}^{-1} \begin{bmatrix} \vdots \\ S_i \\ \vdots \end{bmatrix} \tag{3.22}$$

Iterative methods like the Krylov subspace method are commonly used for matrix inversion, with the goal of reconstructing a diagonally dominant matrix for efficient iteration, termed "pre-conditioning".

## 3.6 Pressure-based solver

A solver for thermal fluid analysis employs a designated computational approach (algorithm) to address fundamental equations outlined in Conservation equations. Numerous computational methods exist, with "pressure-based" and "density-based" being specific calculation method names. Originally trademarks, these terms have become widely adopted as common names in thermal fluid analysis software, including *scFLOW*. The "pressure-based solver" and "density-based solver" within *scFLOW* represent distinct methodologies for solving basic equations, each tailored to ensure mass conservation compliance [22].

The pressure-based solver in *scFLOW* employs a method of pressure correction that addresses mass conservation, momentum conservation, and energy conservation equations in distinct steps. Through adjustment of pressure variations during momentum conservation, both velocity and pressure are calibrated to ensure alignment with the momentum equation's outcomes, thereby satisfying mass conservation principles. Due to the predominant adjustment of pressure to meet mass conservation requirements, this methodology is aptly termed the "pressure-based solver." The pressure-based solver doesn't focus only on mass conservation, especially evident in incompressible fluid analyses where density disappears from mass conservation equations. In such scenarios, traditional methods struggle to solve equations over time due to this absence of density data. However, by concurrently addressing momentum conservation over time, the pressure-based solver adeptly adjusts pressure to ensure mass conservation compliance. This flexibility proves advantageous in scenarios involving minute or rapid density fluctuations, where meticulous adjustment of mesh size and time-step becomes crucial for accurately simulating density changes. Traditional methods may face computational challenges when handling such scenarios, resulting in increased calculation times and potentially compromised accuracy.

One of the notable advantages of the pressure-based solver is that it's not affected by density fluctuations, allowing for dynamic adjustments in calculation speed to accommodate varying flow conditions. This adaptability significantly enhances efficiency in incompressible fluid analyses, where computational resources can be allocated optimally based on the prevailing flow dynamics. By separately addressing each conservation equation, the pressure-based solver enables the derivation of vital flow field parameters (velocity and pressure) without necessitating the resolution of energy conservation equations, particularly in scenarios where energy changes can be disregarded. As a result, this method finds widespread application in incompressible fluid analyses, particularly those where natural convection phenomena are not a primary consideration.

# Chapter 4

## Fundamentals of cosimulation

The potential of co-simulation is limitless in addressing crucial multiphysics problems, driven by the escalating demand to optimize intricate product designs through interdisciplinary approaches. Co-simulation holds the promise of deploying leading solvers across diverse physics domains and spatiotemporal scales.

This chapter outlines the analysis of fluid-mechanism interaction using two software tools, *scFLOW* and *Adams*, enabled by *MSC CoSim*. While *scFLOW* utilizes an implicit scheme for fluid-solving, *MSC CoSim* employs an explicit weak coupling scheme. The exchange of forces, torque, and data for rigid bodies involves virtual nodes.

Values calculated in *Adams* coordinates include:

- Coordinates of rotational center [m];
- Euler angles [rad];
- Integral value of Force on surface [N];
- Integral value of torque on surface [Nm].

*ScFLOW* requires *Adams* coordinate system information for input. The exchange of physical quantities can be visually represented as shown in Fig..

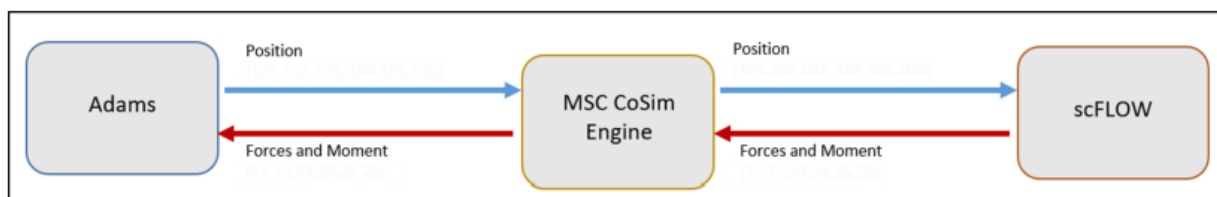


Figure 4.1: Representation of exchange of physical quantities [23]

*CoSim-Pre* simultaneously displays analysis models of solvers in geometry or mesh form, allowing inspection of coupling surfaces and point locations. *CoSim-Launcher* facilitates the execution of co-simulation jobs locally, remotely, or on HPC systems. It utilizes a CMB archive created by *CoSim-Pre*, encompassing all information necessary for linked solvers. The launcher provides status graphs for various solvers, allowing real-time review of co-simulation status.

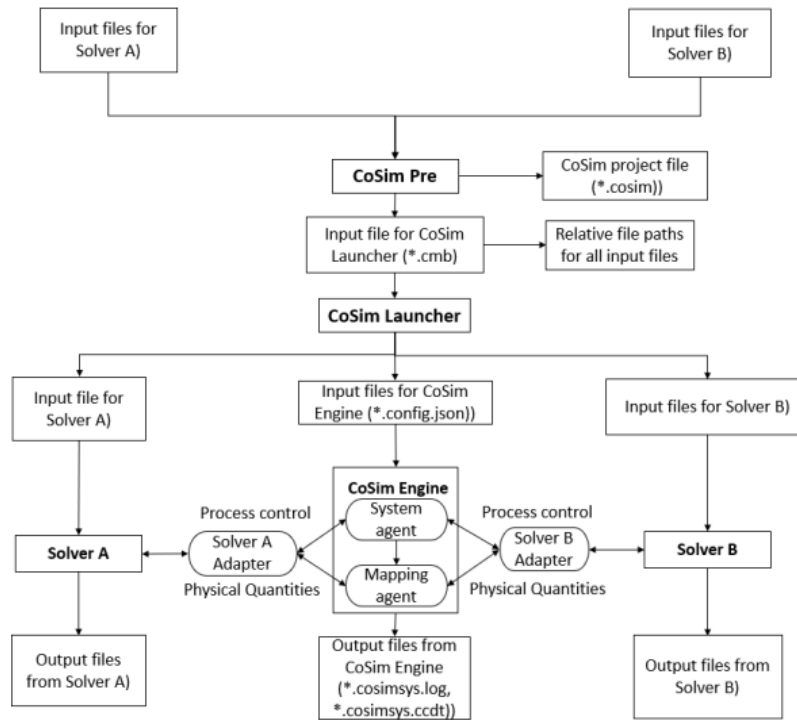


Figure 4.2: Representation of co-simulation workflow [23]

The CoSim engine, functioning within a task-based framework initiated by CoSimLauncher, handles various tasks, including initialization, loop processing for time promotion, and termination procedures. The MSC CoSim engine code, serving as the master code, incorporates a sophisticated control algorithm that facilitates flexible asynchronous communication with slave codes (Adams, scFLOW etc.) during each time integration step. Following successful time integration steps, all slave codes exchange data with the master code. The system is designed to adapt to different time integration steps without a predetermined communication interval. Slave codes, each operating with distinct timesteps, interpolate data based on user-selected "time integration" preferences.

The CoSim engine offers support for a range of algorithms for time interpolation/extrapolation, including quadratic, linear, and constant last. This versatile architecture promotes effective communication and collaboration among the interconnected software codes, allowing for seamless co-simulation across various time integration steps.

The coupled<sup>1</sup> solvers allow the utilization of different time steps, and the CoSim engine provides support for various algorithm types in time interpolation/extrapolation:

- *Quadratic*: This option leverages the historical data of physical quantities (force or displacement) to precisely fit a quadratic curve through the available data points.
- *Linear*: Utilizing a linear fit approach, this option considers the latest two data points for interpolation.

<sup>1</sup>As in [28], there are two primary categories of coupling methods: strong coupling, also known as the direct coupling method, in which coupled equations are directly solved, updating all variables of the interconnected system concurrently; and weak coupling, referred to as the iterative coupling method, that solves each physical model independently, meeting coupling conditions by transferring data between distinct physics models.

- *Constant Last*: This option employs the most recent value in the history of physical quantities for interpolation.

It's noteworthy that the computational meshes of each coupled solver on the connected surface/volume may not align in general. To address this, the CoSim engine offers the following mapping algorithms:

- *Nearest Node Mapping*: This method involves using the three nearest nodes for mapping. The weighting for mapping is calculated based on the area of the triangle formed by the target node and the other two source nodes.
- *Shape Function Based Mapping*: This approach employs shape functions to determine the mapping weight, derived from the connectivity of nodes in the source element.
- *Volumetric Mapping*: This mapping technique directly transfers the values of an elemental variable to the nodes contained in the element without interpolation.

# Chapter 5

## Adams model

The model described in this Chapter, refers to the results obtained in section 9.2.

The first step to perform a simulation is to create a geometry model in *Adams*, including parts, forces and joints. A part is a moving element related to the model having properties of inertia and initial conditions of velocity and position. Different types of parts are available, including rigid bodies, flexible bodies, FE parts and point masses.

A rigid body is an element in the model with mass and inertial properties that remains non-deformable. Introducing a rigid body contributes six degrees of freedom to the overall model. To regulate the motion of these components, constraints like joints can be incorporated to specify their attachment points and determine their relative movement.

Each part has a proper coordinate system, referred to as its local coordinate system. The local coordinate system of a component moves with it, and its initial position defaults to align with the global coordinate system.

The ground part serves as the fixed and stationary element in the model. It serves as the inertial reference frame by default, determining the velocities and accelerations of all other components. It also establishes the global origin (0,0,0) and the reference frame for model construction.

Moreover, it doesn't possess mass properties or initial velocities and doesn't contribute to the model's degrees of freedom.

The valve under study is composed of various parts, as indicated in the figure:

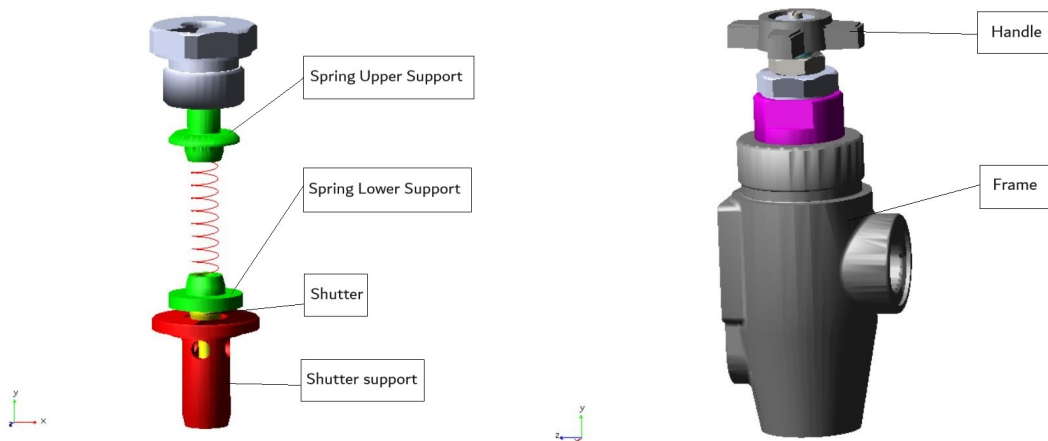


Figure 5.1: Valve main parts

Overall, the valve has a maximum height dimension of 165 mm and a weight of approximately 1.3 kg [2]. In order to meet the total weight requirement, each component has been assigned a density of  $6733.42 \text{ kg/m}^3$ .

## 5.1 Joints

The measure of body movement in relation to other bodies within mechanical systems, known as degrees of freedom (DOF), is fundamental in understanding the system's defining motions. The total degrees of freedom represents the minimum coordinates essential to describe its configuration.

A rigid body in three-dimensional space owns six DOFs. This means that it can move in six independent ways: three translations and three rotations. These DOFs, crucial for defining the system's configuration, serve as the fundamental coordinates.

Representing a mechanical system with more coordinates than degrees of freedom is plausible: this condition necessitates algebraic constraint equations connecting some coordinates (they are not all independent). Various constraints limit specific motion combinations, effectively reducing the model's degrees of freedom.

To define the motion between two components of the model we use joints. They can establish a rigid connection between two parts allowing the following DOF:

Joint name	# DOF	Type of allowed motions
Fixed	0	No movement between parts.
Translational	1	One part translating in relation to another; all axes are aligned.

Table 5.1: Allowed DOF for used joints

In this case, it is first necessary to use a *Fixed joint* to attach the "Shutter Support" to the Ground.

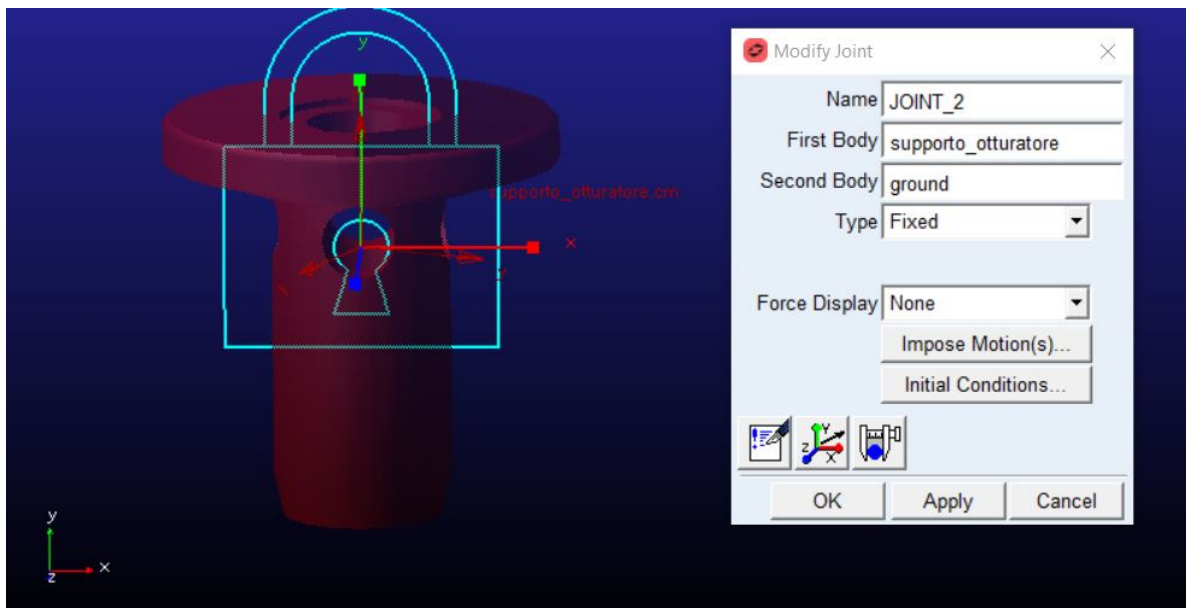


Figure 5.2: Fixed joint for Shutter Support

In the same way, it is necessary to use a *Fixed joint* to attach the "Spring Upper Support" to



the Ground.

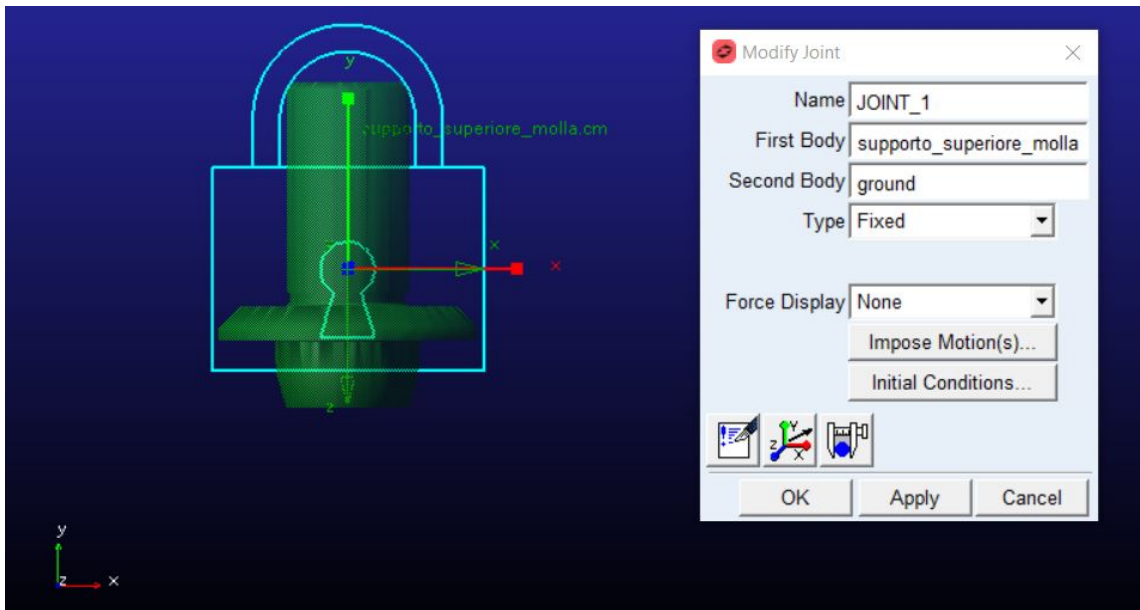


Figure 5.3: Fixed joint for Spring Upper Support

Additionally, since the "Shutter" pushes on the "Spring Lower Support" when the spring compresses, and vice versa the "Spring Lower Support" pushes on the "Shutter" when the spring expands, a *Fixed joint* is used to attach the "Spring Lower Support" to the "Shutter".

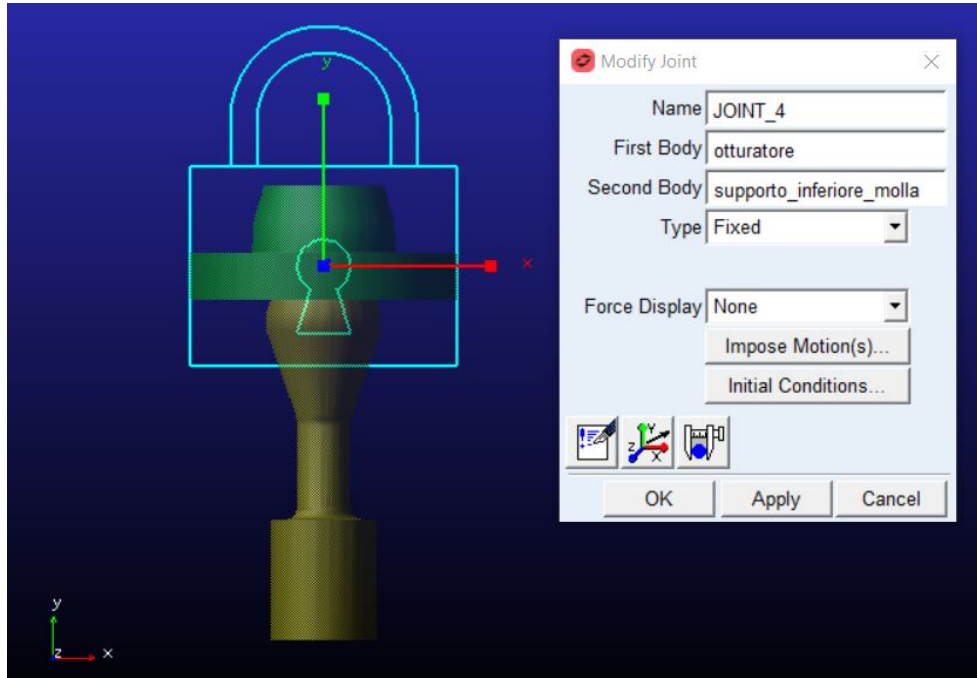


Figure 5.4: Fixed joint for Spring Lower Support and Shutter

Finally, a *Translational joint* is used to model the translation of the "Shutter" relative to the Ground.

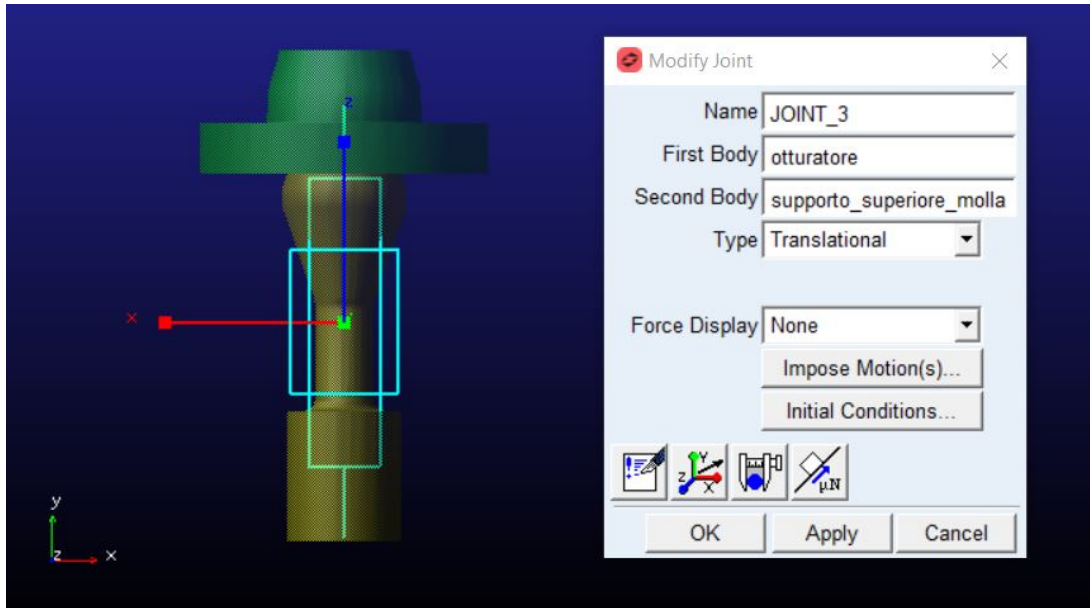


Figure 5.5: Translational joint for Shutter

## 5.2 Translational Spring Damper definition

In order to simulate the presence of a spring between the "Spring Upper Support" and the "Spring Lower Support", a Translational Spring Damper (2.9) is employed.

The preload of the spring is defined as follows [1]:

$$F_0 = \frac{\pi d^2 p^*}{4} \quad (5.1)$$

Where:

- $F_0$  is the preload of the spring;
- $d$  is the seat/damper diameter;
- $p^*$  is the cracking pressure.

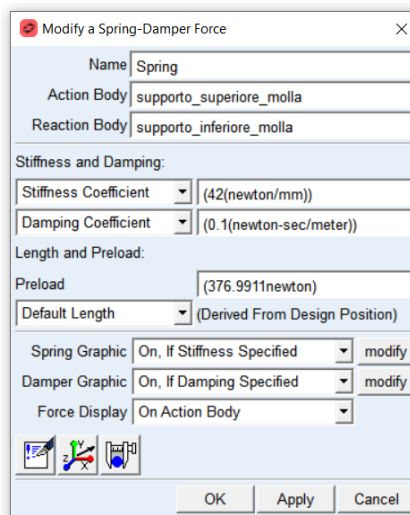


Figure 5.6: Spring parameters

The following values are used:

Variable	Value
$p^*$	75 bar
$d$	8mm

Table 5.2: Main geometric parameters of the valve

It follows that  $F_0 = 376.9911$  (Eq. (5.1)). The Damping is defined of  $0.1 Ns/m$ .

### 5.3 Setting of Shutter end position

In order to model the end position of the Shutter within the valve, a multi-component force (2.7) is applied to Shutter and a Bistop function (2.8) is implemented, as follows:

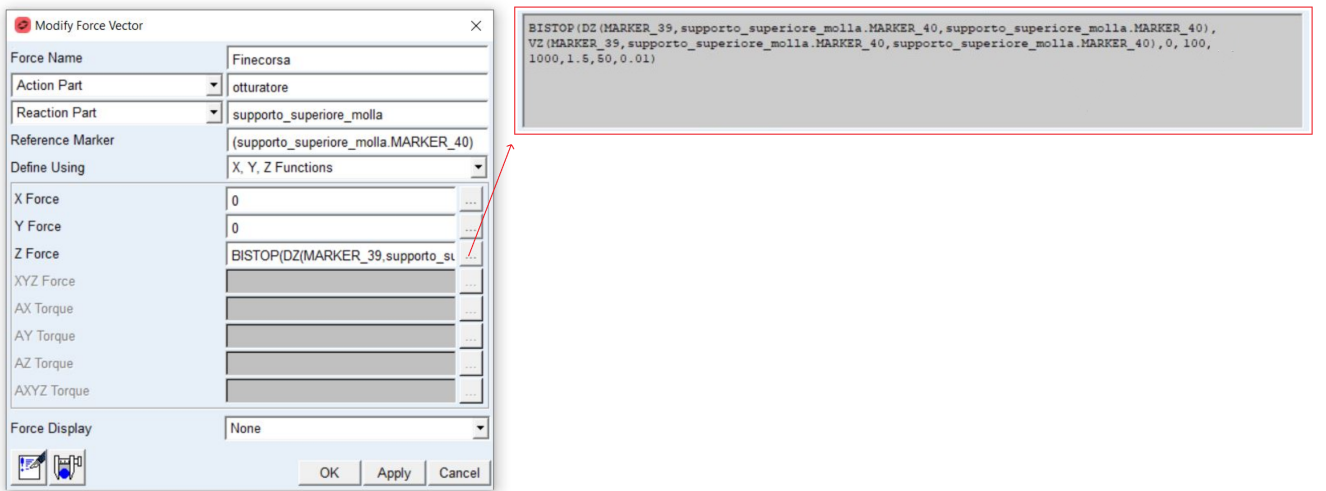


Figure 5.7: Bistop function parameters

The force is directed from the Shutter - the action body - to the Spring Upper Support - the reaction body - along motion direction.

In the force definition, a Bistop function is introduced taking the following parameters as input:

- The  $DZ(i,j,k)$  function provides the z-component of the translational displacement vector from marker j to marker i, expressed in the coordinate system of marker k. If not specified, markers j and k default to the global coordinate system.
  - i: The marker whose origin is considered for the displacement measurement.
  - j: The marker serving as the reference point for displacement calculation.
  - k: The marker whose coordinates define the system for the z-component calculation.
- The  $VZ(i,j,k,l)$  function provides the z-component of the velocity vector difference between marker i and marker j. expressed in the coordinate system of marker k with all vector time derivatives taken in the reference frame of marker l.
  - i: The marker whose origin is considered for the velocity measurement.

- j: The marker relative to which velocity is determined.
  - k: The marker defining the coordinate system for the velocity vector.
  - l: The reference frame for the first time derivative of the displacement vector.
- 0 mm is the lower bound of x;
  - 100 mm is the upper bound of x;
  - 1000 N/mm is the stiffness coefficient;
  - 1.5 is the exponent of the force deformation characteristic;
  - 50 Ns/mm is the maximum damping coefficient;
  - 0.01 mm is the distance at which we decide to apply full damping coefficient.

The lower bound was chosen so that the shutter stops when in contact with its seat; the upper bound so that, since there is no limitation on the upper position that the shutter can reach, it was a sufficiently large value that it is never actually reached. The other parameters used, were chosen as a result of considerations based on experience.

### 5.3.1 Gforce and Dynamic simulation

Finally, in order to carry out the simulation, a Gforce (2.6) is applied to the center of mass of the "Shutter" component. The components of the Gforce are currently empty, awaiting information that will be passed from the co-simulator. The fluid dynamic force calculated using the scFLOW software, which is the force enabling valve movement, thus translates into a displacement of the shutter.

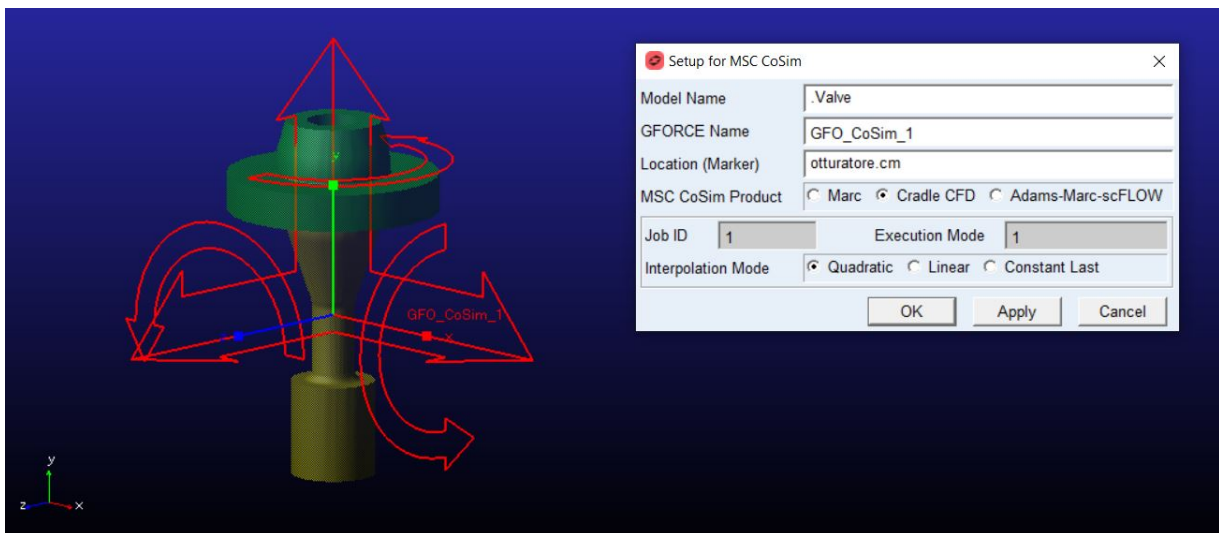


Figure 5.8: Gforce applied to Shutter center of gravity

Finally, a *Dynamic Simulation* (2.4) is performed as follows:

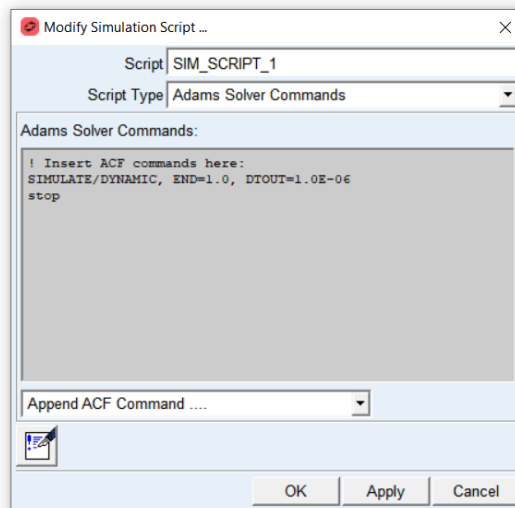


Figure 5.9: Performed dynamic simulation parameters

This configuration means that it will start a dynamic simulation with a range from 0 to 1 seconds and a time step of  $10^{-6}$  s.

# Chapter 6

## scFLOW model

### 6.1 Model set up and definition of the fluid domain

To set up the model in *scFLOW*, the refined geometry of all non-essential geometric elements is imported. During the mesh creation phase, these elements could lead to highly deformed elements or difficulties in adhering to the specified geometry with the predetermined element size.

Furthermore, to simplify co-simulation, the parts undergoing the same movement are unified. In particular, the unification process involves the parts "Spring Lower Support" and "Shutter", as shown in Fig. 6.1 below.

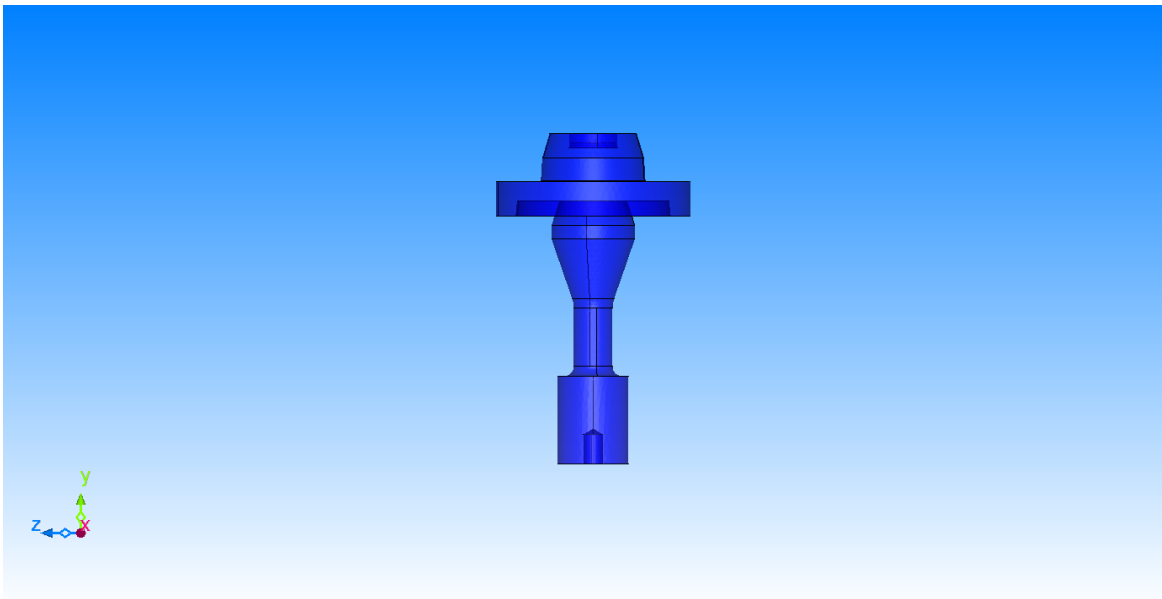


Figure 6.1: Unification of parts Spring Lower Support and Shutter

After importing the CAD model, the next step involves defining the fluid domain. This domain comprises the internal part of the "Frame" between the inlet and the outlet. For defining the fluid domain, we proceed with extracting the empty volume within the existing parts.

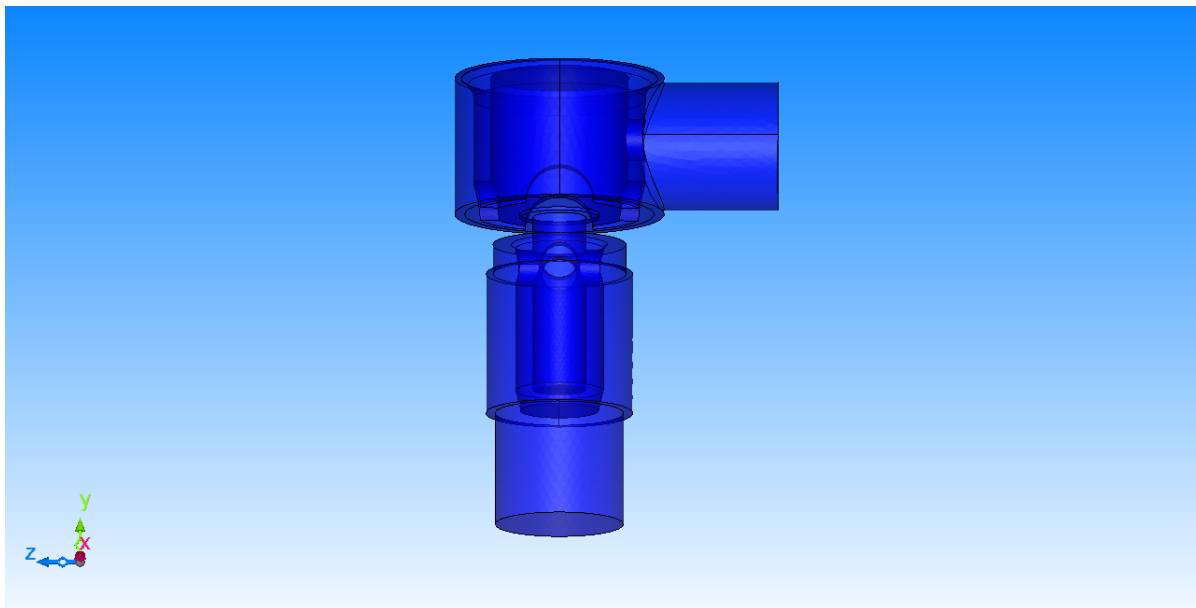


Figure 6.2: Fluid domain

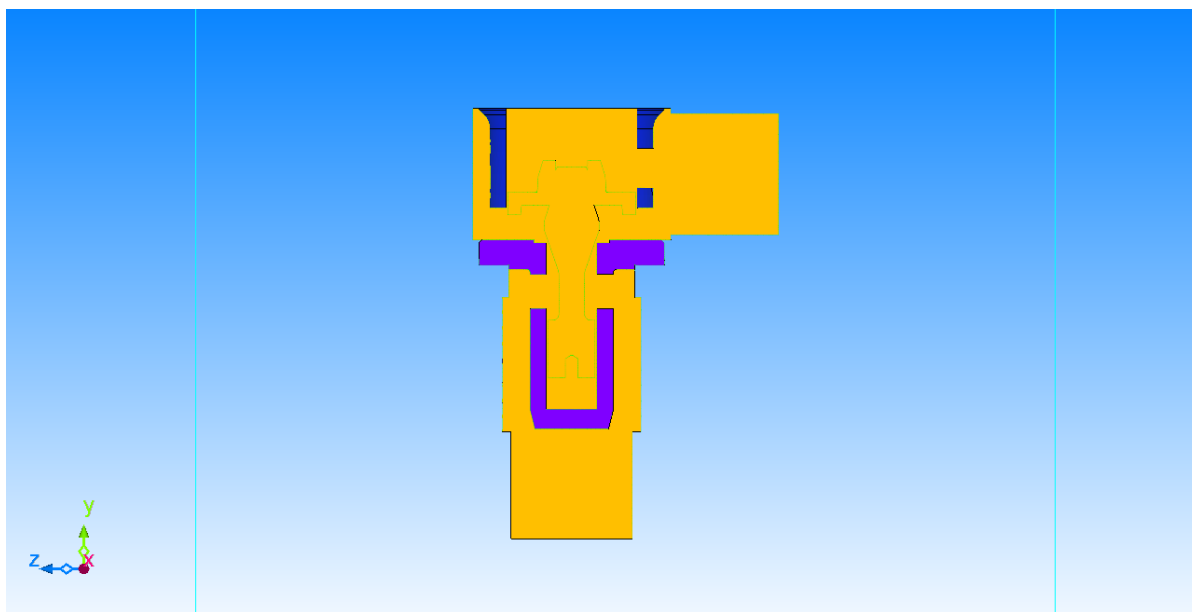


Figure 6.3: Fluid domain

## 6.2 Overset mesh

To allow the movement of the "Shutter" relative to the system and the fluid, it is necessary to define, in addition to the Background mesh, an Overset mesh.

The concept of overset mesh, where multiple meshes overlap, is a solution for handling moving objects.

The overset mesh methodology consists primarily of three sequential steps within its operational framework. Initially, the computation selects the mesh domain (elements) from a collection of overlapping meshes. This selection process, termed *Hole cutting*, involves deactivating the unnecessary mesh domain for computation, treating it as inactive and outside the analysis domain.

Subsequently, the scheme constructs two crucial lists: the *acceptor* elements designated for inter-mesh communication within the active domain and the *donor* elements, serving as counterparts for communication with the acceptor elements. These initial steps constitute the pre-processing phase, setting the stage for the main computation.

During the main computation, coefficients matrices derived from discretizing governing equations on individual meshes are interconnected and solved using the constructed element lists. In scenarios where the computation domain incorporates a moving mesh, the pre-processing steps, namely hole cutting and list construction, recur at each time step.

In order to create the just-defined *Component mesh*, enable both displacements and rotations according to the following scheme:

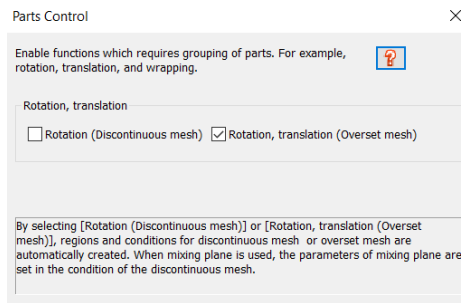


Figure 6.4: Setting for overset mesh

The fluid domain related to the component mesh is obtained by creating an offset of the combined system of the "Shutter" and "Lower Spring Support":

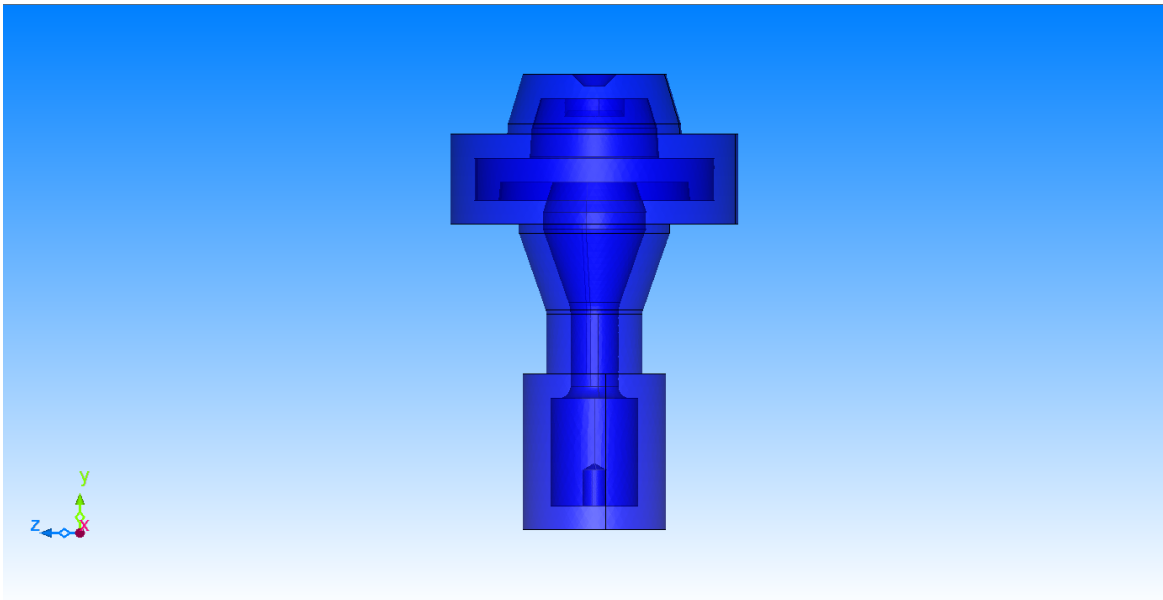


Figure 6.5: Fluid domain for component mesh

When all essential elements are properly defined, we have to distinguish the primary computational domain from those associated with the overset mesh. This crucial step facilitates the utilization of four distinct meshes during the configuration of the mesh.



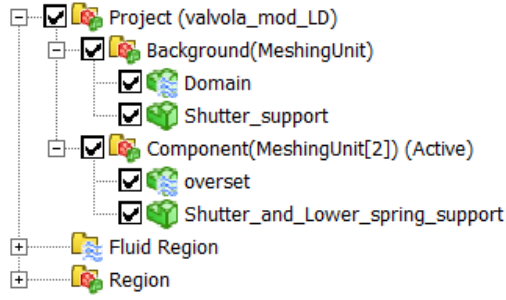


Figure 6.6: Background and component meshing units

## 6.3 Analysis model

In order to proceed with the subsequent operations, the software needs to create an internal model by tessellating the geometry. This operation allows for a geometry check, preventing errors during mesh creation due to poor-quality or negative elements.

### 6.3.1 Part material

There are three classes of materials:

- *Fluid*: represents incompressible or compressible fluids. In the case of compressible flow simulations, it is necessary to activate the energy equation;
- *Obstacle*: objects serving as obstacles or boundaries for the fluid, but are thermally inert;
- *Solid*: represents solid materials with properties such as density, thermal conductivity, specific heat, etc. In thermal exchange simulations, solids actively participate.

In this case, an oil with the following properties is used as the fluid:

Property	Value
density	866 [ $kg/m^3$ ]
dynamic viscosity	0.0415 [ $Pa \cdot s$ ]
bulk modulus	$1.710^9$ [ $Pa$ ]

Table 6.1: Fluid property

All other elements of the domain are set as Obstacles.

### 6.3.2 Regions

Regions are references linked to the geometry to identify surfaces or volumes in the model. They are used to define boundary conditions, local mesh controls, measurement surfaces during post-processing, etc.

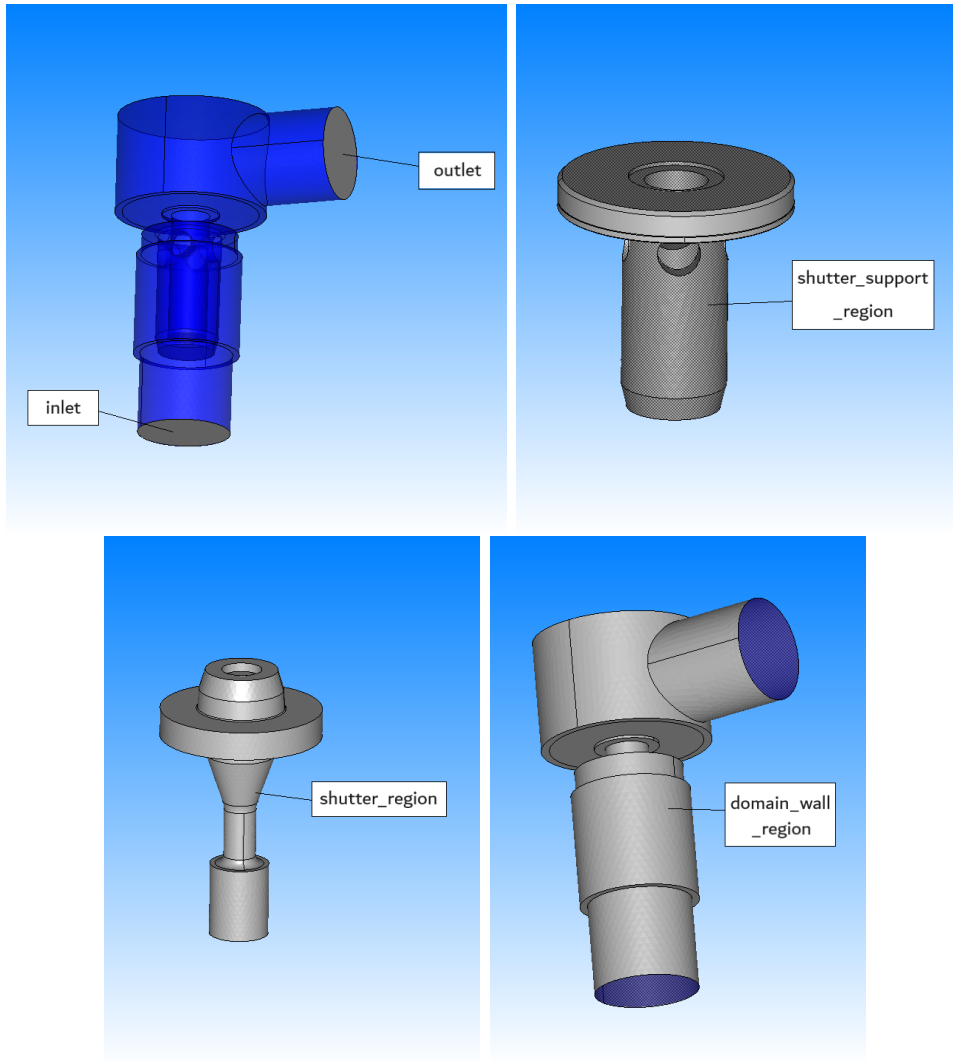


Figure 6.7: Defined regions

A numerical region typically refers to a specific region or domain within the computational domain where certain numerical settings, such as mesh refinement or solver parameters, are applied. It allows users to locally refine the mesh or adjust simulation settings to capture detailed flow behavior in specific areas of interest.

In this case, a numerical region (*vol-ref*) is defined in the area of greatest interest, as follows:

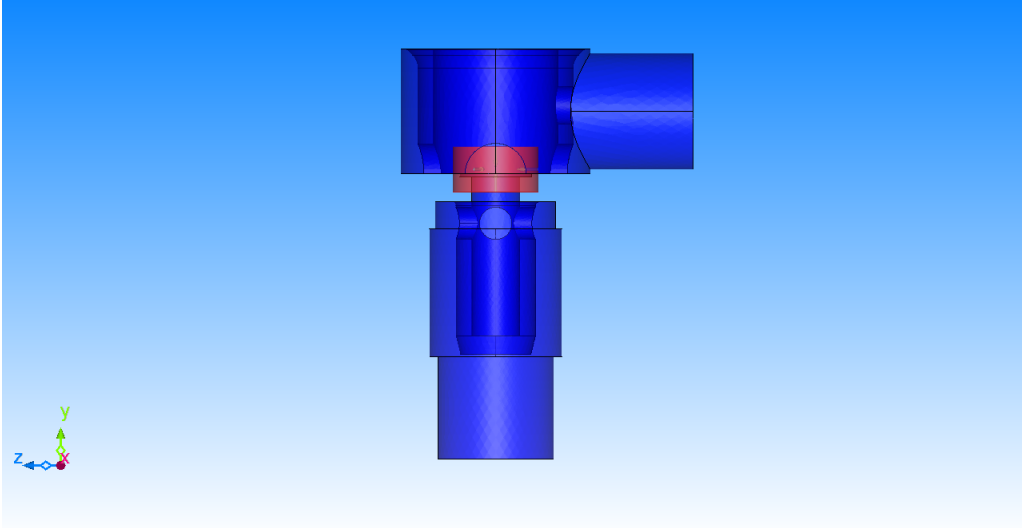


Figure 6.8: Numerical region

## 6.4 Conditions

In this section, we describe the settings used for the simulation and the relevant data for each ones.

A transient<sup>1</sup> analysis is performed, enabling the condition of moving elements, particularly according to an *Adams* coupled law. Additionally, since the fluid is compressible, the energy equation is activated. We use a *Standard*  $k - \varepsilon$  (3.4.1) Model as the turbulence model. The operational time step is  $10^{-6}$  s with a simulation performed for 10000 steps.

### 6.4.1 Flow boundary conditions

Boundary conditions are applied to the faces that separate the inner and outer regions of the analysis domain. It falls into different categories—Dirichlet<sup>2</sup> boundary condition, Neumann<sup>3</sup> boundary condition, and periodic<sup>4</sup> boundary condition—based on how the values are specified. In this case, we define an inlet flow rate condition equal to  $40$  l/min; to prevent the downstream pressure of the valve from unrealistically dropping below absolute zero, a backpressure at the outlet is simulated: an outlet pressure condition equal to  $50$  bar is defined, as follows:

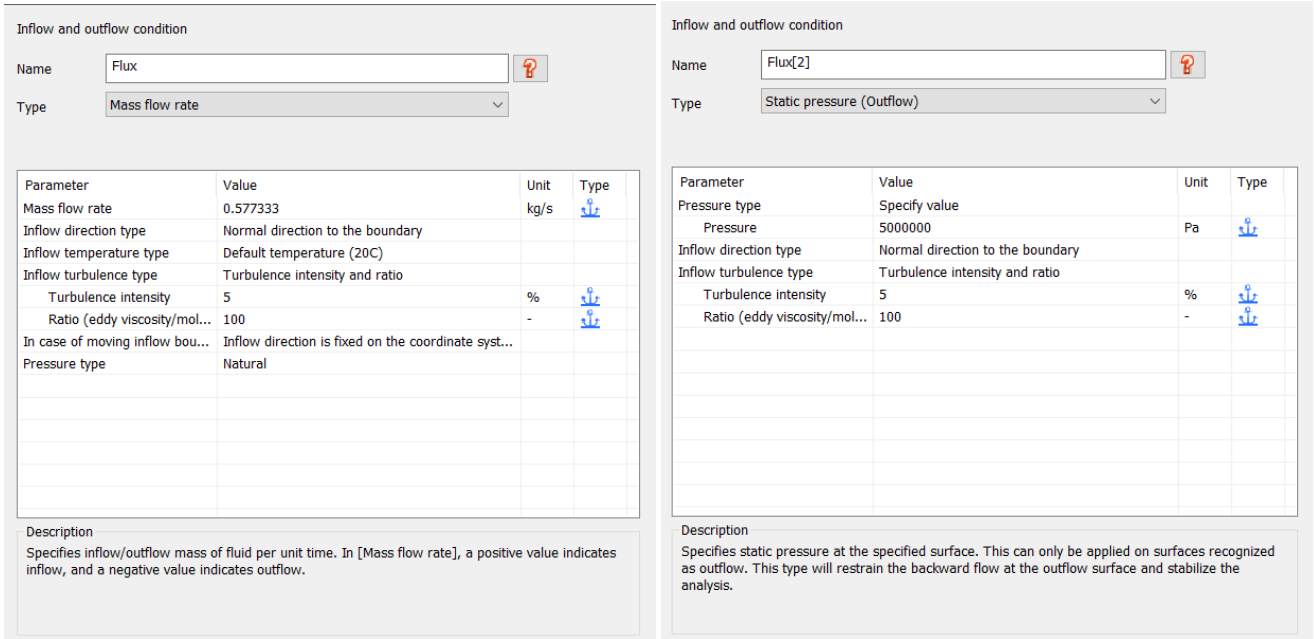
---

<sup>1</sup>Transient analysis refers to the simulation of fluid flow and heat transfer phenomena over time. Unlike steady-state simulations where the solution does not vary with time, transient simulations capture the evolution of the flow and temperature fields over a specified time period.

<sup>2</sup>In Dirichlet boundary conditions, numerical values are directly given to boundary faces.

<sup>3</sup>In Neumann boundary conditions, we give to boundary faces the gradient of a variable.

<sup>4</sup>In Periodic boundary conditions two boundary faces are treated as a single entity because the analysis target has a periodic geometry.



(a) Inlet condition

(b) Outlet condition

Figure 6.9: Inlet and outlet conditions

### 6.4.2 Wall boundary conditions

Fluid-solid or external wall interfaces induce shear stress on the fluids, necessitating specific wall conditions for shear stress. More precisely, these conditions involve determining the frictional force exerted by the fluid on the walls. Two commonly employed assumptions for these wall conditions are the free-slip and no-slip conditions.

Under the free-slip wall condition, there is no friction between the fluid and the walls, yet the flow direction is influenced by the walls. Although the flow is directed by the walls, the kinetic energy remains unchanged, maintaining a zero-gradient of momentum on the boundaries. This condition is applicable in scenarios involving flows with minimal viscosity and for the sides of the pressure drop region not to have unnecessary frictions.

The no-slip condition allows for friction. Initially assuming zero velocity at the wall, the friction force is then calculated based on the velocity gradient near the wall. This results in the existence of a momentum boundary layer on the wall. The actual friction force depends on flow conditions within the boundary layer, particularly in turbulent boundary layers where the velocity gradient undergoes complex changes.

In this case, referring to regions of Section 6.3.2, we set up the following conditions:

Region	Boundary condition
Domain_wall_region	No-slip
Shutter_region	No-slip
Shutter_support_region	No-slip

Table 6.2: Boundary conditions

### 6.4.3 Moving elements

In conclusion, it is necessary to link the two models from *Adams* and *scFLOW* by establishing how the communication between pressure forces and displacements occurs.

Each element, treated as a surface, is assigned a "Mechanism coupled (Adams)" movement type. This entails the insertion of the corresponding element ID, an output from the Gforce created in *Adams*. This process is reiterated for each mobile element, assigning a unique ID and associating it with the reference surface on scFLOW, corresponding to the *Adams* component where the GForce originated.

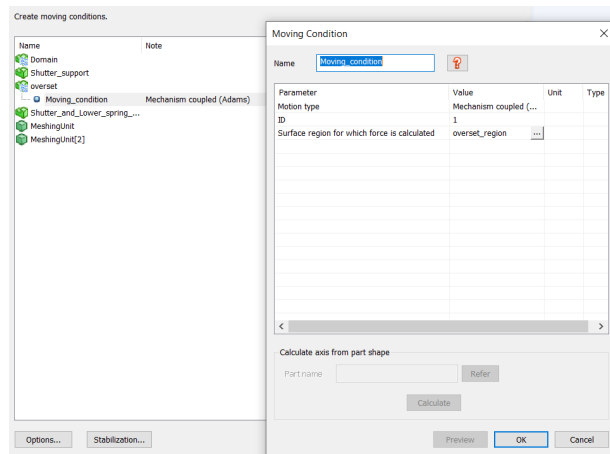


Figure 6.10: Setting of moving condition

Ensuring the seamless integration of calculations mandates uniformity in the units of measurement and the orientation of the reference system across both codes.

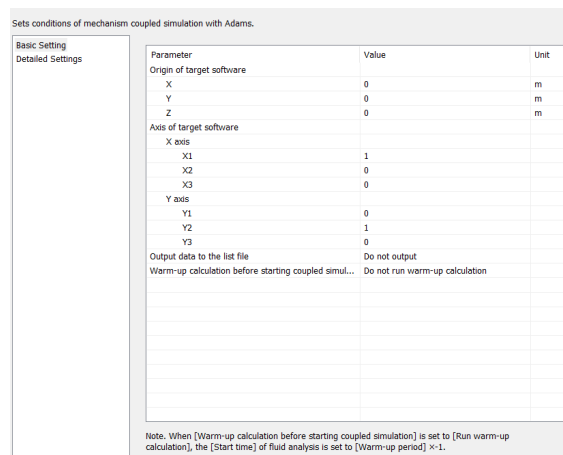
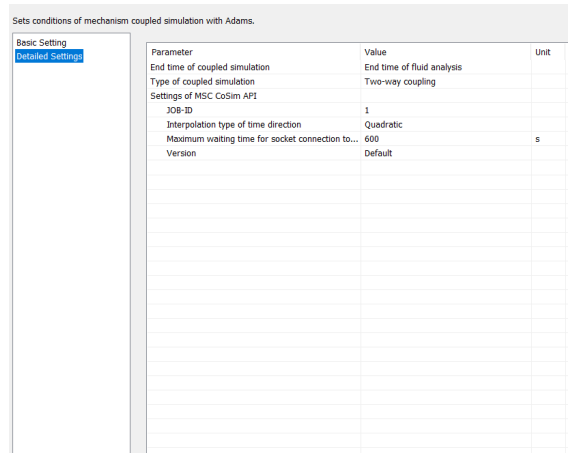


Figure 6.11: Orientation of the reference system

Prior to delving into mesh-related tasks, the co-simulation type was chosen. Opting for a two-way coupling in the current analysis facilitates bilateral communication, concluding data exchange at the conclusion of the fluid dynamic simulation. Finally, a quadratic interpolation of time direction was chosen.



Parameter	Value	Unit
End time of coupled simulation	End time of fluid analysis	
Type of coupled simulation	Two-way coupling	
Settings of MSC CoSim API		
JOB-ID	1	
Interpolation type of time direction	Quadratic	
Maximum waiting time for socket connection to...	600	s
Version	Default	

Figure 6.12: Quadratic interpolation

## 6.5 Octree and Mesh

In scFLOW, unstructured<sup>5</sup> mesh elements are generated by forming clusters of cuboids using the octree method.

Octree is an approach to partitioning three-dimensional space, involving the repeated division of space into eight sections known as octants. When implementing octree division, a cuboid undergoes successive divisions at the center of each side. This process creates a hierarchical structure where each octant further refines the spatial subdivision (Fig. 6.13).

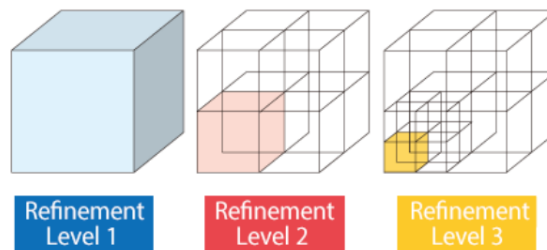


Figure 6.13: Refinement levels for octree generation [22]

The creation of the mesh is thus divided into two distinct phases:

1. Octree: The body is enclosed in a Root Octant and successively divided into octants until a specified minimum size is reached. This process serves to size the mesh, specifying mesh resolution to determine the scale and accuracy of the analysis.
2. Polyhedral mesh: Create a polyhedral mesh with prism layer elements to be used for the analysis.

To define the size of the Octants, there are three types of controls:

- Target a number of elements and let the octree generator decide for the sizes.

<sup>5</sup>An unstructured mesh is characterized by irregularly aligned mesh elements, which can be triangles or rectangles (for two-dimensional applications) or polyhedral elements (for three-dimensional applications).

- Determine the sizes with a minimum value which will be applied to the surfaces; then, the octree is coarsened.
- Control the octree with detailed parameters and sizes on surfaces and volumes.

It is also known that the grid needs to be refined under the following conditions:

- Strong pressure or velocity gradients;
- Curved shape surfaces;
- Small edges;
- Moving regions.

Additionally, in scFLOW, there is a function known as *influence range*, which allows specifying how the transition from the finer to coarser grid occurs. Values of *influence range* greater than 0 imply that the specified cell size at the wall is maintained for a certain number of layers within the domain, starting from the wall.

For the definition of the mesh, particular attention is placed on the first element at the wall to describe the boundary layer accurately. For this reason, various layers of prisms are used. In fact, a uniform distance from the wall allows for an accurate calculation of velocity and temperature gradients, ensuring stability in the calculations.

Since each turbulence model works well with a specific minimum distance to the wall and consequently precise values of  $y^+$ , reference is made to the following values:

Model	$y^+$
SST $k - \omega$	1 - 5
$k - \varepsilon$	10 - 100

Table 6.3:  $y^+$  values with reference to used models

Considering the working fluid, its properties, and the conditions of the flow field, we can calculate:

$$h = \frac{y^+ \nu}{u^*}$$

Where:

$$u^* = \sqrt{\frac{\tau_w}{\rho}}, \quad \tau_w = \frac{1}{2} C_f \rho U_\infty^2, \quad C_f = 0.0576 Re_x^{-1/5}, \quad Re_x = \frac{\rho U_\infty D}{\mu}$$

Where:

- $h$ : dimension of the first octant at the wall;
- $y^+ = y/l_\tau$  ( $l_\tau$  characteristic length of the boundary layer);
- $u^* = u/U_\tau$  ( $U_\tau$  characteristic velocity of the boundary layer);
- $\rho$ : density;
- $\tau_w$ : viscous stress at the wall;

- $\nu$ : kinematic viscosity;
- $\mu$ : dynamic viscosity;
- $D$ : characteristic size of the body (diameter);

Defined the size of the first wall prism, the subsequent elements are defined as follows:

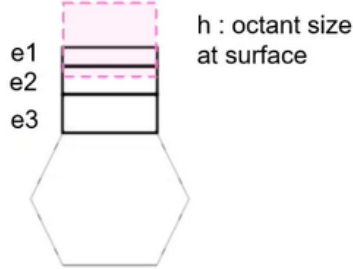


Figure 6.14: Prism growing on wall [22]

The variation rate of thickness is defined, referring to Fig. 6.14, as:

$$\frac{e_2}{e_1} = \frac{e_3}{e_2}$$

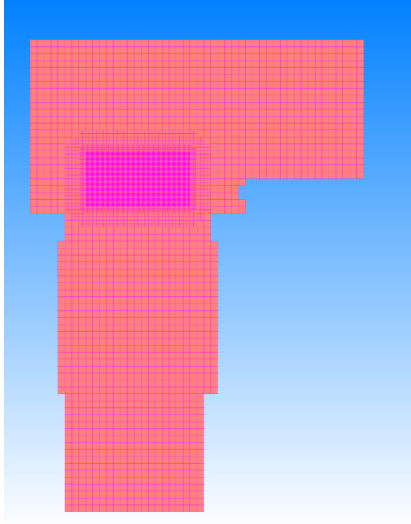
Finally, the following configurations are available for mesh creation:

- Model shape oriented: attempts to replicate the shape and peculiarities of the geometry as accurately as possible; this implies that the size of the octree must represent the detail well: geometry is clean and easy, and the octant is well defined.
- Stability oriented: geometry is not so clean and complex, and the octant may not be well defined.
- Detailed settings: allow you to control all parameters such as surface mesh, timing of prism layer insertion, volume mesh, element size, etc.

### 6.5.1 Background octree and mesh

In this case, a choice is made to create a uniform grid by specifying the dimensions of the elements: in particular, global controls are defined for the entire domain, and local controls refine the grid locally according to the following scheme:





(a) Background octree

Volume region	Dimension [m]
<i>Domain</i>	0.001
<i>vol-ref</i>	0.0001

(b) Grid dimension for each region

Figure 6.15: Background octree definition

To create the mesh, the following parameters are chosen:

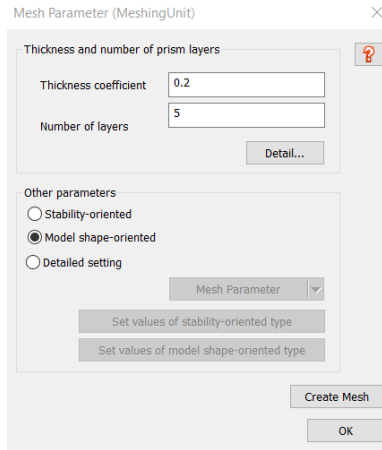


Figure 6.16: Background mesh definition

As a result:

$$e_3 = h \times 0.2$$

Furthermore, the parameters through which to define the mesh can be further detailed. In particular, the following are chosen:

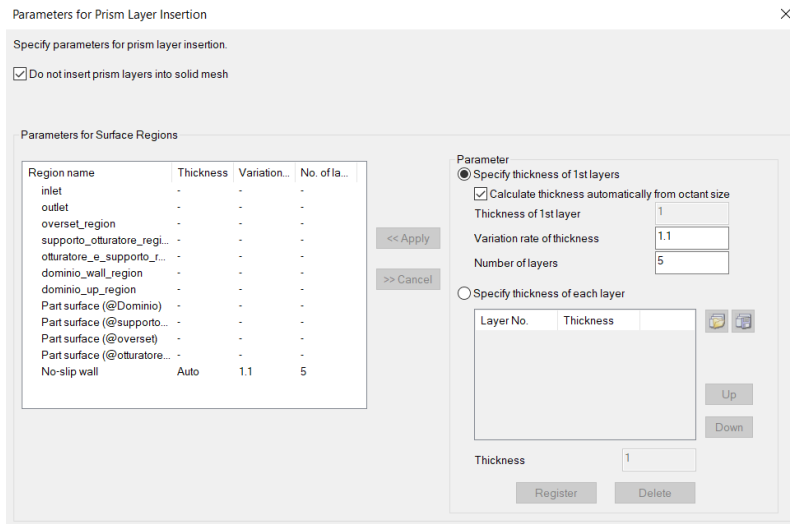


Figure 6.17: Background mesh details

In this case, the size of the first wall prism is automatically calculated by the software, and a variation rate of thickness of 1.1 is applied.

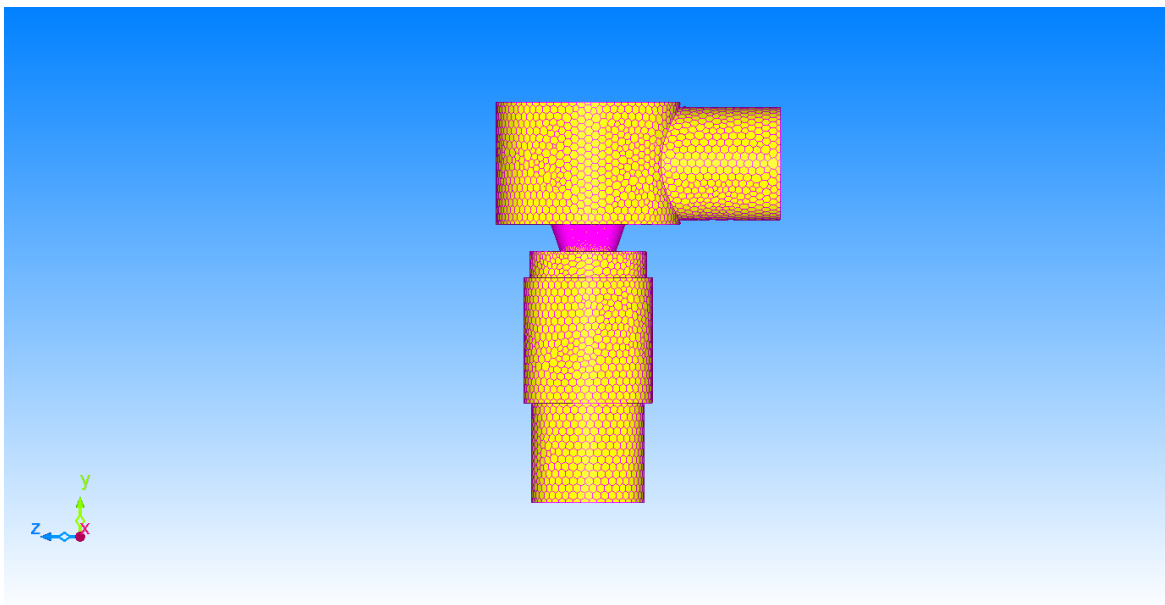
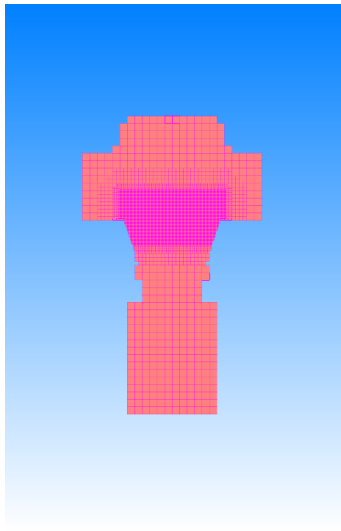


Figure 6.18: Background mesh

### 6.5.2 Overset octree and mesh

As defined previously for the background mesh (6.5.1), in this case as well we create a uniform grid by specifying the dimensions of the elements:



(a) Overset octree

Volume region	Dimension [m]
<i>Domain</i>	0.001
<i>vol-ref</i>	0.0001

(b) Grid dimension for each region

Figure 6.19: Overset octree definition

To create the mesh, the following parameters are chosen:

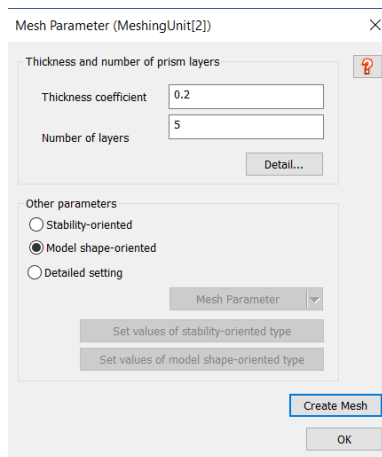


Figure 6.20: Overset mesh definition

Furthermore, the parameters through which to define the mesh are further detailed:

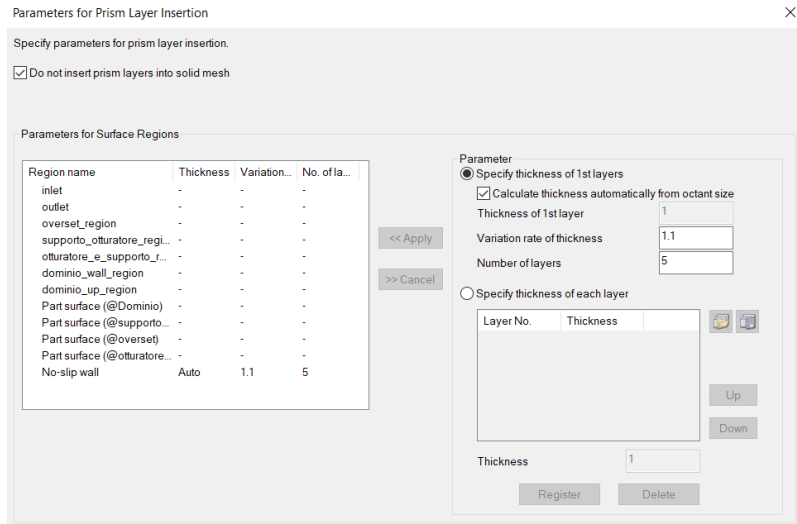


Figure 6.21: Overset mesh details

In this case as well, the size of the first wall prism is automatically calculated by the software, and a variation rate of thickness of 1.1 is applied.

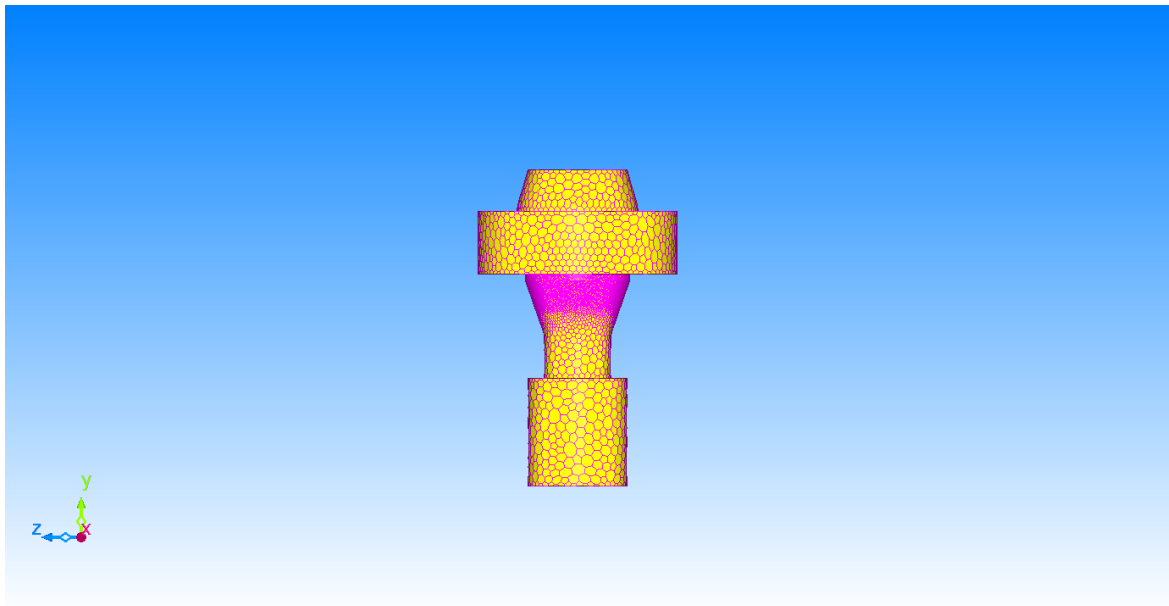


Figure 6.22: Overset mesh

# Chapter 7

## Co-simulation model

When setting up a co-simulation, each GForce created is assigned an ID, which is appended to its name. It is crucial to ensure that the IDs for individual GForces are distinct to facilitate proper integration into ScFLOW. The Job ID is specified to correspond to the ScFLOW Job ID, particularly useful in scenarios involving co-simulation between one Adams process and multiple scFLOW processes. This configuration helps instruct MSC CoSim about the correspondence between GForces in the model and their interaction with specific ScFlow processes.

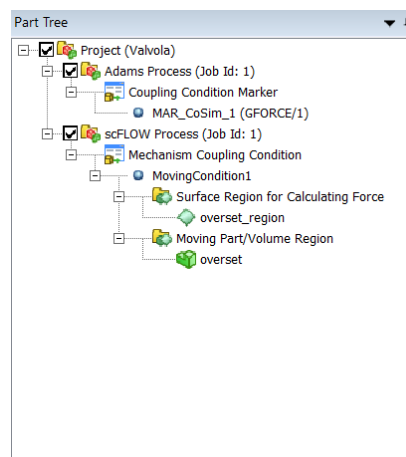


Figure 7.1: Adams and scFLOW configurations

To initiate the co-simulation setup, it's important to set up the Co-simulation Conditions, triggering the display of the Adams-scFLOW Co-simulation-specific Condition Wizard. In particular, we have to create coupling pairs choosing components from both the *Adams* GForce marker and scFLOW Moving condition.

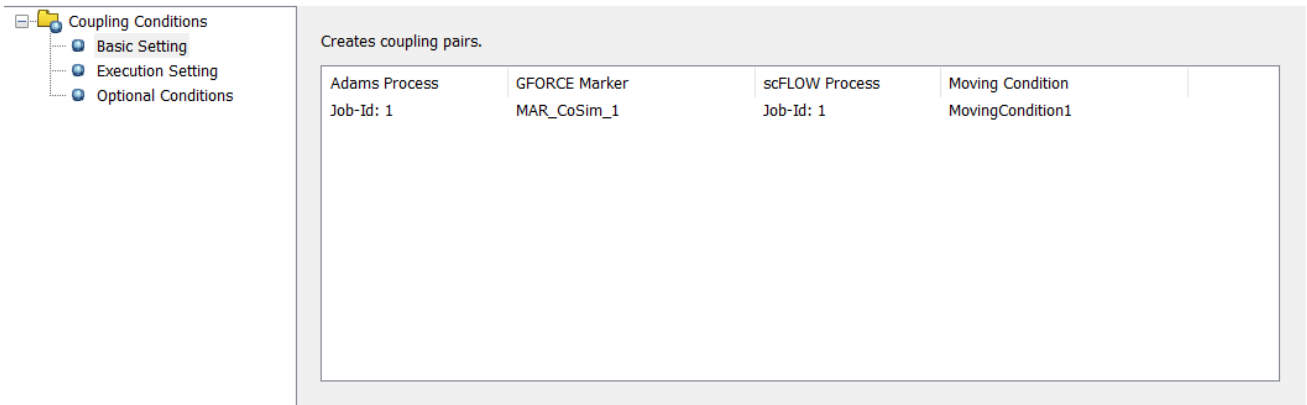


Figure 7.2: Coupling pairs of Adams GForce marker and scFLOW Moving condition

An essential step involves configuring the Execution Setting to optimize co-simulation functionality. This setting dictates the sequence in which participating processes proceed in the co-simulation. Notably, the software with a higher execution mode number takes precedence. If the mode number is set to 0, the application software pauses upon detecting the termination of the communication partner software. Conversely, with a mode number greater than zero, the application software continues its process until completion, utilizing extrapolated data from the MSC CoSim process, even if the communication partner has concluded.

Process	Parameter	Value
Adams Process (Job Id: 1)	Priority for execution order	Default(0)
scFLOW Process (Job Id: 1)	Interpolation type of time direction	Default(Constant last)

(a) Adams settings

Process	Parameter	Value
Adams Process (Job Id: 1)	Priority for execution order	Default(0)
scFLOW Process (Job Id: 1)	Interpolation type of time direction	Default(Quadratic)
	Consideration of virtual mass	Default(Do not consider)

(b) scFLOW settings

Figure 7.3: Cosimulation settings

## Chapter 8

# Grid independence analysis

Determining the optimal grid refinement point involves balancing computational efficiency, time constraints, and maintaining acceptable accuracy in CFD predictions within defined tolerance levels. This problem is addressed through a grid convergence study or mesh refinement study. This process entails generating a sequence of grids for the geometry under consideration, conducting CFD computations, and analyzing result variations across different grid levels, including coarse, medium, and fine resolutions.

The grid convergence study operates on the fundamental principle that increasing grid refinement leads to a gradual reduction in spatial discretization errors, eventually approaching zero and yielding a grid-independent solution. This means the solution becomes independent of mesh resolution, with no further refinement offering improvement.

In the mesh refinement study, systematic grid enhancements are performed across the entire computational domain with each successive level in the grid hierarchy. Surface and volume mesh density are methodically increased: traditionally, surface elements double in number, while volume cell count triples with each subsequent grid. In structured grids, this equates to a 1.5 times increase in points along each coordinate direction. The discretization error on the finest grid should be smaller than the required accuracy.

A grid independence analysis was performed through steady-state simulations [1], maintaining a constant poppet lift at 1 mm and a flow rate of 50 L/min, to verify grid convergence.

After importing the geometry, we proceed with defining the domain regions (6.3.2) and boundary conditions (6.4.1, 6.4.2) as detailed in Chapter 6. This procedure was conducted with a constant grid size across the entire domain and employed four different grid models in the *vol-ref* region (Figure 6.8).

The four grid models, labeled *coarse*, *mid*, *fine* and *extra fine*, detailed in Table 8.1, were such that the *fine* grid type was used for the simulation (6.5).

	COARSE			MID		
Basic settings	Min size 0.001	Max size 0.001 m		Min size 0.001	Max size 0.001 m	
Detail	Region <i>vol-ref</i>	Size 0.001 m	I.R. 0	Region <i>vol-ref</i>	Size 2.5e-4 m	I.R. 0

	FINE			EXTRA FINE		
Basic settings	Min size 0.001	Max size 0.001 m		Min size 0.001	Max size 0.001 m	
Detail	Region <i>vol-ref</i>	Size 1e-4 m	I.R. 0	Region <i>vol-ref</i>	Size 5e-5 m	I.R. 0

Table 8.1: Coarse, mid, fine and extra fine grids

The simulations presented in this study were conducted using the RANS models Standard  $k - \varepsilon$ . To generate the mesh, we selected the option by which software automatically calculates the thickness of the first layer of wall elements based on the octant size. In general, it is crucial to provide an initial estimate of the  $y^+$  value near the solid wall when generating the mesh. This ensures adequate resolution of the boundary layer according to the requirements of the chosen turbulence model. Subsequently, after making an initial estimate of  $y^+$  at 1, a verification is performed to calculate the thickness of the first layer of wall elements in order to achieve the desired  $y^+$  value. This calculation is facilitated using a tool available on the website [15]. An increment of 10% was applied to the thickness of the upper prism layers to obtain the necessary number of prism layers to reach the set element size for the external flow.

	Thickness of 1st layer	Variation rate of thickness	Number of layers
Coarse	Auto	1.1	5
Mid	Auto	1.1	5
Fine	Auto	1.1	5
Extra fine	Auto	1.1	5

Table 8.2: Discretization of the boundary layer

Following grid generation, simulations are conducted on each grid, and flow field parameters of interest are plotted against grid size.

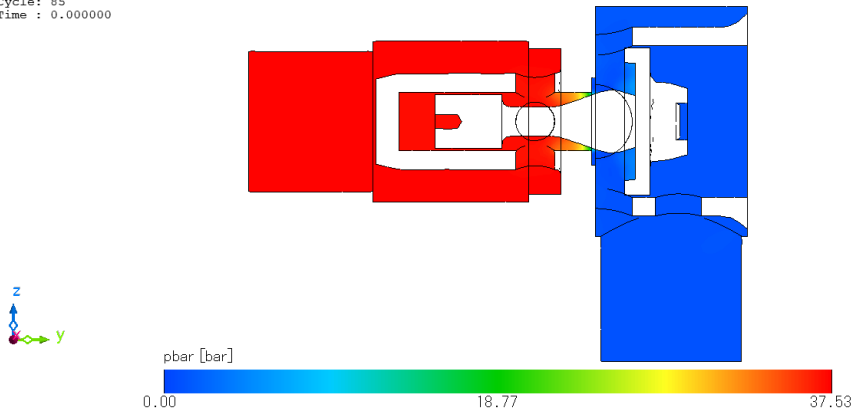
As in [1], the pressure at the inlet in the computational domain is chosen as the convergence criterion for the grid. The measurements of the variable are obtained by performing a scalar integration<sup>1</sup> on the inlet section, which yields the average pressure over the selected area. The obtained results are presented below:

---

<sup>1</sup>A scalar integration function calculates the total value of a scalar property, such as temperature or pressure, over a specific area. It is often used to compute averages for scalar quantities across specific regions.

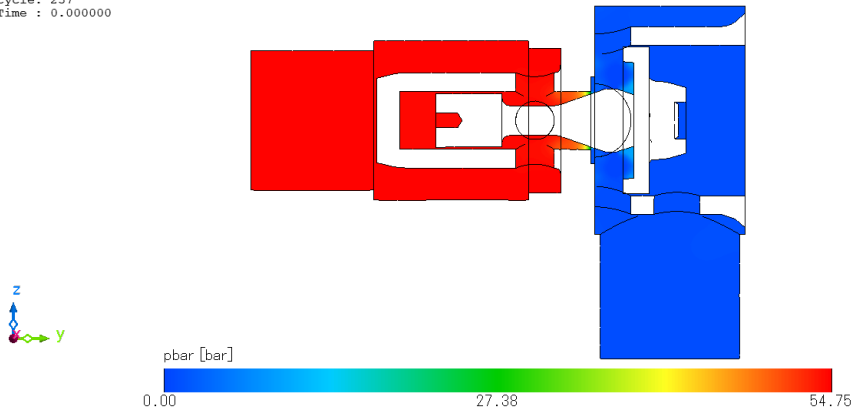


File : valvola\_mod\_LD\_85.fph  
Cycle: 85  
Time : 0.000000



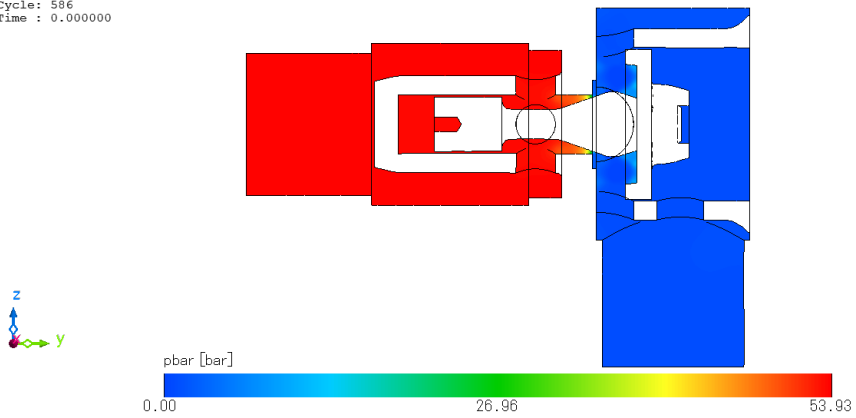
(a) Pressure field for coarse grid

File : valvola\_mod\_LD\_257.fph  
Cycle: 257  
Time : 0.000000



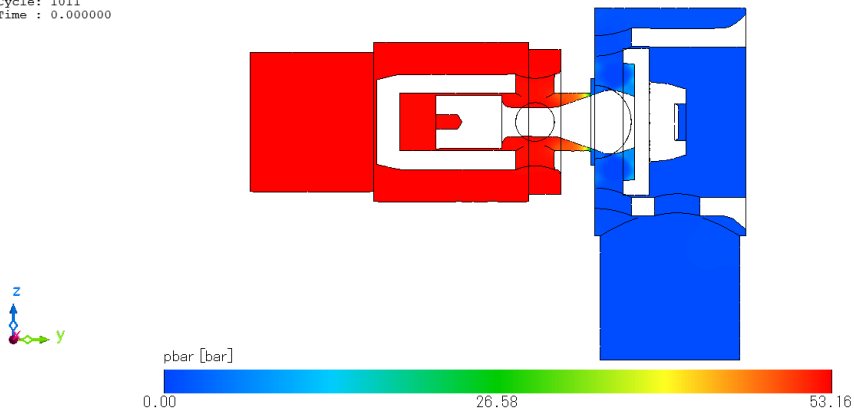
(b) Pressure field for mid grid

File : valvola\_mod\_LD\_586.fph  
Cycle: 586  
Time : 0.000000



(c) Pressure field for fine grid

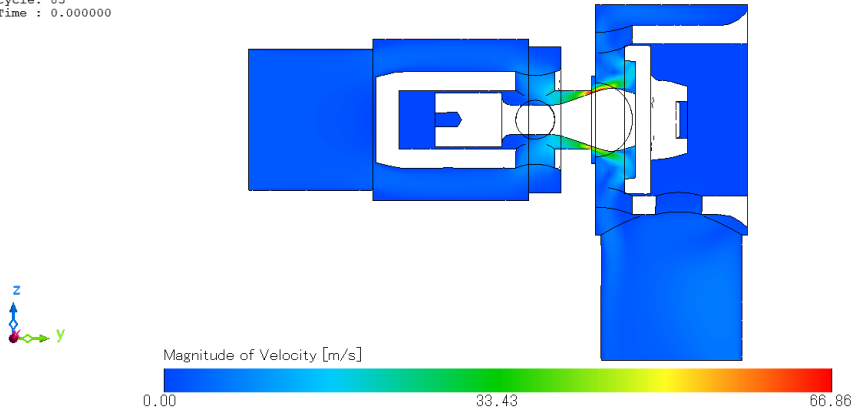
File : valvola\_mod\_LD\_1011.fph  
Cycle: 1011  
Time : 0.000000



(d) Pressure field for extra fine grid

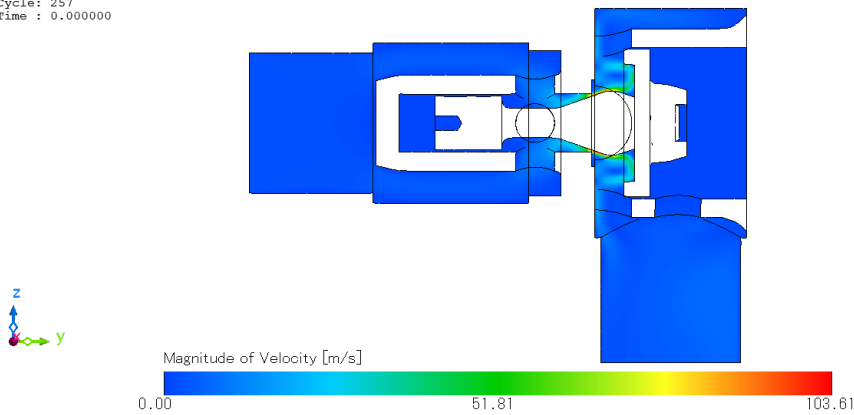
Figure 8.1: Pressure fields

File : valvola\_mod\_LD\_85.fph  
Cycle: 85  
Time : 0.000000



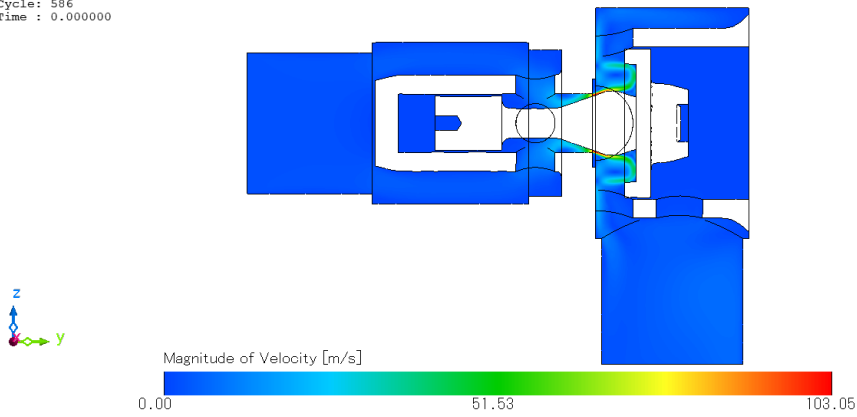
(a) Velocity field for coarse grid

File : valvola\_mod\_LD\_257.fph  
Cycle: 257  
Time : 0.000000



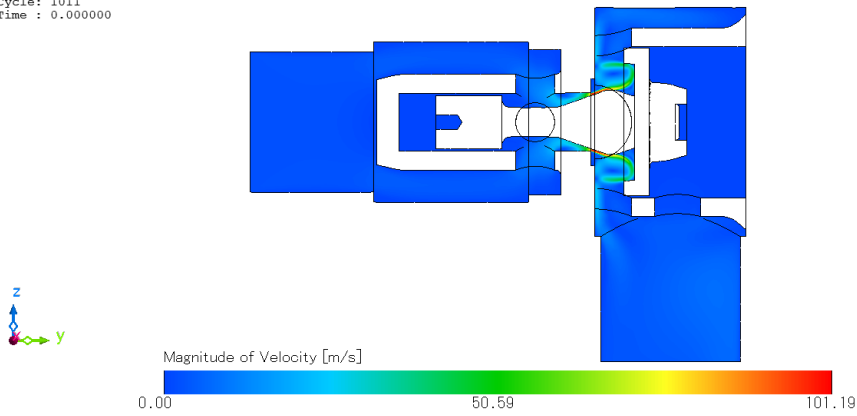
(b) Velocity field for mid grid

File : valvola\_mod\_LD\_586.fph  
 Cycle: 586  
 Time : 0.000000



(c) Velocity field for fine grid

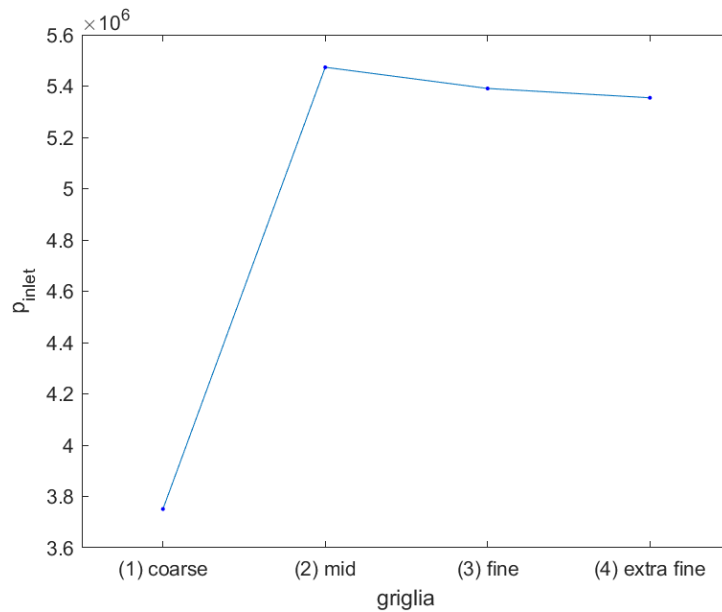
File : valvola\_mod\_LD\_1011.fph  
 Cycle: 1011  
 Time : 0.000000



(d) Velocity field for extra fine grid

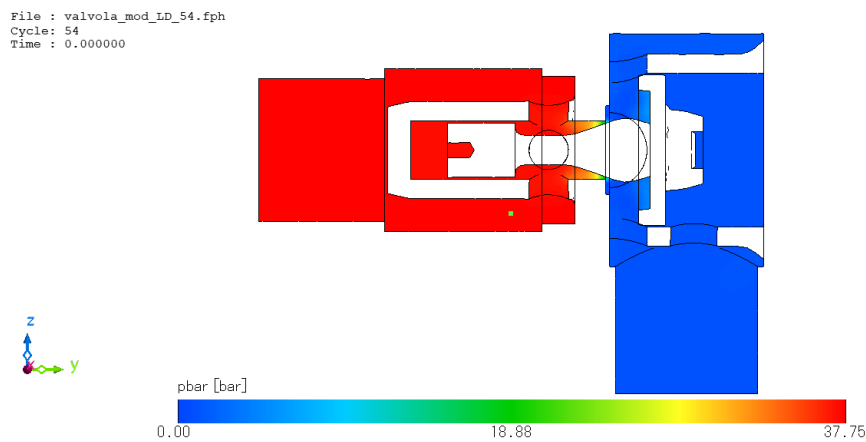
Figure 8.2: Velocity fields

Typically, there's a significant change in values from smallest to coarse grids. Starting from the fine grid, the change in solution becomes minimal, approaching a grid-independent solution. From the plotted graph, the smallest grid yielding a grid-independent solution for routine production runs is selected the optimal grid offering the right solution with minimal solver runtime.

Figure 8.3: Grid independence study with  $k - \varepsilon$  model

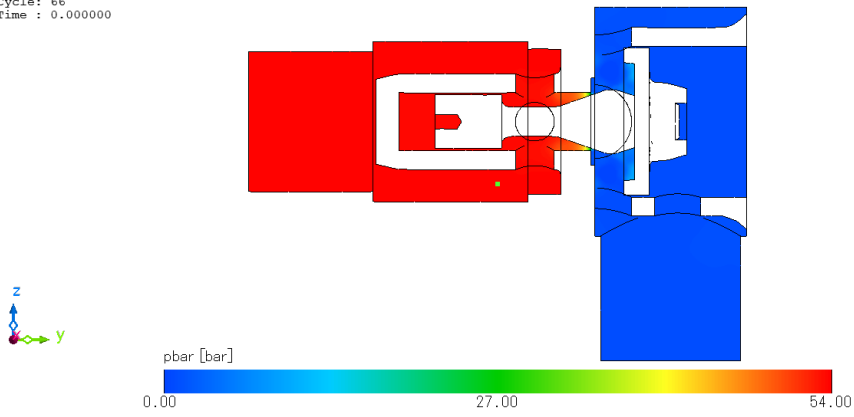
For a more accurate analysis, the same analyses are repeated using the  $k - \omega$  method, keeping the same grid and mesh configuration presented above (Table 8.1 and Table 8.2).

In this case, it was set as a convergence criterion that the physical quantities had an error of less than  $10^{-3}$ , unlike the previous case ( $10^{-4}$ ) since an initial exploratory analysis shows that it is not possible to reach this value below 5000 iterations, which is considered acceptable at this stage.



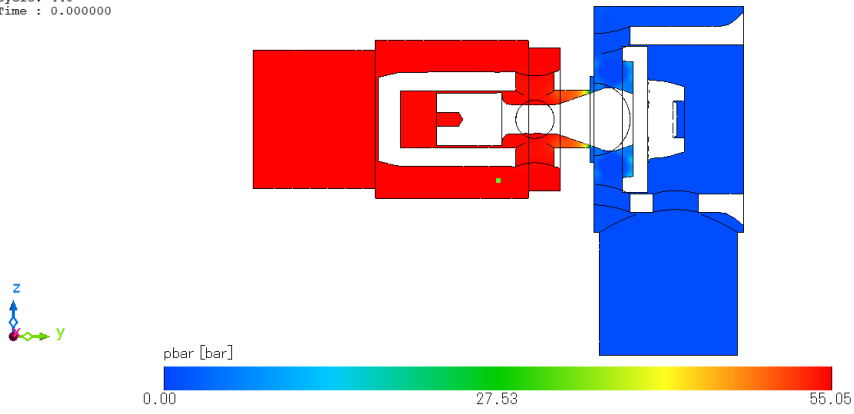
(a) Pressure field for coarse grid

File : valvola\_mod\_LD\_66.fph  
Cycle: 66  
Time : 0.000000



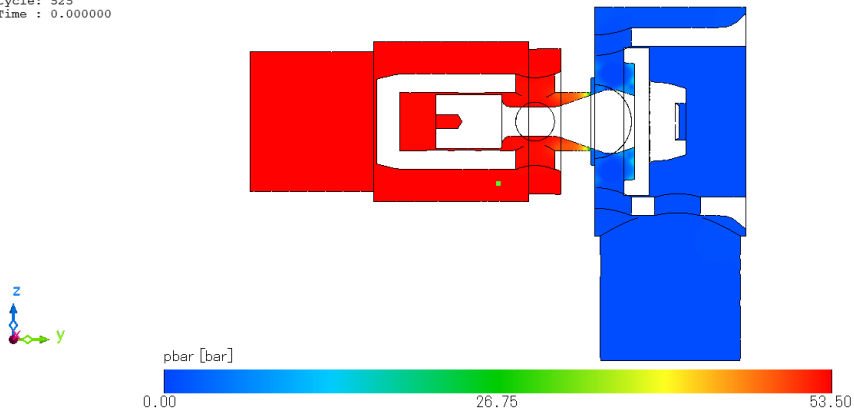
(b) Pressure field for mid grid

File : valvola\_mod\_LD\_440.fph  
Cycle: 440  
Time : 0.000000



(c) Pressure field for fine grid

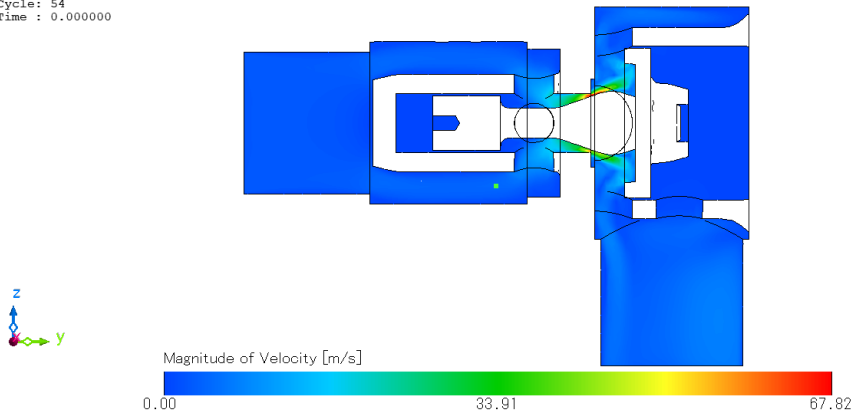
File : valvola\_mod\_LD\_525.fph  
Cycle: 525  
Time : 0.000000



(d) Pressure field for extra fine grid

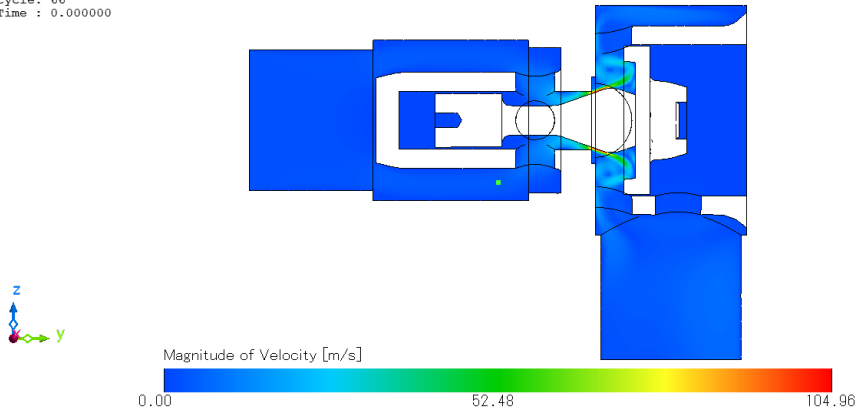
Figure 8.4: Pressure fields

File : valvola\_mod\_LD\_54.fph  
Cycle: 54  
Time : 0.000000



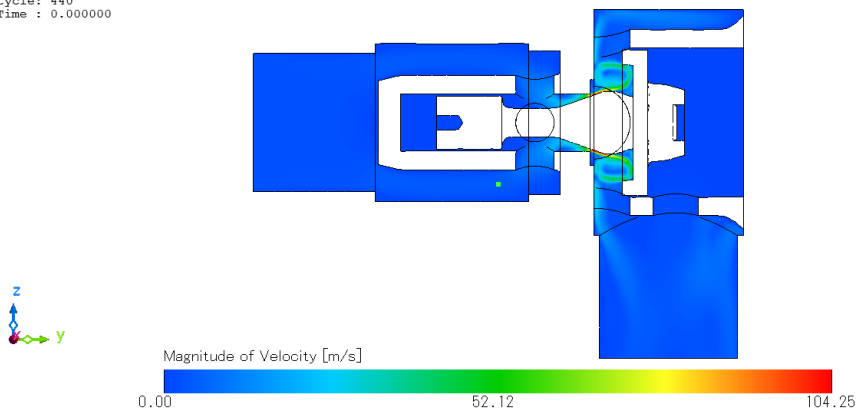
(a) Velocity field for coarse grid

File : valvola\_mod\_LD\_66.fph  
Cycle: 66  
Time : 0.000000

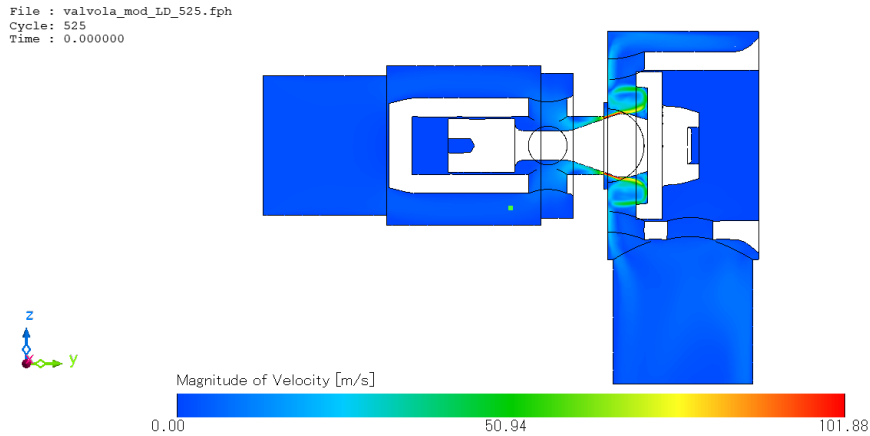


(b) Velocity field for mid grid

File : valvola\_mod\_LD\_440.fph  
Cycle: 440  
Time : 0.000000



(c) Velocity field for fine grid



(d) Velocity field for extra fine grid

Figure 8.5: Velocity fields

Thus, it can be seen that, again starting from the fine grid, the change in solution becomes minimal, approaching a grid-independent solution.

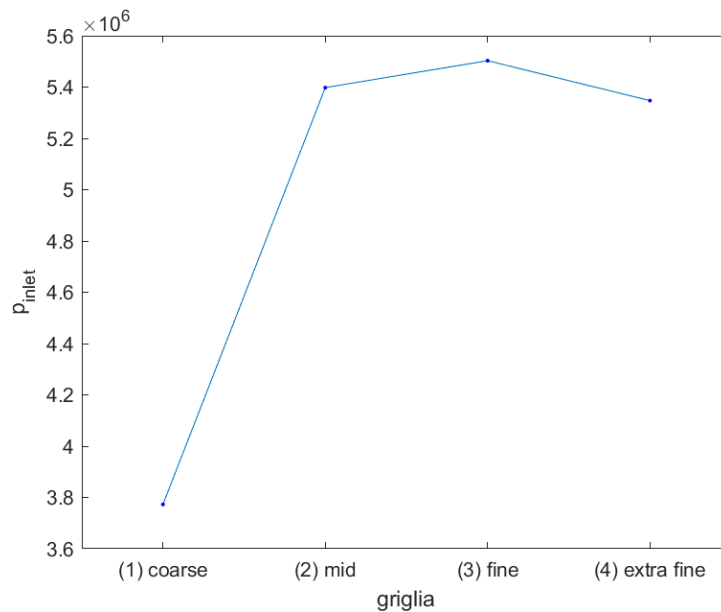


Figure 8.6: Grid independence study with  $k - \omega$  model

In this case, it is also noted graphically that grid convergence is not achieved, so the solution obtained still depends strongly on the characteristic size of the grid used, and the discretization error is higher than the previous case.

Grid convergence studies ensure simulation results stem from boundary conditions and physics, not mesh resolution. Achieving grid independence requires discerning the difference between asymptotic numerical values and true values. While grid-independent numerical values eliminate spatial discretization errors, the asymptotic value may still be affected by modeling errors, leading to convergence toward a value different from the true one.

## 8.1 Convergence order evaluation

When performing simulations with different grid resolutions, if the numerical solution tends to an asymptotic value, then the scheme is considered convergent.

The discretization error is given by:

$$E = U_h - U_0 = kh^p + \dots \quad (8.1)$$

where  $U$  is a generic quantity of interest extracted from the CFD calculation;  $k$  is a constant;  $p$  is the order of the scheme;  $h$  is the characteristic size of the cells. The *asymptotic range* is the range in which the scheme begins to behave as predicted in the theoretical Taylor analysis, where the cell size tends to zero. Evaluating the known order of convergence using the exact solution allows us to verify whether or not we fall within the asymptotic range; if not, further grid refinement could be necessary.

When the exact solution is not available for complex fields and geometries to calculate the discretization error, it can be estimated using various techniques as presented below.

### Richardson Extrapolation (with Theoretical Order)

Suppose we have two grids of sizes  $h$  and  $rh$ , where  $r > 1$ . The error is:

$$\begin{cases} U_h - U_0 = kh^p + \dots \\ U_{rh} - U_0 = kr^p h^p + \dots \end{cases}$$

Assuming  $p$  is equal to the theoretical value (asymptotic range), the system is characterized by two equations in two unknowns ( $U_0, k$ ). Solving this system yields:

$$\begin{cases} k = \frac{U_h - U_0}{h^p} \\ U_{rh} - U_0 = \frac{U_h - U_0}{h^p} r^p h^p \end{cases}$$

From which:

$$U_0 = \frac{r^p U_h - U_{rh}}{r^p - 1} \quad \implies \quad E_h = \frac{U_h - U_0}{U_0}$$

Where  $E_h$  is the relative error. In conclusion, given two simulations with two different grid levels, Richardson extrapolation extracts the exact solution from these as if it were obtained with an infinitely fine grid. Richardson extrapolation works well within the asymptotic range as it assumes that the error decreases according to a law derived from Taylor analysis, and is therefore true with sufficiently fine grids.

### Grid Convergence Index

In a complex problem, there are several possible sources of error, such as discretization or modeling.

GCI allows us to assess the error due to discretization, having estimated the exact solution with Richardson extrapolation, from which the error is derived; it is evaluated as:

$$GCI = F_s E_h$$

Where  $F_s$  is a scaling factor, which takes into account the ratio of grid refinement steps depending on the problem; in our case  $F_s = 3$ .



### Richardson Extrapolation (with Empirical Order)

Richardson extrapolation with empirical order is more effective when  $p$  is not within the asymptotic range (for example, when shock waves are present). Compared to the previous case, an additional unknown is introduced: for this reason, three grid levels are considered.

Assuming  $r = 2$ ; the error is:

$$\begin{cases} U_h - U_0 = kh^p \\ U_{2h} - U_0 = k2^p h^p \\ U_{4h} - U_0 = k4^p h^p \end{cases}$$

The system consists of 3 equations in 3 unknowns ( $U_0, p, k$ ). We obtain:

$$p = \frac{\ln\left(\frac{U_{4h} - U_{2h}}{U_{2h} - U_h}\right)}{\ln 2}$$

$$U_0 = U_h - \frac{U_{2h} - U_h}{2^p - 1}$$

From which the relative error can be estimated as:

$$E = \frac{U_h - U_0}{U_0}; \quad GCI = F_s E$$

#### 8.1.1 Results

In conclusion, we report the obtained values of pattern order ( $p$ ), Error  $E_h$  and  $GCI$  as described before. For the turbulence model  $k - \varepsilon$ :

Richardson - theoretical order	
	$k - \varepsilon$ model
$p$	1
$E_h$	0.0087
$GCI$	0.0262
Richardson - empirical order	
	$k - \varepsilon$ model
$p$	0.8330
$E_h$	0.0112
$GCI$	0.0337

Table 8.3: Convergence order evaluation for  $k - \varepsilon$  model

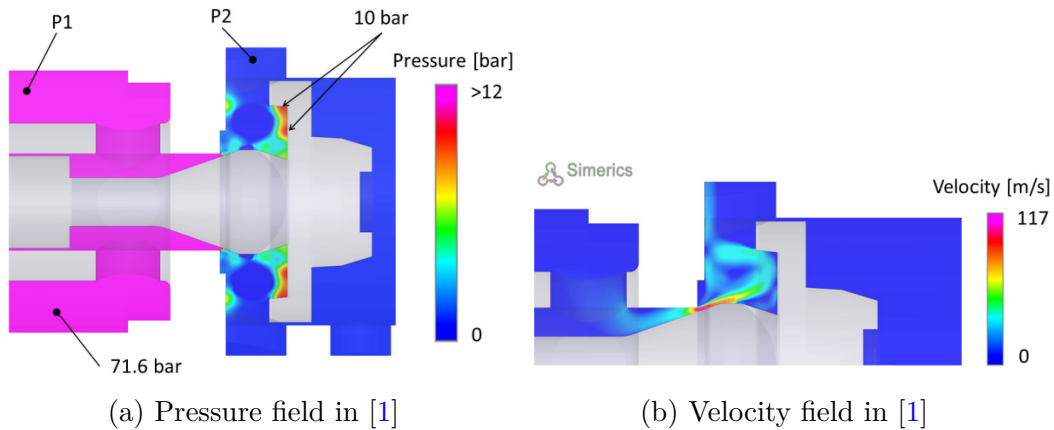
In this case, the results confirm those obtained in Figure 8.3: the relative discretization error is small in the case of theoretical order; in the case of empirical order, the order of convergence is very close to the theoretical value of 1, and similarly, the relative discretization error is contained. The same study is repeated with the turbulence model  $k - \omega$ .

Richardson - theoretical order	
	$k - \omega$ model
$p$	1
$E_h$	0.03
$GCI$	0.0899
Richardson - empirical order	
	$k - \omega$ model
$p$	-0.5677
$E_h$	-0.0171
$GCI$	-0.0512

Table 8.4: Convergence order evaluation for  $k - \omega$  model

As shown in the table, the empirical order  $p$  turns out to be negative: this means that the calculation is far from convergence, so we would have  $p = 1$ ; consequently, the values of  $E_h$  and  $GCI$  also turn out to be negative. This result is consistent with having set as a convergence criterion with a threshold equal to  $10^{-3}$ .

### 8.1.2 Model verification



Starting from the obtained results, a comparison is performed with those presented in [1]. It is important to note that the comparison is purely qualitative: despite the similar valve opening, the analysis under study is a stationary analysis performed at a fixed lift of the shutter at 1 mm; conversely, the comparison is made with results from a dynamic analysis, at the end of which the presented results are obtained with a slightly different valve opening, even if very similar. The two simulations, although executed using the same turbulence model, exhibit differences in the results. Notably, from the comparisons, it is observed that the attained velocities and the pressure field have slightly lower values than those of the reference [1]. The difference may arise from a combination of factors related to the numerical methods used and variations in the implementation of the fluid dynamics model.

Despite achieving grid convergence and utilizing a similar turbulence model, each CFD simulation software employs different algorithms for equation approximation or resolution, leading to results that may differ even minimally; additionally, even with analogous boundary conditions, the distinct implementation of resolution algorithms can influence flow calculations within

the domain cells and, consequently, the computation of conservative variables. Finally, rounding errors and approximations can accumulate and impact the final results, with each software handling errors differently.

However, the primary distinction lies in the use of only liquid, which has a compressibility of  $0.5 \text{ GPa}^{-1}$ , employed for stabilization purposes in the calculation. This differs from the reference [1], which includes a gas volume percentage of 2%.

The presence of gas can alter the behavior of fluid flow. For instance, the existence of bubbles or vortices can modify the velocity and pressure distribution within the domain or in specific regions, impacting heat exchange and local temperatures in the fluid. Additionally, simulations involving a liquid-gas interface in the computational domain require specific models to handle the effects of this interface. These models account for surface tension, frictional forces, and density variations between the two phases; if the gas possesses different thermal or chemical properties than the liquid, it can influence temperature distribution and the concentration of chemical species. Lastly, the number of cells used is lower in the second case, and the fluid passage section opening of 1 mm is approximately of the same order as that associated with reaching the equilibrium position from the "Shutter" side, but it remains an approximation.

However, qualitatively, the two simulations yield similar results with errors on the order of 5%, considered acceptable for performing the cosimulation with *Adams*.

# Chapter 9

## Results

In a hydraulic valve, the stable operating conditions are significantly influenced by the flow forces resulting from the alteration of fluid momentum within the valve. It can be shown that, with a consistent pressure drop across the valve, the steady-state force increases in direct proportion to the displacement of the spool, as a virtual spring. Additionally, the force's magnitude is influenced by the fluid velocity at the point where the jet exits the deflector, thus contingent on the degree of jet dispersion.

The following results provide an initial evaluation of the valve's behavior at the end of the opening transient, examining numerous effects following the variation of specific parameters analyzed below. It is emphasized that, to capture all the fluid dynamics aspects of the problem under consideration, it is essential to use very fine computational grids; in this specific case, sufficiently fine computational grids have been used while also considering the computational times required to perform a cosimulation.

### 9.1 First analysis

The initial approach to the problem involved simplifying the valve geometry. Indeed, the "Cartridge assembly" part posed a critical issue as it was not possible to measure whether the pressure between the walls of the chamber surrounding the spring was exactly equal to that of outlet port. The purpose of the analysis, conducted without damping, was to obtain an initial approximate assessment of the displacements and velocities of the "Shutter" part, in order to evaluate possible damping values to use so that the problem reflects real conditions.

An initial exploratory analysis was then carried out by modifying the geometry of the walls, particularly by reducing their extent, as follows:

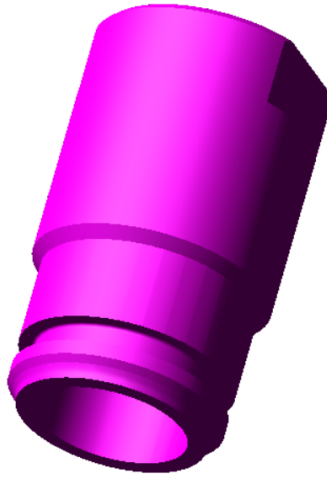


Figure 9.1: Cartridge assembly modifications

A preliminary analysis was then performed using the following conditions:

Outlet pressure (bar)	Damping (Ns/m)
50	No

Table 9.1: First analysis configuration

From the analysis, the following results were obtained:

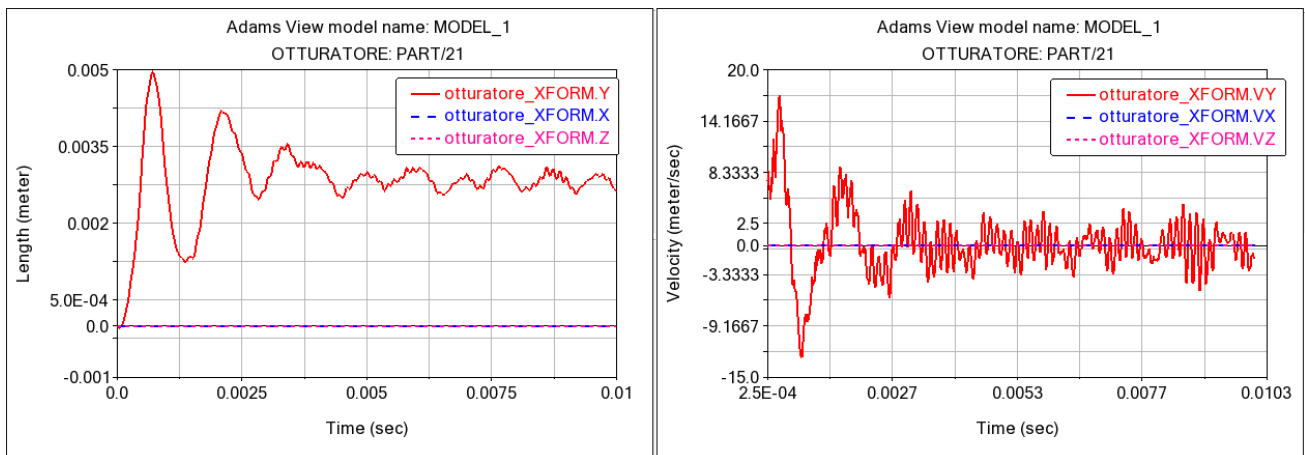


Figure 9.2: Shutter displacement and Shutter speed in x, z and y-direction

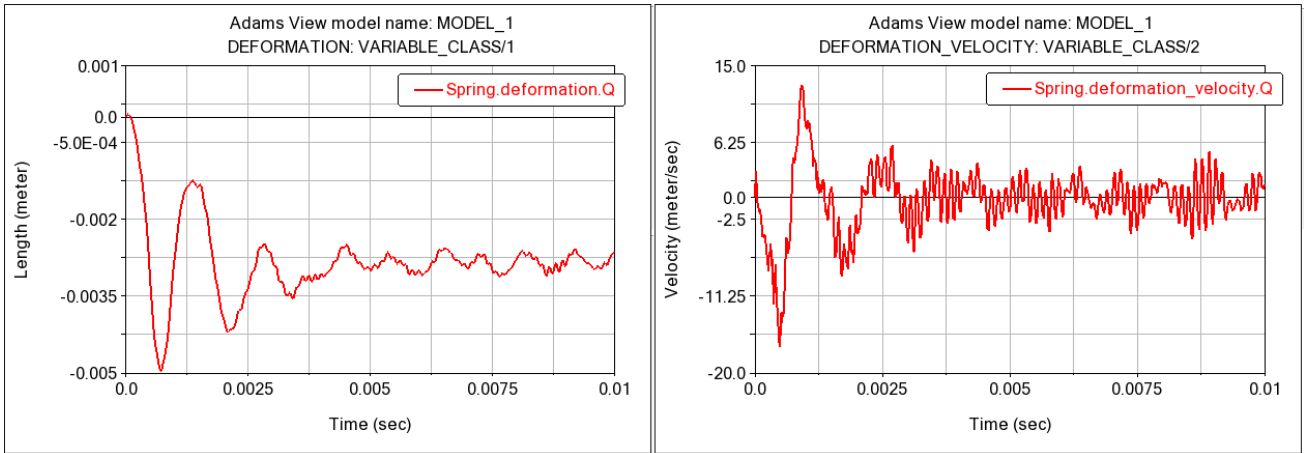


Figure 9.3: Spring deformation and Spring deformation velocity

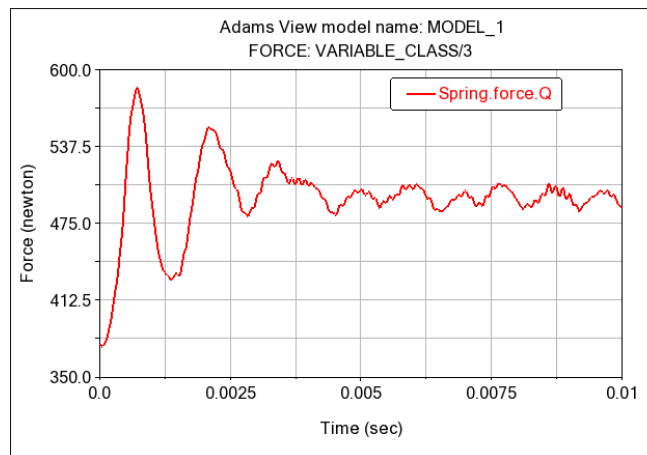


Figure 9.4: Spring force

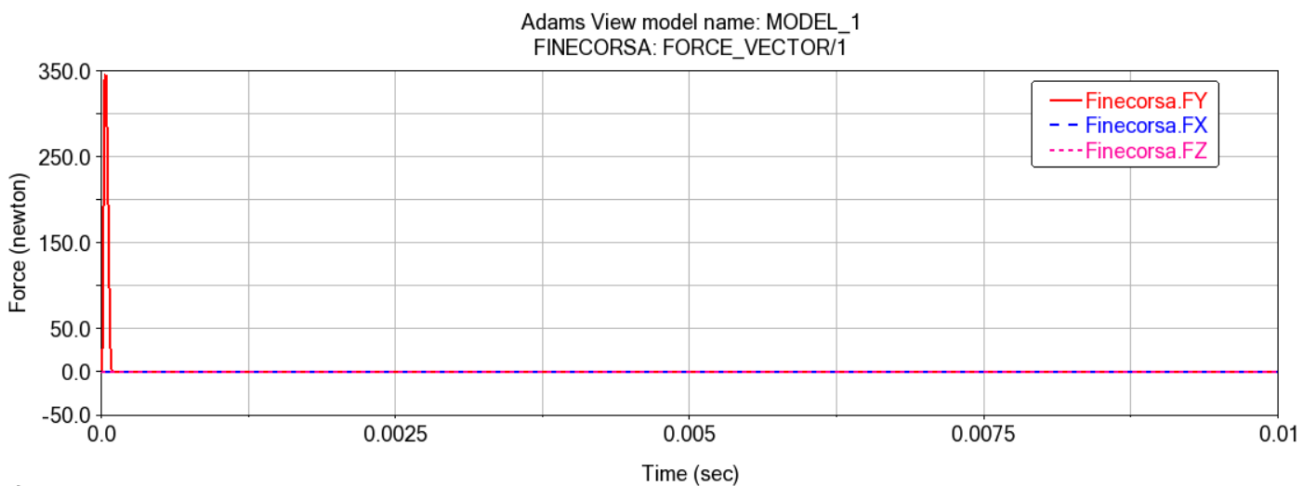


Figure 9.5: Shutter Limit Switch force

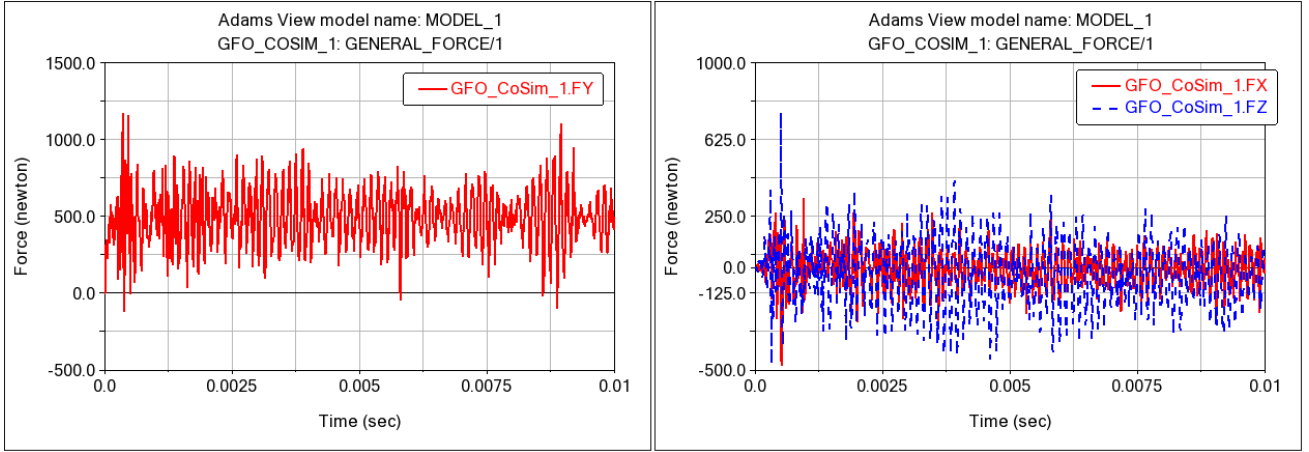


Figure 9.6: Gforce in y-direction and Gforces in x and z directions

As a result, the oscillations of the shutter around its equilibrium position dampen, as well as its velocity after approximately 0.0025 s oscillates around zero (shutter at rest). The magnitude of the "Limit Switch" force has an initial peak due to the contact of the shutter with its seat, preventing penetration in the y-direction motion.

Finally, the values of the force acting on the shutter following the opening and passage of the fluid flow at the inlet indicate that the major force acting on the shutter is in the direction of motion (y), although the components in other directions are also significant.

From the evaluation of the displacement and velocity magnitudes involved, since typically for a realistic problem, it is empirically found that:

$$F_e = 10 \div 100 F_v \quad (9.1)$$

where  $F_e$  represents the elastic forces and  $F_v$  the viscous forces, it is inferred that a compatible damping value for the problem at hand is  $c = 0.1 \text{ Ns/m}$ .

## 9.2 More accurate analysis

Therefore, using the same boundary conditions, the simulation is repeated with the complete "Cartridge assembly" part in the following configuration:

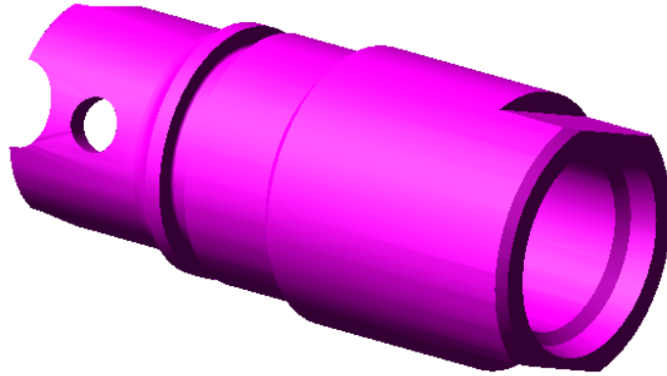


Figure 9.7: Cartridge assembly modifications

For similar reasons to the previous case, a hole is made in the right wall to ensure that the pressure in the chamber is equal to that of the outlet port, while maintaining the geometry of the "Cartridge assembly" part intact.

Since the equilibrium position of the shutter is not influenced by the presence of a damping component, a damping coefficient is assigned to the spring-damper system as determined in the previous evaluation (Eq. 9.1). The simulation starts with the conical shutter already slightly open, with a lift of approximately 1 mm. The following results are obtained:

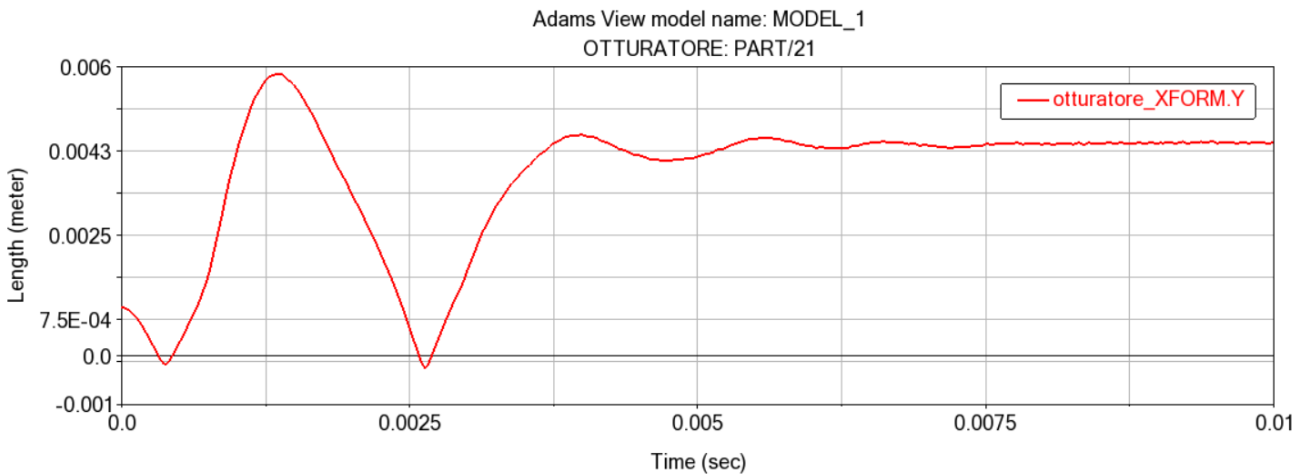


Figure 9.8: Shutter displacement in y-direction



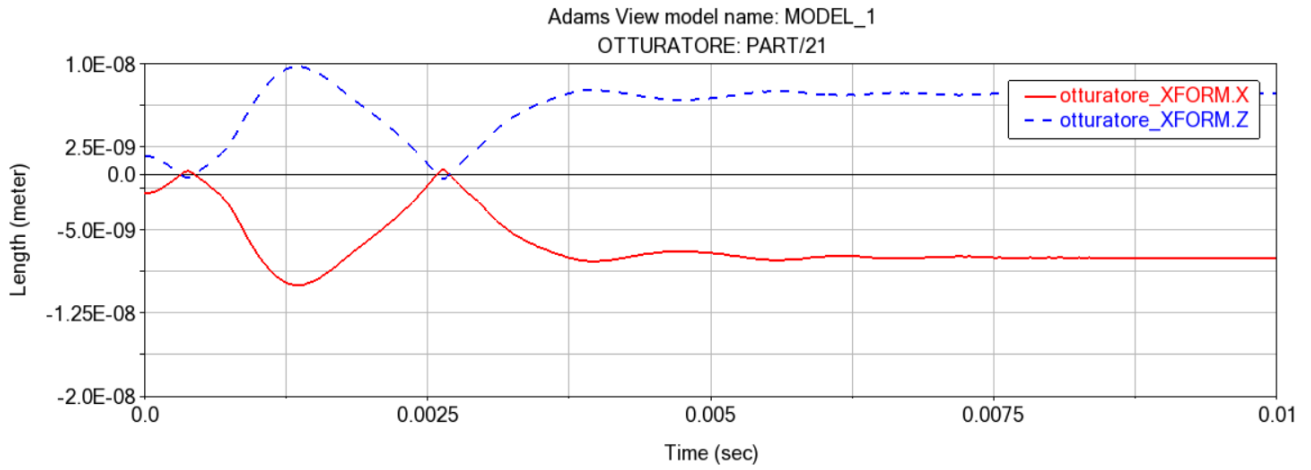


Figure 9.9: Shutter displacement in x and z-direction

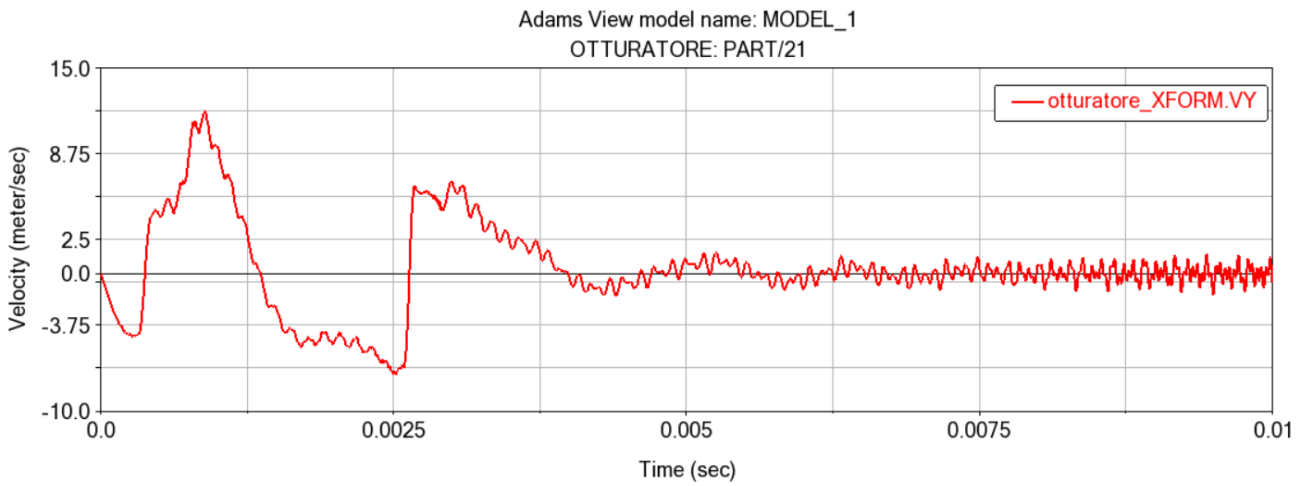


Figure 9.10: Shutter speed in y-direction

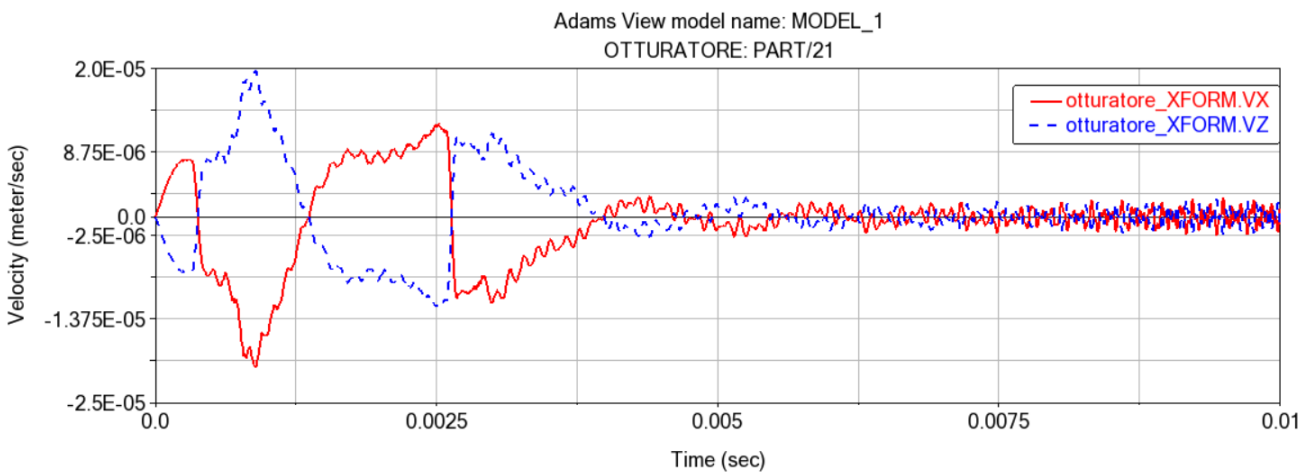


Figure 9.11: Shutter speed in x and z-direction

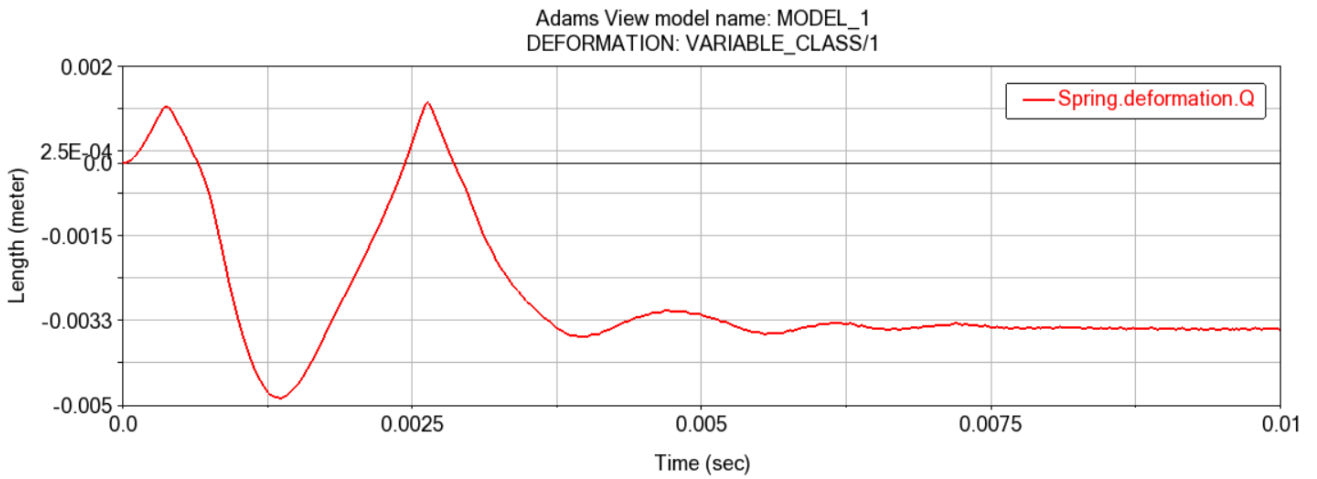


Figure 9.12: Spring deformation

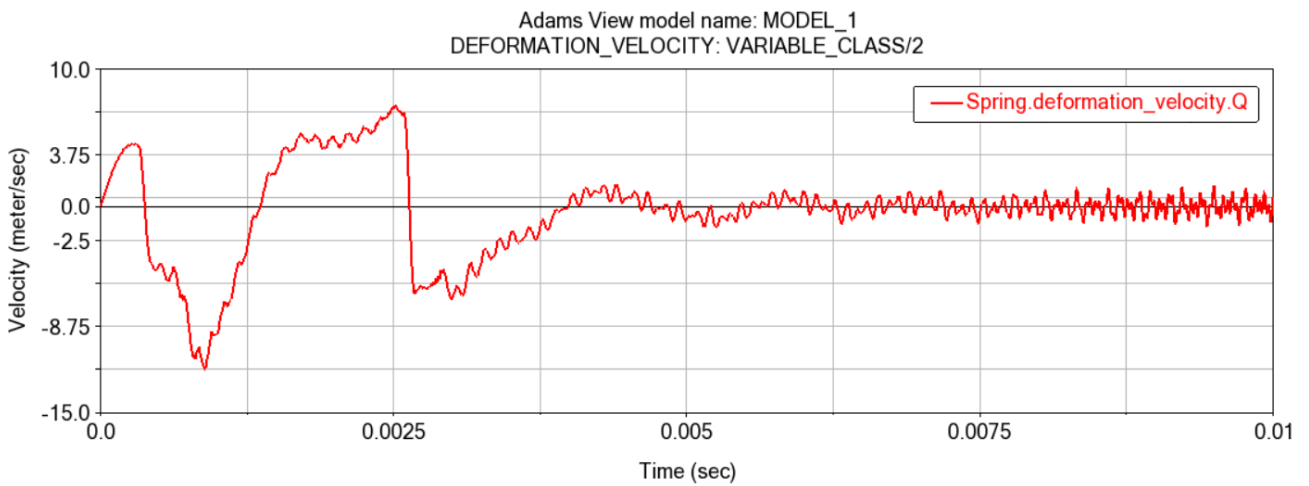


Figure 9.13: Spring deformation velocity

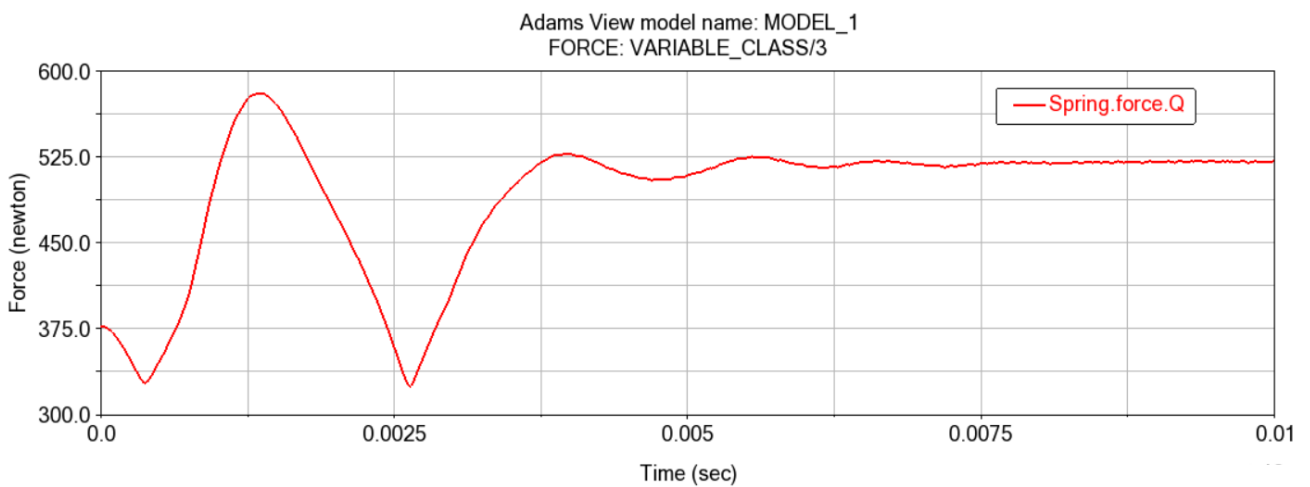


Figure 9.14: Spring force

From the results, it is observed that the Shutter has reached an equilibrium position along the

y-direction with a lift of approximately 4.3 mm from its base; its velocity in the y-direction also oscillates around zero (Shutter at rest). The displacement and velocity components in the x and z directions are small compared to those in the direction of motion.

Similarly, the deformation of the spring and the elastic force follow the same trend as the displacement of the "Shutter" part, and the spring deformation velocity follows the trend of the shutter velocity. In particular, both the spring displacement and the spring force have a linear trend with respect to the displacement of the shutter:

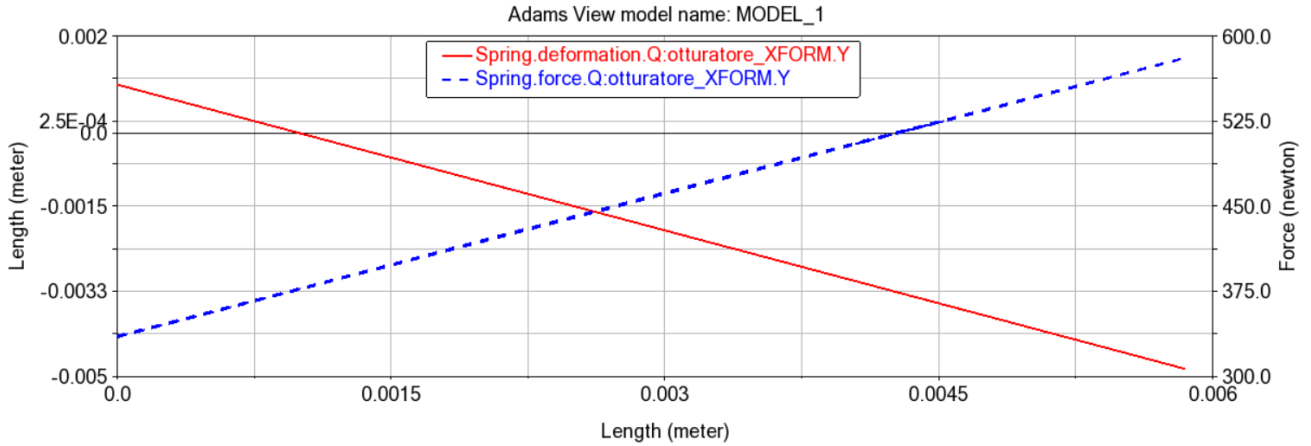


Figure 9.15: spring displacement and the spring force linear trend with respect to the displacement of the shutter

The "Lower Spring Support" part also has similar displacements and velocities as the "Shutter" part, since it is constrained with a fixed joint to it, as follows:

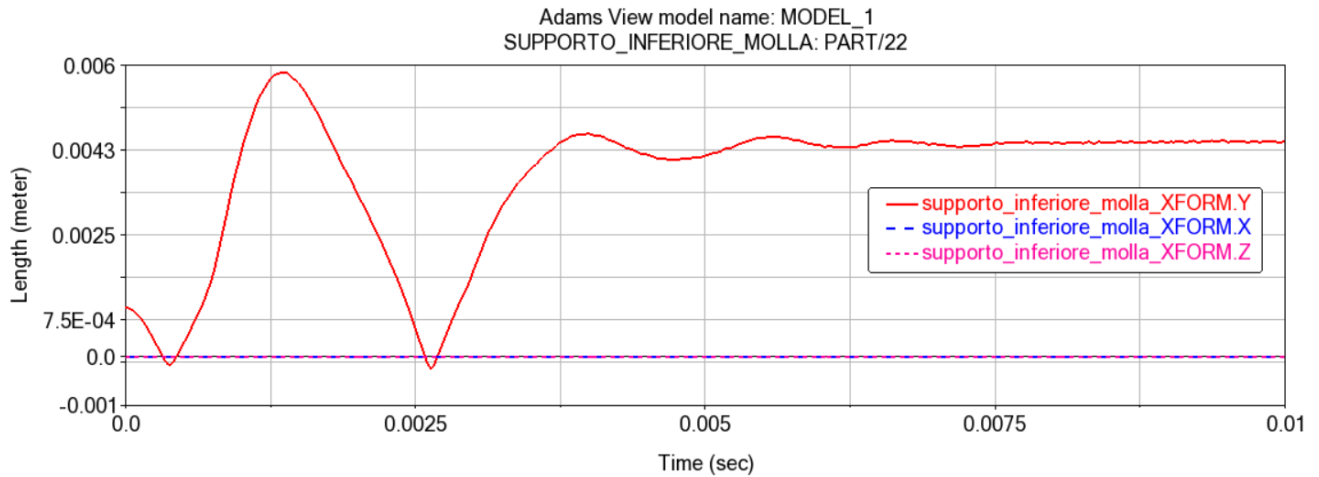


Figure 9.16: Lower Spring Support displacement in y, x and z-directions

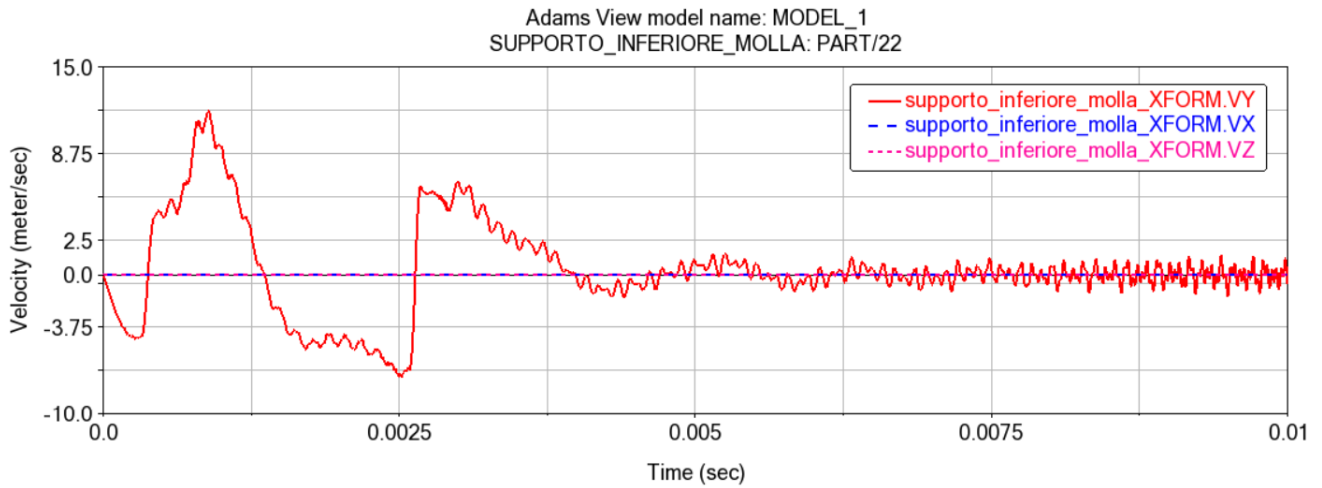


Figure 9.17: Lower Spring Support speed in y, x and z-directions

The forces and moments acting on the closure are then shown.

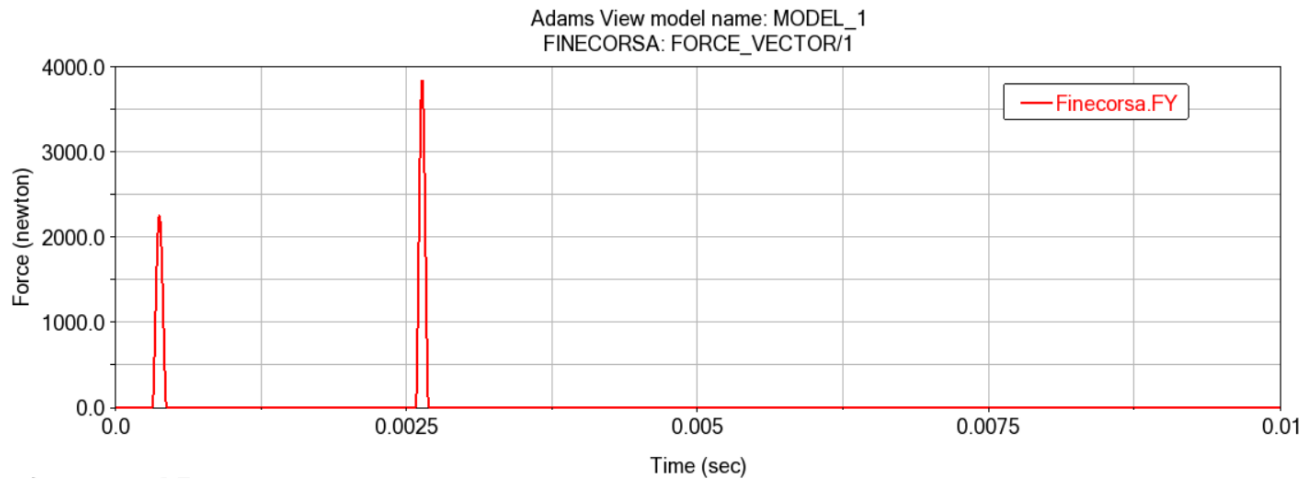


Figure 9.18: Shutter Limit Switch force

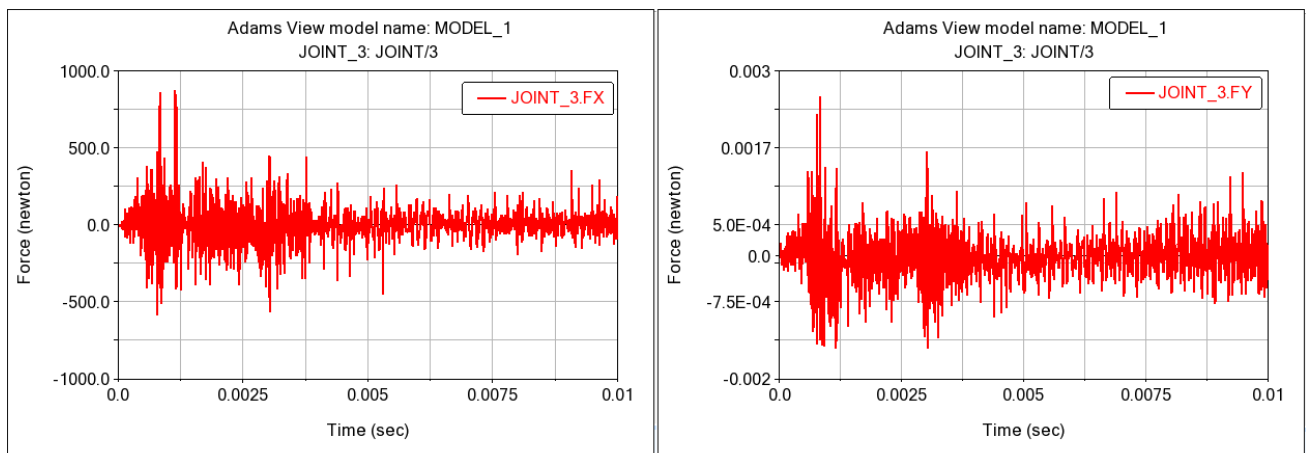


Figure 9.19: Translational joint Force in x and y-directions

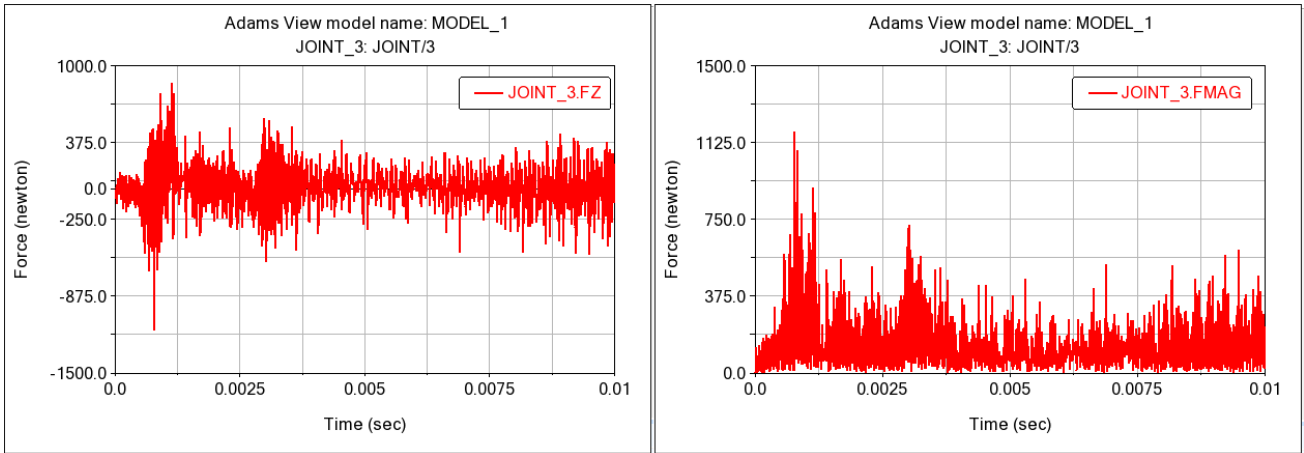


Figure 9.20: Translational joint Force in z-direction and magnitude

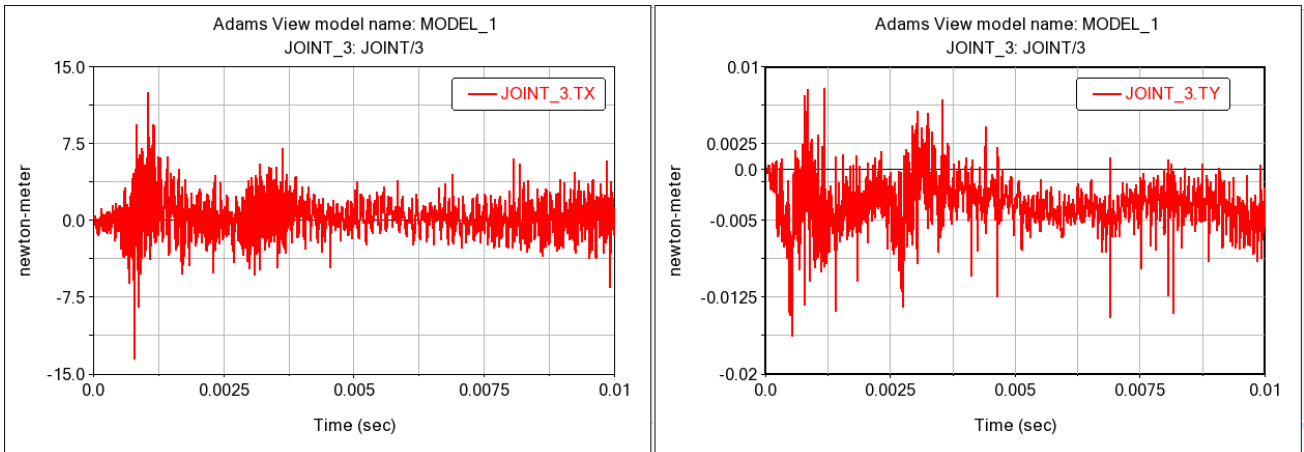


Figure 9.21: Translational joint Torque in x and y-directions

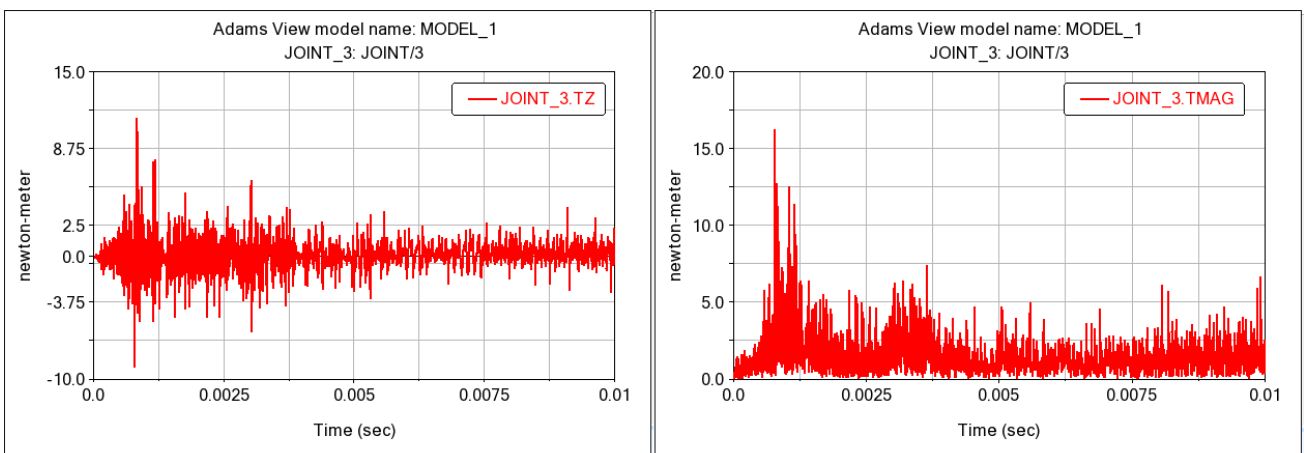


Figure 9.22: Translational joint Torque in z-direction and magnitude

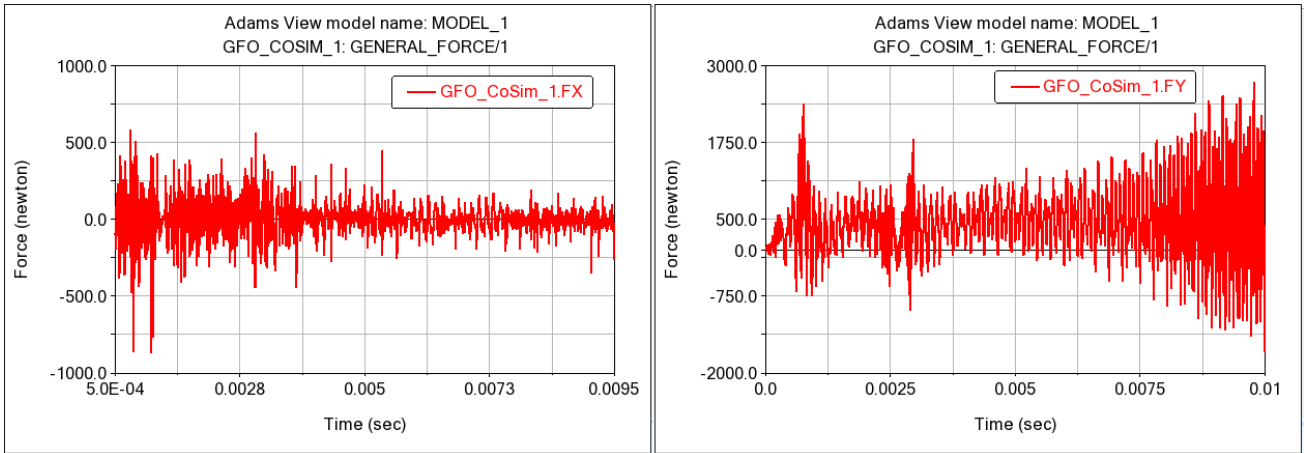


Figure 9.23: Gforce in x and y-directions

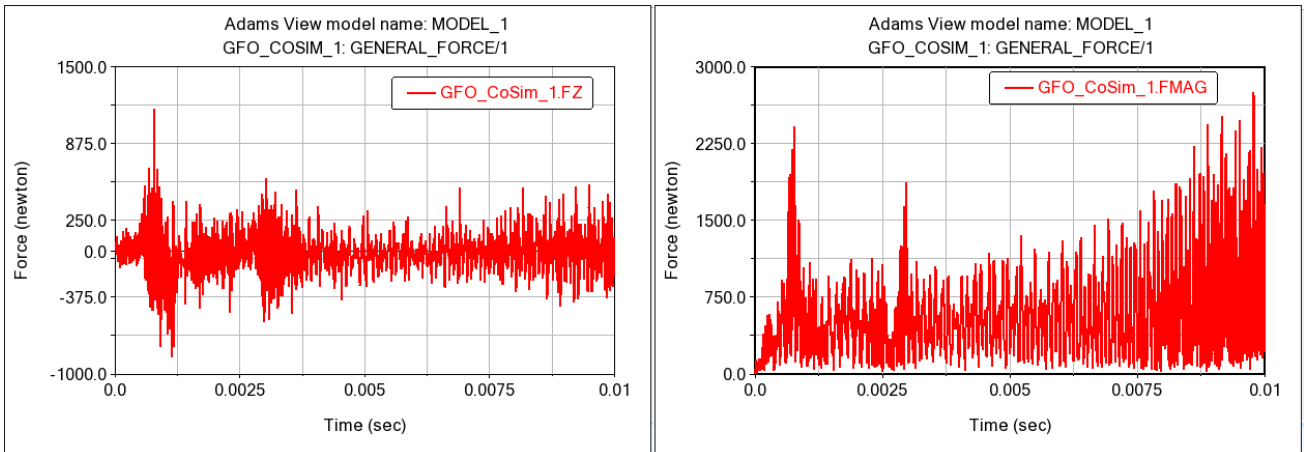


Figure 9.24: Gforce in z-direction and magnitude

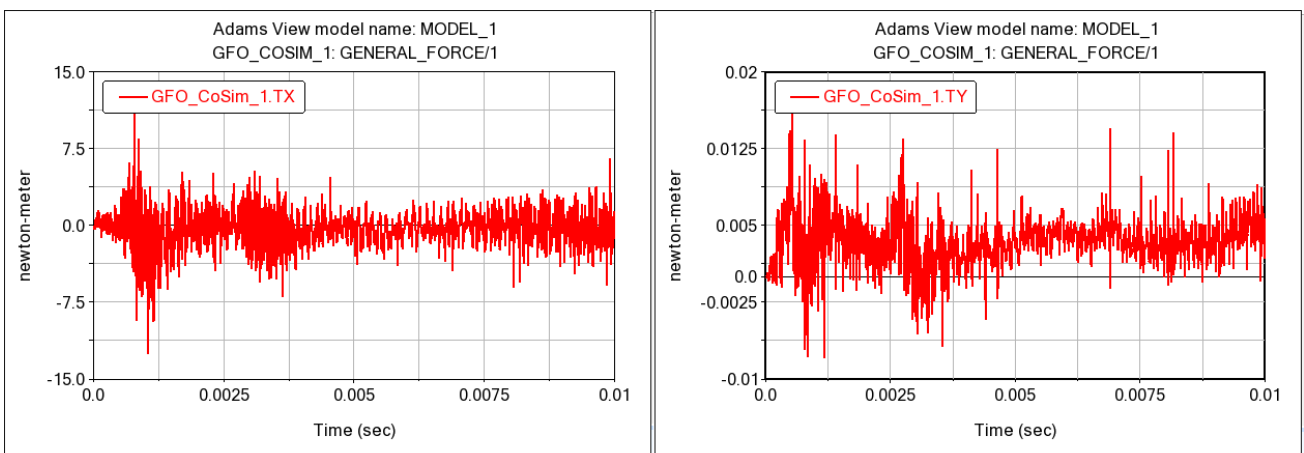


Figure 9.25: Gforce, torque components in x and y-directions

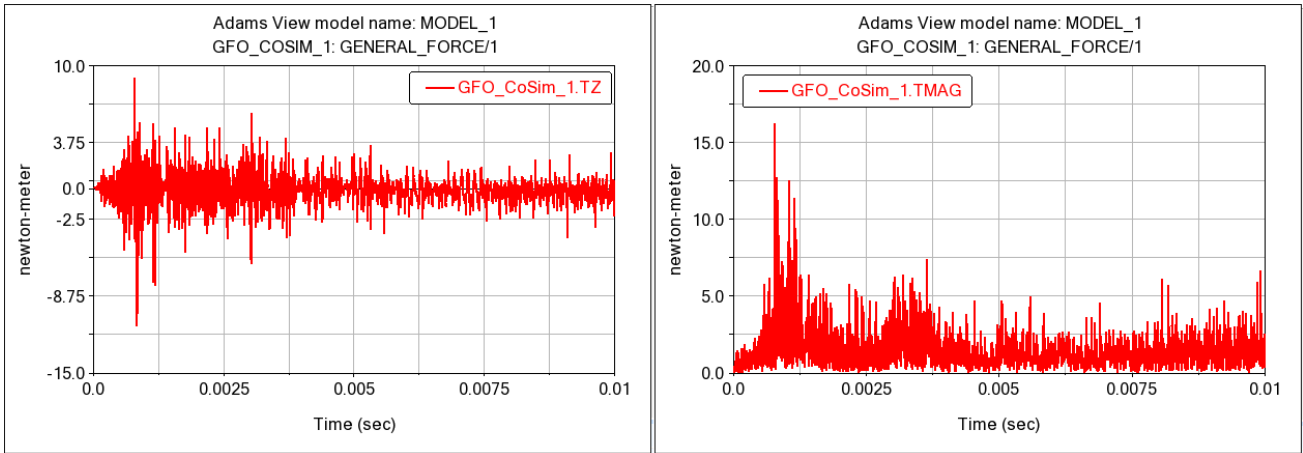


Figure 9.26: Gforce, torque components in z-direction and magnitude

The magnitude of the "Limit Switch" force has an initial peak and another one due to the contact of the shutter with its seat, preventing penetration in the y-direction motion. Lastly, the values of the force acting on the shutter following the opening and passage of the fluid flow at the inlet; at the end of the simulation, the value of the force acting on the shutter should be constant, and the oscillations are numerical errors not representative of reality.

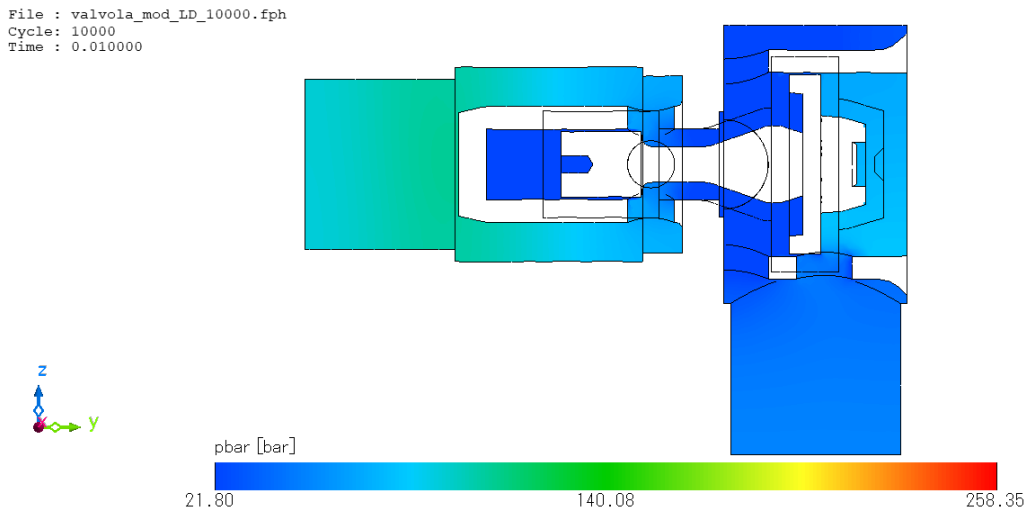


Figure 9.27: Pressure field at end of simulation

File : valvola\_mod\_LD\_10000.fph  
 Cycle: 10000  
 Time : 0.010000

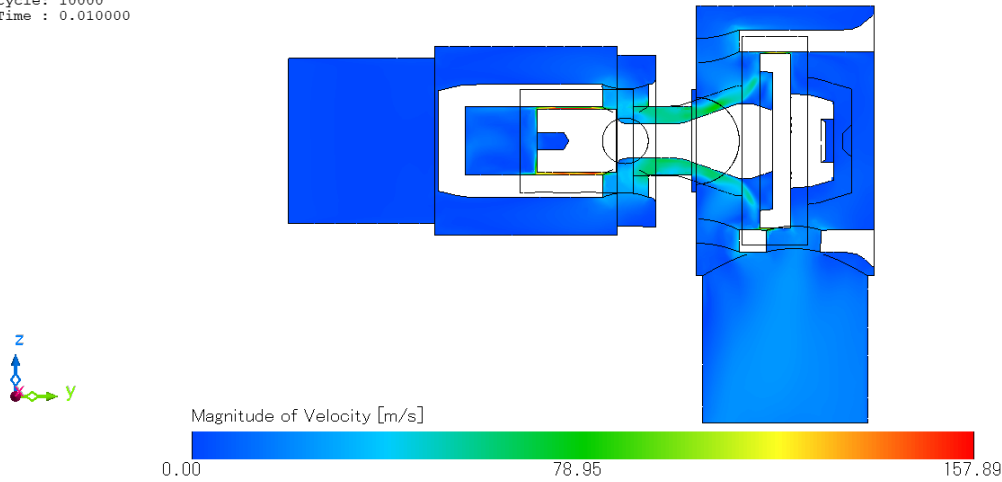


Figure 9.28: Velocity field at end of simulation

File : valvola\_mod\_LD\_10000.fph  
 Cycle: 10000  
 Time : 0.010000

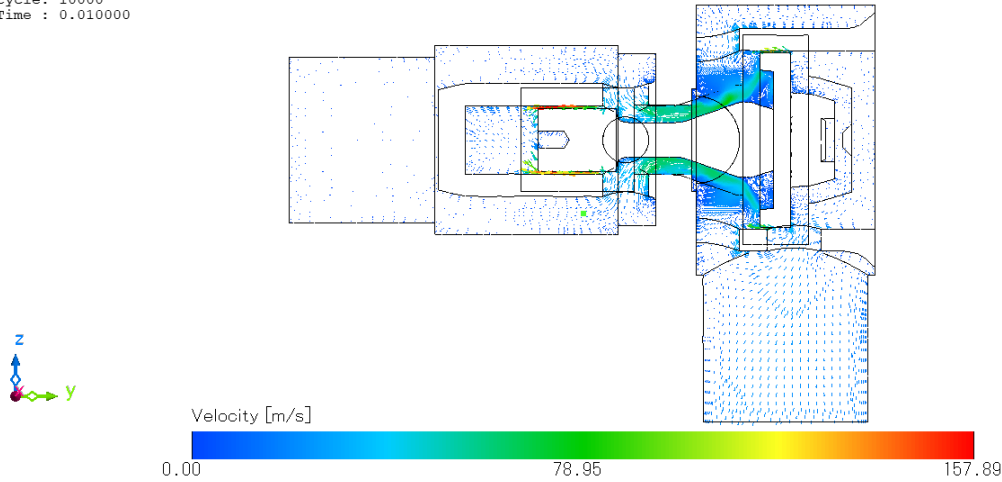


Figure 9.29: Velocity vector field at end of simulation

The presented results represent pressure and velocity fields at the end of the co-simulation (10000 cycles). At the end of the simulation, it can be assumed, through comparison with the results in *Adams*, that the shutter has reached an equilibrium position, and the results can be assumed as those obtained at the end of the transient.

The pressure is higher in the chamber below the shutter, in accordance with reality, and decreases towards the outlet. The pressure drop between inlet and outlet is not excessive thanks to the valve opening due to the movement of the shutter.

From the velocity field, recirculation vortices generated following the flow reflection on the deflector are noticeable; to better capture the phenomenon, it would be advisable to further densify the grid near the recirculation.

Furthermore, an increase in velocity is observed between the lower cylindrical part of the shutter, acting as a damper, and its support: the clearance, which has a beneficial damping effect, however, has the disadvantage of slightly lengthening the transient.

However, some inconsistencies are observed. In the pilot chamber, the pressure is lower than



outside, as if the spool were still moving to the right and the volume of the pilot chamber were still increasing. On the other hand, in Figure 9.28, it can be seen that in the annular gap, the fluid velocity is not zero. Therefore, either it is a drag flow due to the velocity of the spool, or it is a flow entering because the chamber is increasing in volume.

In this regard, a steady-state analysis can be performed with the spool displacement held constant, in order to evaluate the correct pressure field.

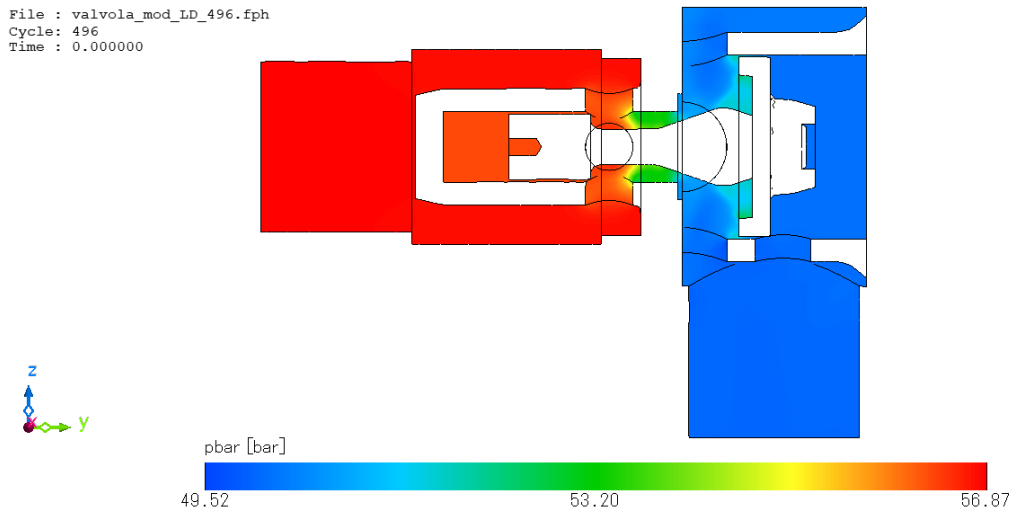


Figure 9.30: Pressure field in steady-state simulation

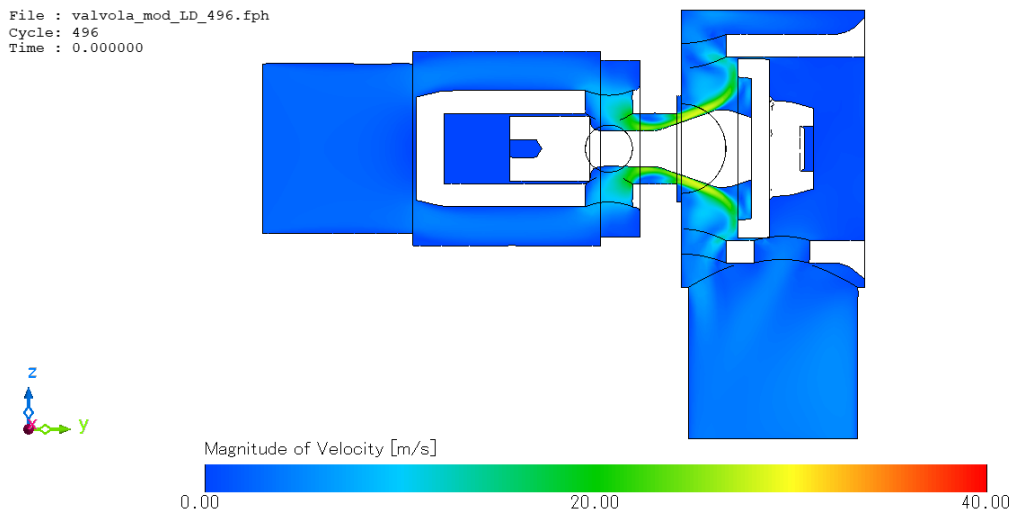


Figure 9.31: Velocity field in steady-state simulation

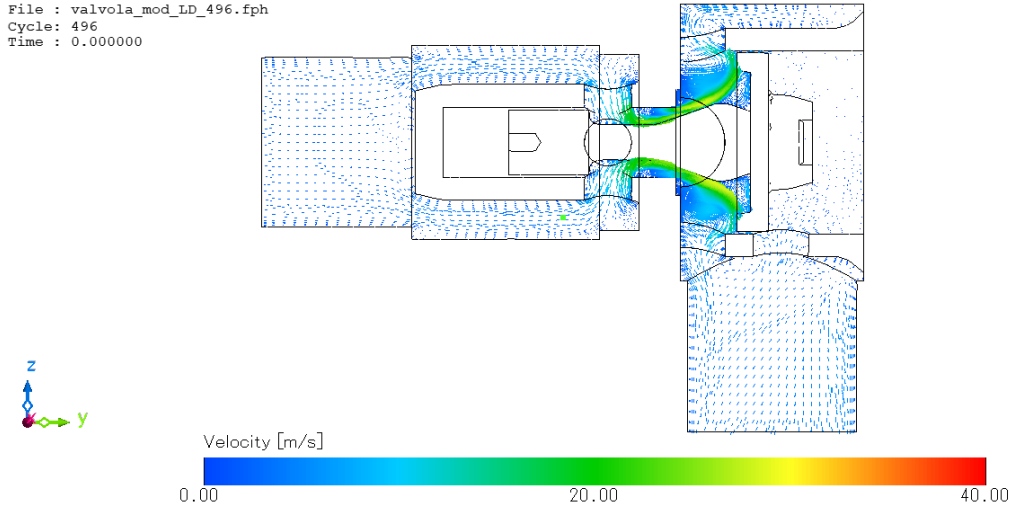


Figure 9.32: Velocity vector field in steady-state simulation

As can be seen from the comparison, the pressure and velocity values at the end of co-simulation are higher than those obtained from the stationary analysis. Qualitatively, however, the results are comparable, except in the domain section above the shutter, where the error in the pressure range is marked.

There are several possible sources of error: first, in order to obtain excellent results, it is necessary to refine the grid considerably more around the moving part, so that the pressure force acting on the shutter is calculated accurately. Second, a critical issue is the stabilization of the pressure field when the shutter reaches an equilibrium position. In this case, the evolution of the pressure field obtained from the co-simulation could not be representative of reality. This may be due to numerical instability related to the use of a too large time step. By imposing a constant time step during the simulation, as the field evolves during the transient, it's possible that the time step was satisfactory at the beginning of the simulation but violated numerical stability toward the end of the simulation. However, by directly imposing the CFL number, the time step is automatically adjusted, and if strong convective terms appear in the fluid, the time step is automatically reduced to maintain the same CFL number.

### 9.3 Effect of damping

To evaluate the damping effect on the transient, since according to Eq. 9.1,  $c = 0.11; Ns/m$  has been assumed, a simulation is performed in the No Damping configuration to turn off all damping forces and create a pure spring. A simulation is conducted starting from the closed valve configuration, i.e., the shutter is in contact with its seat. The following results are obtained:

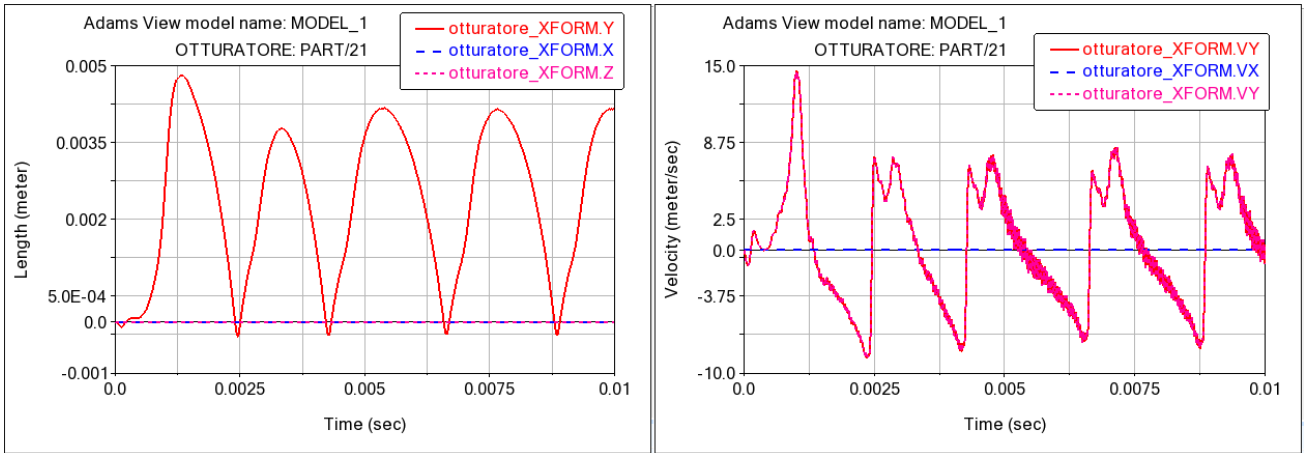


Figure 9.33: Shutter displacement and Shutter speed in x, z and y-direction

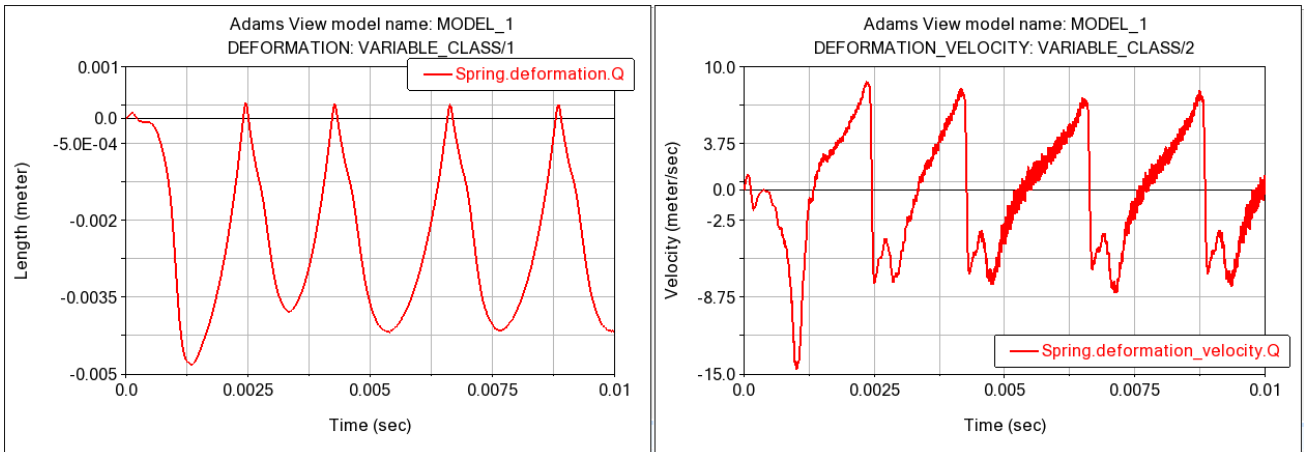


Figure 9.34: Spring deformation and Spring deformation velocity

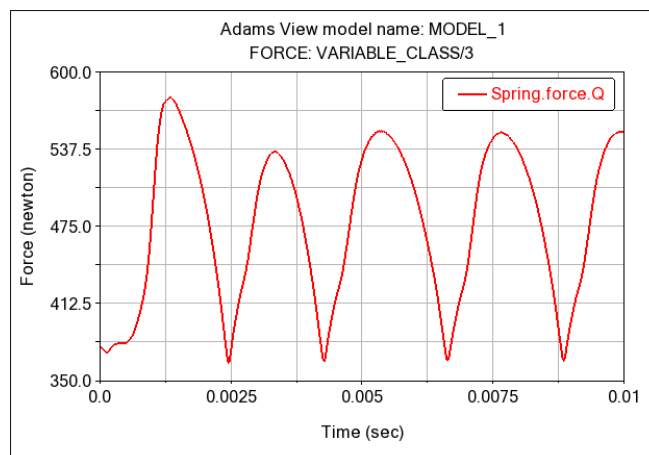


Figure 9.35: Spring force

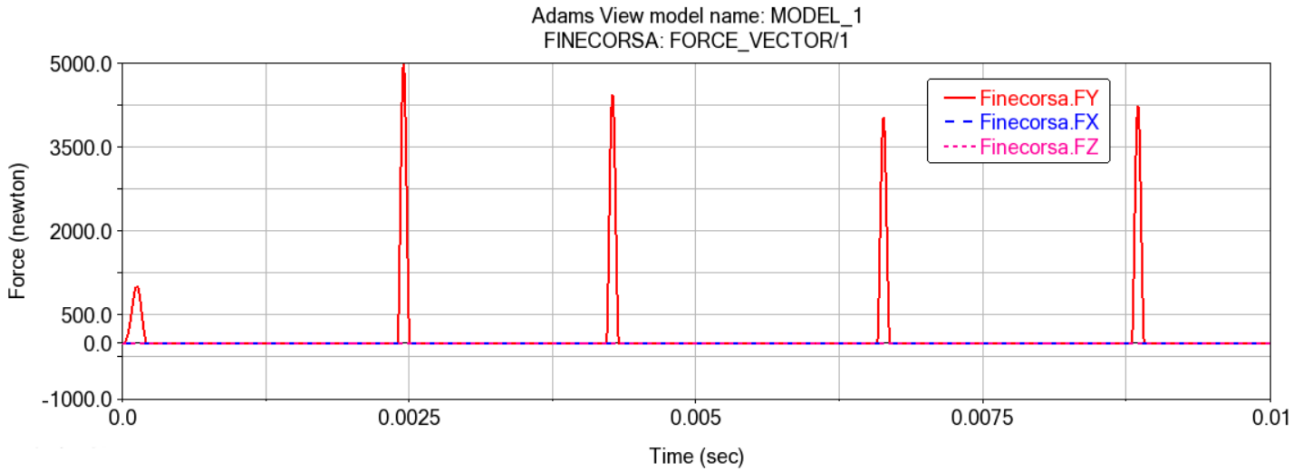


Figure 9.36: Shutter Limit Switch force

It is clear that the transient is significantly longer, and the oscillations dampen much more slowly compared to the damped case. In fact, the default 10000 cycles set for the simulation are not sufficient to reach the equilibrium position, which nonetheless remains the same as it is not influenced by damping.

Indeed, several peaks in the "Limit Switch" force are observed, representing the contact of the shutter with its seat.

## 9.4 Effect of initial lift

Since all simulations were performed starting from a lift of the "Shutter" part of 1 mm to partially reduce the transient, a simulation is conducted under the same conditions but in the No Damping configuration, in order to be compared with the previous one.

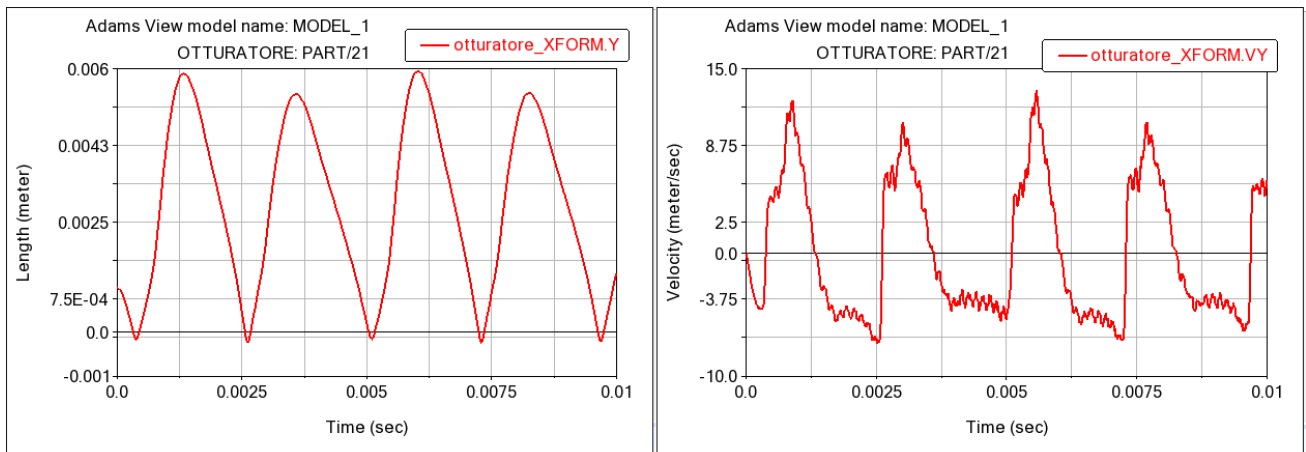


Figure 9.37: Shutter displacement and Shutter speed in y-direction

It is observed that the effect of the 1 mm lift has minimal influence on the transient. There is a reduction in the magnitude of the velocity of the "Shutter" part but not in the displacements.

## 9.5 Effect of preload

Two simulations are conducted to assess the effect of preload variation on valve performance. In order not to overly complicate the analysis, which requires quite high computational times, the effect of preload is evaluated only on the first oscillation to observe its effect on the maximum lift of the "Shutter" part.

The first simulation is carried out by increasing the valve's cracking pressure from 75 to 125 bar. It is expected that, under the same boundary conditions, a higher cracking pressure, and thus a higher preload assigned to the spring, will reduce the opening of the shutter.

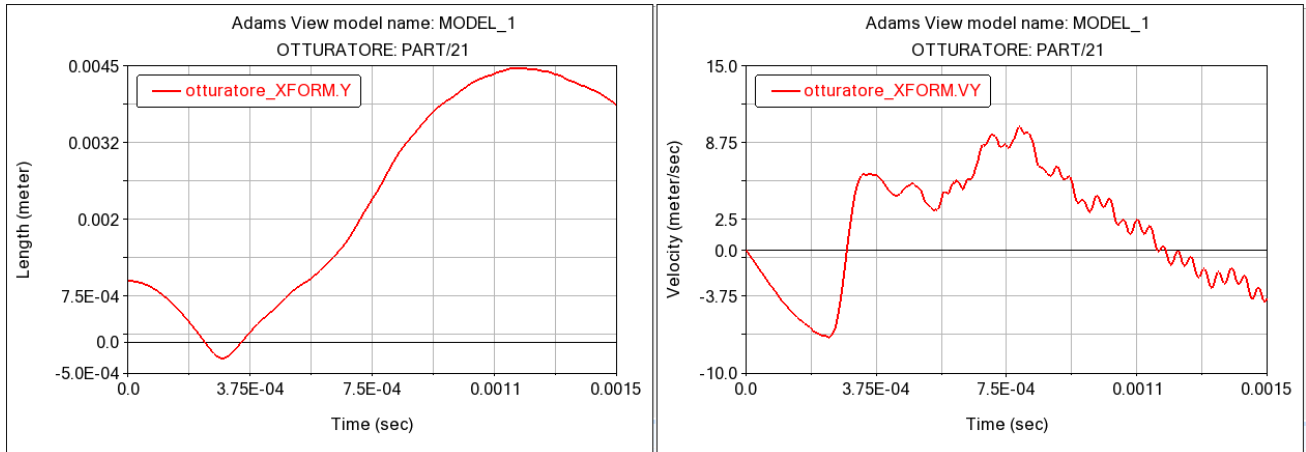


Figure 9.38: Shutter displacement and Shutter speed in y-direction

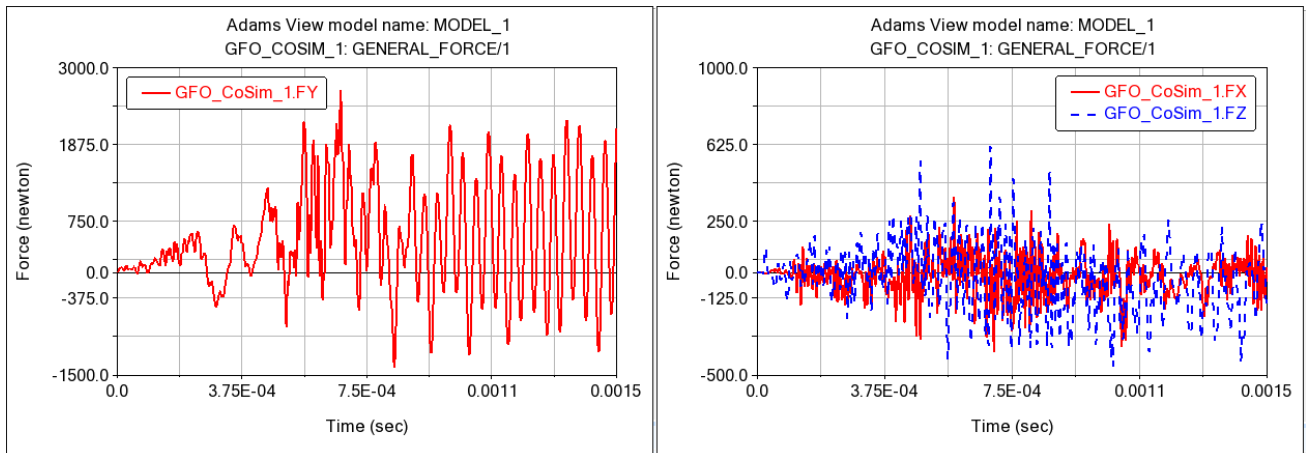


Figure 9.39: Gforce in y-direction and Gforces in x and z-directions

File : valvola\_mod\_LD\_1250.fph  
 Cycle: 1250  
 Time : 0.001250

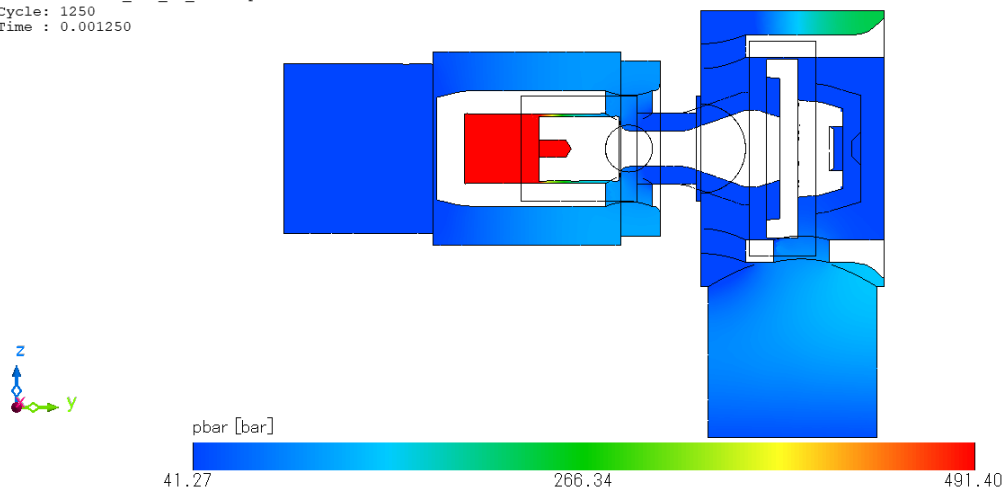


Figure 9.40: Pressure field at maximum shutter lift

File : valvola\_mod\_LD\_1250.fph  
 Cycle: 1250  
 Time : 0.001250

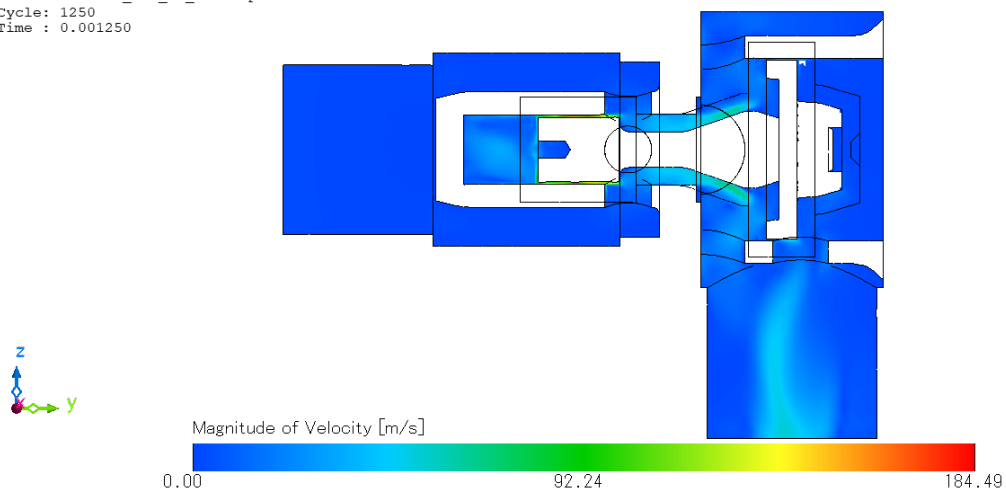


Figure 9.41: Velocity field at maximum shutter lift

Indeed, it is observed that the maximum opening is reduced compared to the first case. The force acting on the shutter also exhibits higher values because a greater force is required to open the valve, as reflected in the pressure field.

It is worth noting that the comparison with the first case should be limited to the first oscillation and not over the entire simulation time.

The analysis is repeated by decreasing the valve's cracking pressure from 75 to 25 bar.

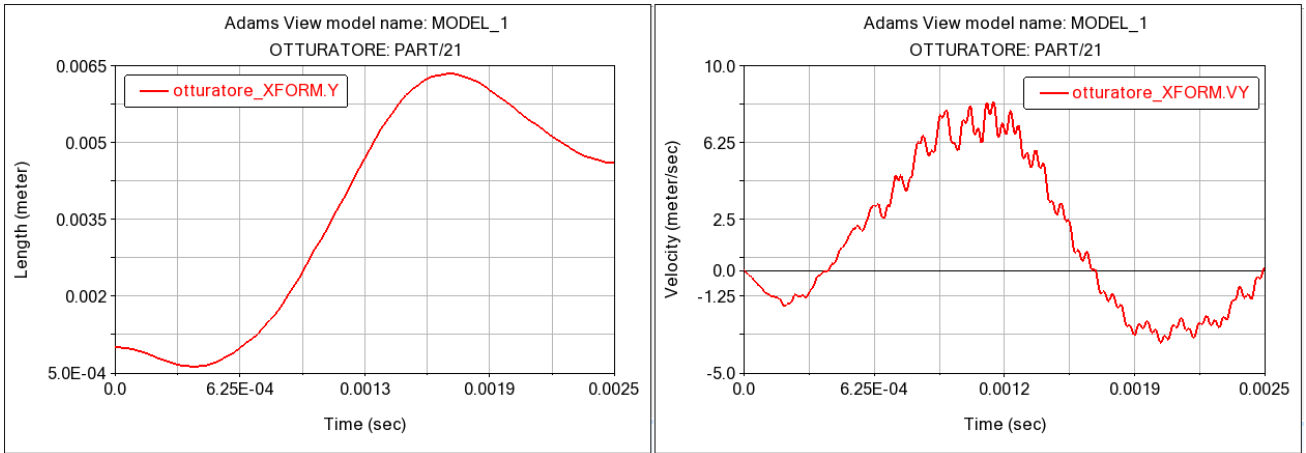


Figure 9.42: Shutter displacement and Shutter speed in y-direction

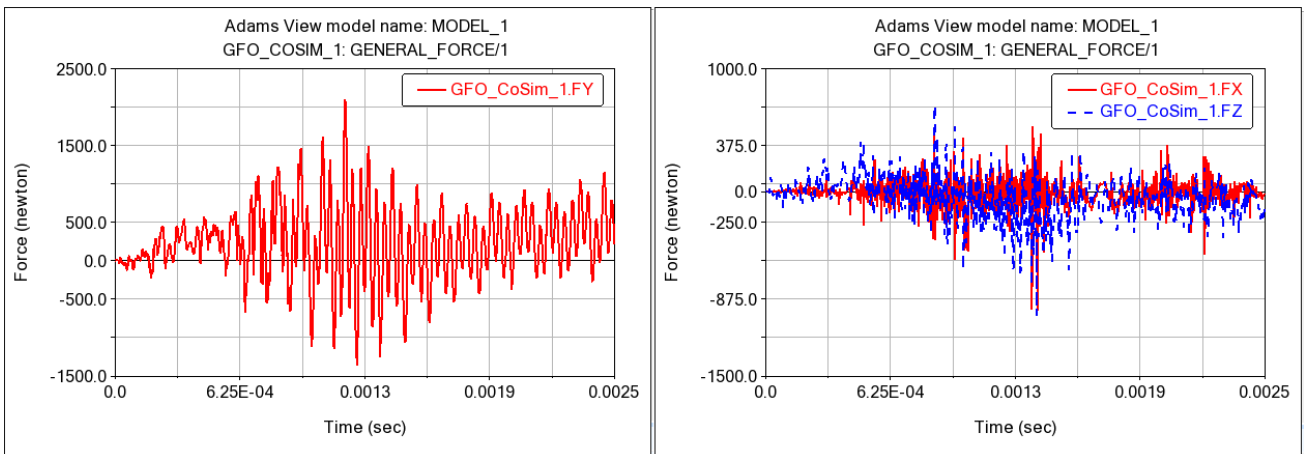


Figure 9.43: Gforce in y-direction and Gforces in x and z-directions

File : valvola\_mod\_LD\_2200.fph  
Cycle: 2200  
Time : 0.002200

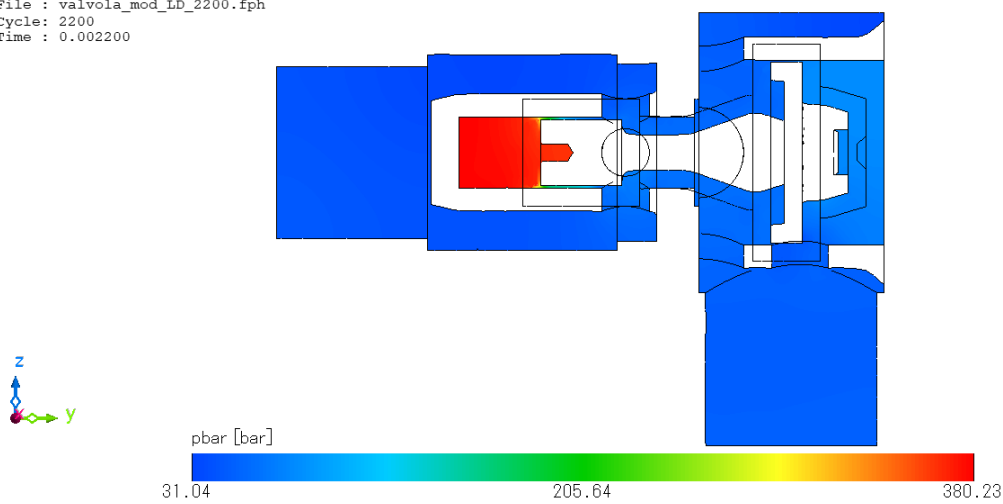


Figure 9.44: Pressure field at maximum shutter lift

File : valvola\_mod\_LD\_2200.fph  
 Cycle: 2200  
 Time : 0.002200

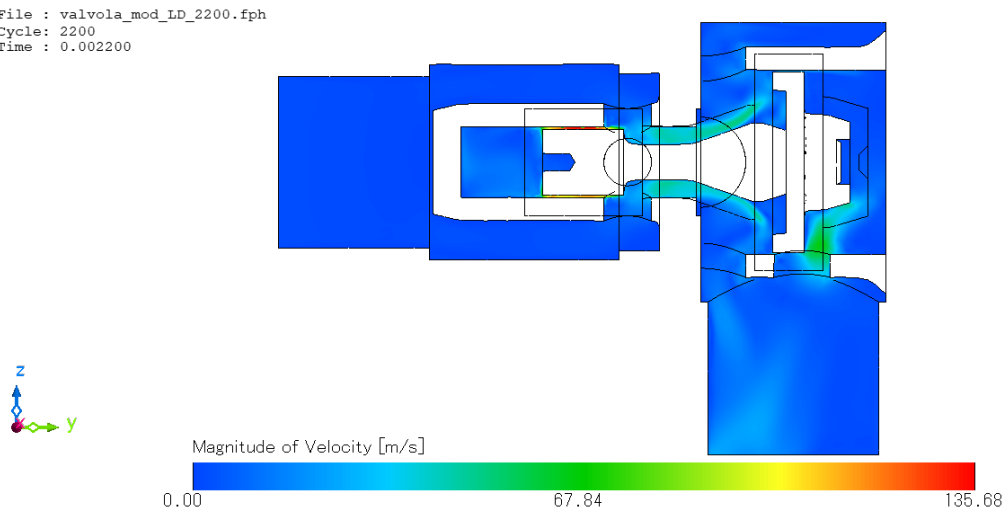


Figure 9.45: Velocity field at maximum shutter lift

In this case, since the preload of the spring is significantly reduced, higher displacements are obtained, and the force acting on the shutter in the direction of motion is lower compared to the previous case. It is also observed that after an initial peak in the opening displacement, the shutter already begins to oscillate with amplitudes close to the equilibrium position, unlike the other cases where the oscillations are more pronounced.

## 9.6 Effect of outlet pressure

Finally, the effect of boundary conditions on the valve behavior is evaluated. Specifically, the outlet pressure is modified from 50 to 60 bar. Therefore, a lower valve opening and a smaller displacement of the "Shutter" part are expected.

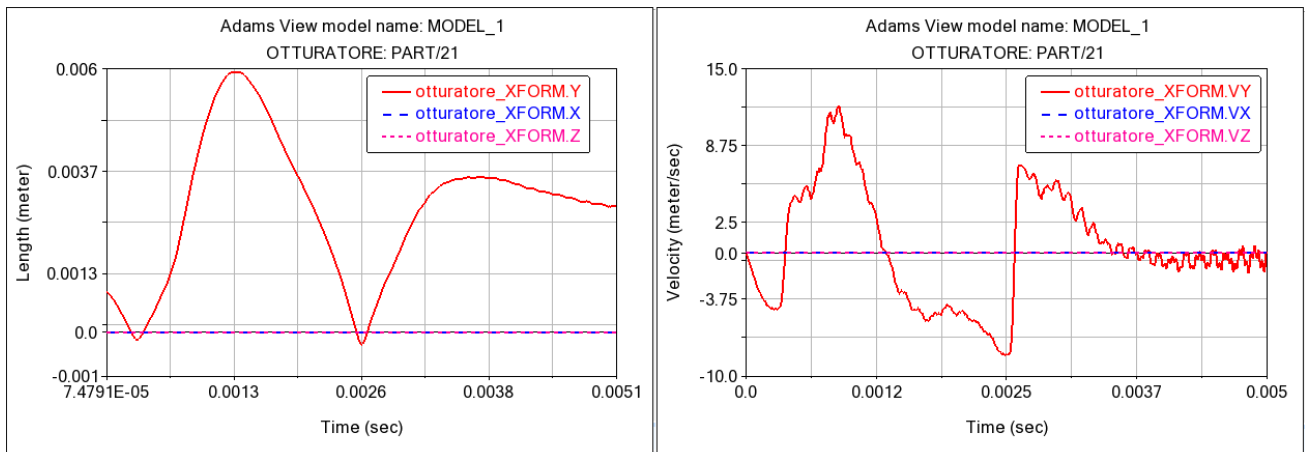


Figure 9.46: Shutter displacement and Shutter speed in y-direction



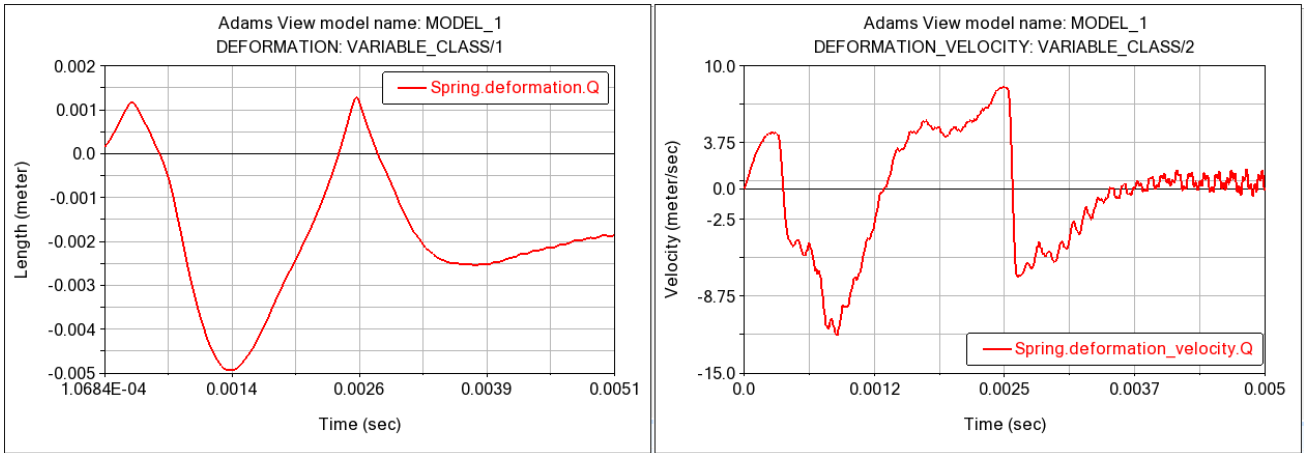


Figure 9.47: Shutter displacement in x and z-direction

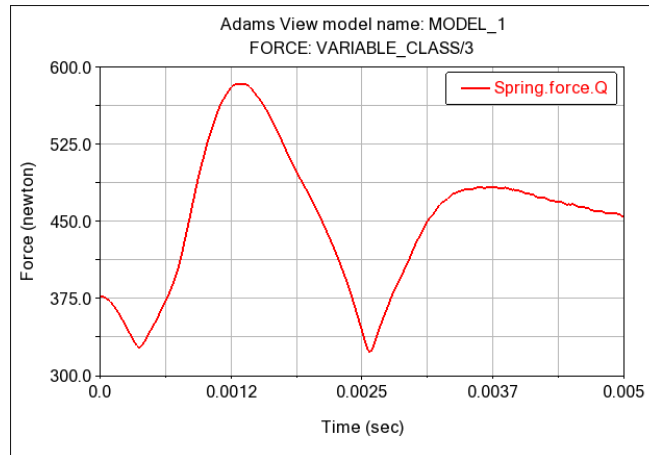


Figure 9.48: Shutter speed in y-direction

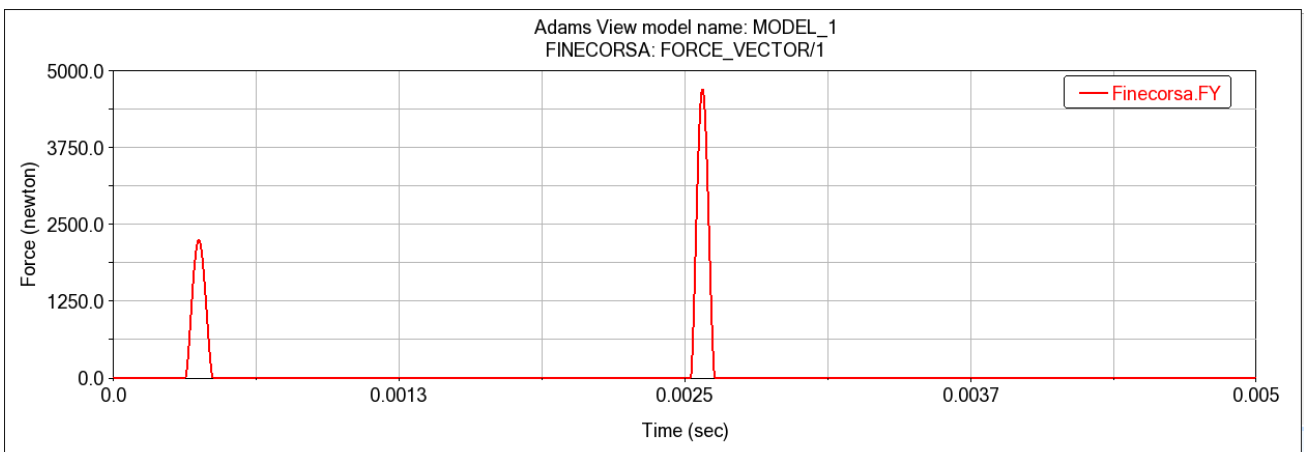


Figure 9.49: Shutter Limit Switch force

File : valvola\_mod\_LD\_5050.fph  
Cycle: 5050  
Time : 0.005050

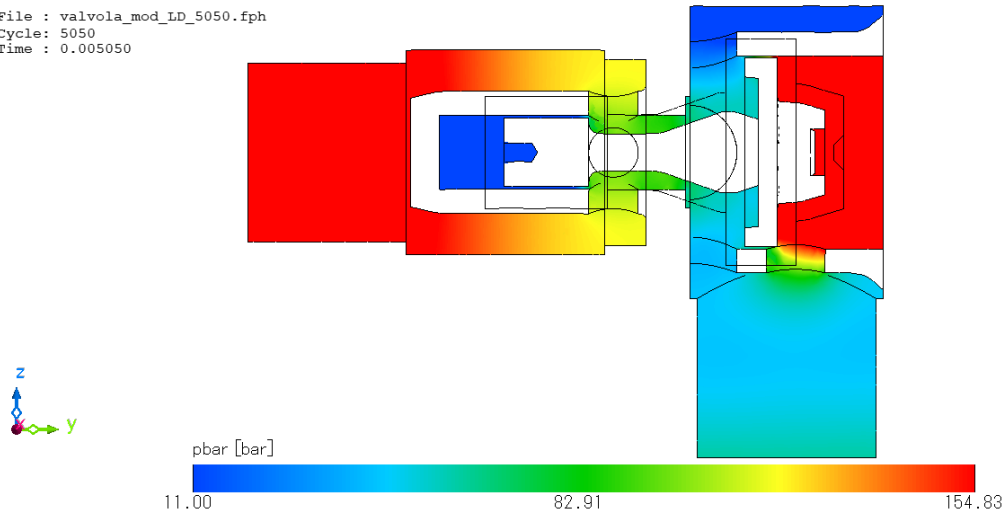


Figure 9.50: Pressure field at end of simulation

File : valvola\_mod\_LD\_5550.fph  
Cycle: 5550  
Time : 0.005550

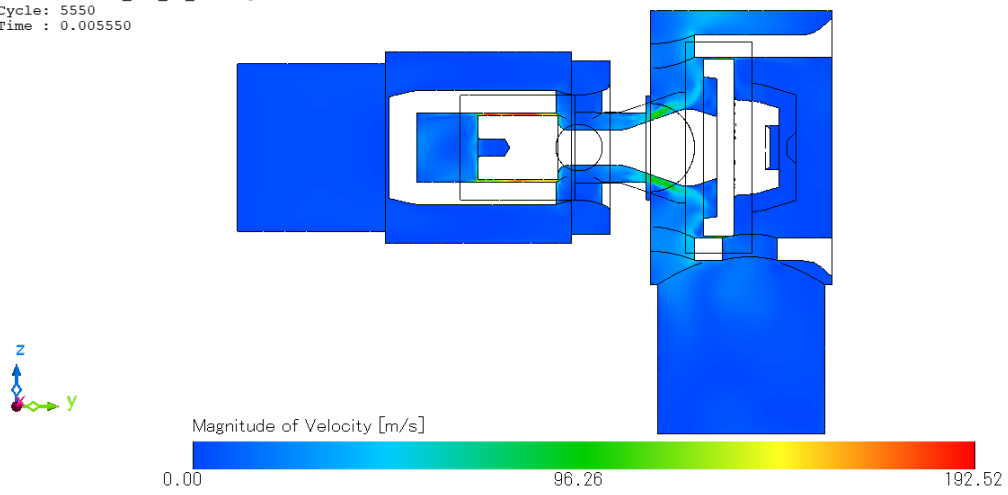


Figure 9.51: Velocity field at end of simulation

File : valvola\_mod\_LD\_5550.fph  
Cycle: 5550  
Time : 0.005550

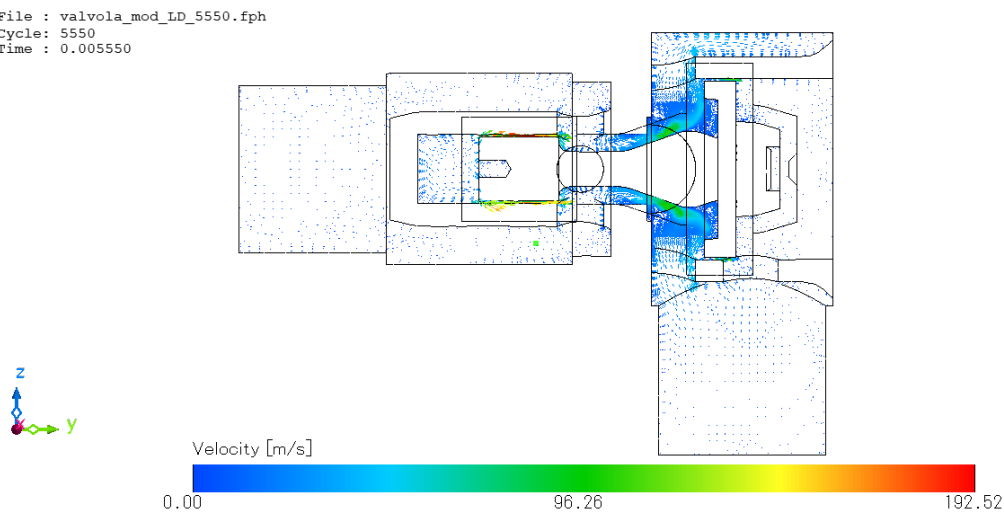


Figure 9.52: Velocity vector field at end of simulation

The results presented represent the pressure and velocity fields in the middle of the full co-simulation, for computational simplicity, that is, when the shutter position begins to stabilize on an equilibrium position. As in Chapter 9.2, it is reasonable to assume that the trend is similar. Thus, through comparison with the results in *Adams*, it can be assumed that the shutter has reached an equilibrium position.

Since the pressure in the outlet has a greater value, in this case the equilibrium position is reached for a smaller displacement in the y-direction than in the case before mentioned. Again, the pressure field, although qualitatively correct, has higher values in the domain part above the shutter.

The velocity field, on the other hand, has a trend that mirrors the real case, and the recirculation vortices generated following the flow reflection on the deflector are noticeable.

# Chapter 10

## Conclusion

This thesis work focused on the preliminary study of the transient opening of a Pressure Relief Valve, until an equilibrium position is reached.

Since in a hydraulic valve, the stable operating conditions are significantly influenced by the flow forces, when there is a consistent pressure drop across the valve, the force experienced at a steady state increases proportionally with the displacement of the spool, as a virtual spring. Moreover, the force magnitude is affected by the velocity of the fluid where it exits the deflector, which relies on the extent of jet dispersion.

The study was carried out by coupling CFD and multibody simulation software; the approach, which is innovative from an engineering perspective, is to use co-simulation software, MSC Cosim, to couple the two software scFLOW and *Adams*.

As a first step, models were made individually in the above two software. In *Adams*, a spring-damper system was inserted in order to allow movement of the moving part, calculating its preload based on the cracking pressure, at which the valve opens; later, kinematic and flexible couplings were added, allowing for displacements in appropriate directions, constraining others. Coupling with fluid dynamics is performed through the use of a Gforce, whose force and displacement components are filled during the course of co-simulation by the forces of pressures calculated by CFD simulation software.

In scFLOW, the geometry used on *Adams* itself was imported, simplifying elements that were redundant or could cause errors during the calculation. We then define a background domain, which is fixed for the duration of the calculation, and an Overset domain, which is defined around the moving part and follows its movement during the calculation. Since realizing a co-simulation using incompressible fluids can result in instability, very small compressibility was assigned to the fluid to avoid instability in the calculation. In addition,  $k - \varepsilon$  was used as the turbulence model. Finally, to verify that the results obtained from the simulation were independent of the discretization, i.e., the grid, a convergence analysis was performed using a stationary simulation with the  $k - \varepsilon$  and  $k - \omega$  models to check the correctness of the model.

Comparison with the benchmark results [1] shows some discrepancies that may be due to several factors in combination:

- The data used to perform the simulations are different; in particular, in the case of the present work, a liquid-gas interface is not convolved, which would require specific models to handle the effects of this interface and to take into account, for example, if the gas possesses different thermal or chemical properties than the liquid, that can influence temperature distribution and the concentration of chemical species. As a result, cavitation models or

related features are not used to account for the formation and collapse of vapor bubbles in the fluid.

- The simulation was performed by establishing a fixed time step during the calculation. Although this is the most efficient method of performing the co-simulation, as a result of the field evolution during the transient, it is possible that numerical stability was violated toward the end of the simulation. A possible correction would be to impose the number of CFL and automatically adapt the time step to the calculation.
- 

Despite the aforementioned discrepancies, it is crucial to recognize the value of this approach in providing a multidisciplinary perspective and capable of integrating different software simultaneously. The present work falls within this context, and the main objective of this research was to assess the potential and limitations of this methodology, using the case of a Pressure Relief Valve as an example. Various factors, such as system complexity, model approximations, and simulation conditions, can affect the accuracy of the results. Thus, this study aims to contribute to the understanding and effective application of this methodology, highlighting its potential and outlining areas that require further development and investigation, and is merely a starting point for analysis that can be the subject of further study and investigation in the future.

# Bibliography

- [1] G. Altare, M. Rundo, and M. Olivetti. *3D dynamic simulation of a flow force compensated pressure relief valve*. 2016.
- [2] Atos. *Pressure relief valve type ARE Catalogue - Table C020-15/E*.
- [3] J. F. Blackburn, G Reethof, and J. L. Shearer. *Fluid Power Control*. MIT, 1960.
- [4] J. Boussinesq. *Théorie analytique de la chaleur mise en harmonie avec la thermodynamique et avec la théorie mécanique de la lumière: Refroidissement et chauffage par rayonnement, conductibilité des tiges, lames et masses cristallines, courants de convection, théorie mécanique de la lumière*. 1903.
- [5] B. Chaouat. *The State of the Art of Hybrid RANS/LES Modeling for the Simulation of Turbulent Flows*. ONERA, France, 2017.
- [6] M. Hellemans. *The Safety Relief Valve Handbook - Design and Use of Process Safety Valves to ASME and International Codes and Standards*. 2009.
- [7] B. Hussein, D. Negrut, and A. A. Shabana. *Use of the Implicit HHT-I3 and the Explicit ADAMS Methods with the Absolute Nodal Coordinate Formulation*. 2007.
- [8] F. Kadar and G. Stepan. *Nonlinear dynamics and safety aspects of pressure relief valves*. Budapest University of Technology and Economics, 2023.
- [9] M. Kertész. *The role of the stiffly stable integrators in nonlinear dynamic simulations*. Slovak University of Technology, Bratislava, 2015.
- [10] G. Licskó, A. Champneys, and C. Hős. *Dynamical Analysis of a Hydraulic Pressure Relief Valve*. Proceedings of the World Congress on Engineering 2009 Vol II, 2009.
- [11] F. R. Menter. *Two-Equation Eddy-Viscosity Turbulence Models for Engineering Applications*. NASA Ames Research Center, Moffett Field, California, 1994.
- [12] S. Mukherjee. *Pressure-Relief System Design*. Lurgi India Company Ltd.
- [13] D. Negrut and B. Harris. *ADAMS Theory in a Nutshell*. Department of Mechanical Engineering, The University of Michigan, Ann Arbor, 2001.
- [14] A. Nordsieck. *On Numerical Integration of Ordinary Differential Equations*. 1962.
- [15] CFD online. <https://www.cfd-online.com/Tools/yplus.php>.
- [16] M. R. Osborne. *On Nordsieck's method for the numerical solution of ordinary differential equations*. Australian National University, Canberra, Australia.
- [17] S. Osterland and J. Weber. *Analytical analysis of single-stage pressure relief valves*. Chair of Fluid-Mechatronic Systems, Technische Universität Dresden, Germany, 2019.
- [18] R. R. Ryan. *ADAMS - Multibody System Analysis Software*, in *Multibody Systems Handbook* (Schiehlen W.) Ann Arbor, Michigan, 1990.

- [19] W. Schiehlen. *Multibody System Dynamics: Roots and Perspectives*. Institute B of Mechanics, University of Stuttgart, Germany, 1997.
- [20] P. Smith and R. W. Zappe. *Valve Selection Handbook - Engineering fundamentals for selecting the right valve design for every industrial flow application*. 2004.
- [21] MSC Software. *Adams online help*.
- [22] MSC Software. *Analysis Method in User's Guide for Cradle-scFlow*. 2023.
- [23] MSC Software. *MSC CoSim User's Guide - 2023.1*.
- [24] P.R. Spalart S.R. e Allmaras. *TA One-Equation Turbulence Model for Aerodynamic Flows*. Reno, NV : 30th Aerospace Science Meeting and Exhibit, AIAA Paper 92-0439, 1992.
- [25] M. Tuominen. *Multibody simulations as a product development tool: introduction to ADAMS and two analyses*. Aalto University School of Engineering, 2015.
- [26] P. Wesseling. *Principle of Computational Fluid Dynamics*. Delft University of Technology, 2009.
- [27] D. C. Wilcox. *Turbulence Modeling for CFD - 3rd edition*. DCW Industries, Inc., 2006.
- [28] Q. Zhang and S. Cen. *The coupling methods*, in *Multiphysics Modeling - Numerical Methods and Engineering Applications*. 2016.