UNIVERSITY OF ICELAND

Magister Scientiarum thesis
in Mechanical Engineering

Computational Fluid Dynamics Simulation of a Box Designed to Ventilate Water Mist

Hafþór Helgi Heiðarsson

Advisors: Halldór Pálsson, Ásdís Helgadóttir
and Elías M. V. Siggeirsson

Faculty of Industrial Engineering, Mechanical Engineering and Computer Science

COMPUTATIONAL FLUID DYNAMICS SIMULATION OF A BOX DESIGNED TO VENTILATE WATER MIST

Hafþór Helgi Heiðarsson

30 ECTS thesis submitted in partial fulfillment of a
Magister Scientiarum degree in Mechanical engineering

Advisors
Halldór Pálsson
Ásdís Helgadóttir
Elías M. V. Siggeirsson

Master's Examiner
Ragnar Lárusson

Faculty of Industrial Engineering, Mechanical Engineering and Computer Science
School of Engineering and Natural Sciences
University of Iceland
Reykjavík, December 2022

Computational Fluid Dynamics Simulation of a Box Designed to Ventilate Water Mist
CFD Simulations of a Ventilation Box
30 ECTS thesis submitted in partial fulfillment of a M.Sc. degree in Mechanical engineering

Faculty of Industrial Engineering, Mechanical Engineering and Computer Science
School of Engineering and Natural Sciences
University of Iceland
107, Reykjavik
Iceland

Sími: 525 4000

Bibliographic information:
Hafþór Helgi Heiðarsson, 2022, Computational Fluid Dynamics Simulation of a Box Designed to Ventilate Water Mist, M.Sc. thesis, Faculty of Industrial Engineering, Mechanical Engineering and Computer Science, University of Iceland.

Printing: Háskólaprent, Fálkagata 2, 107 Reykjavík
Reykjavík, Iceland, December 2022

Abstract

Valka developed a fish processing machine called the Valka Cutter, which uses high pressure water to cut away bones and portions fish. This machine has three sections, an x-ray section, a camera section and a cutting section. While operating water mist from the cutting section spreads into other sections of the machine. This can cause problems in the camera section. To avoid this, two air blowers were placed to push the mist from the front to the end where an outlet and a chimney are located. The water mist is drawn out through the chimney, and directed through a box to dispose of the mist and prevent it from entering the surrounding working environment. In this thesis the flow through this box is modeled and analyzed at different speeds, along with an alternative model with a different placement of the outlet pipe, simulated only at top speed. All computational fluid dynamics simulations were done with OpenFOAM. Higher speed showed a more desirable outcome in both models, i.e. less mist left the box through the outlet and rather stayed in the system or left through the bottom. Experimental data is needed to compare to the numerical results and further validate these results.

Útdráttur

Valka þróaði fiskvinnsluvél sem er kölluð the Valka Cutter, sem notar háþrýstivatn til að skera burt bein og hluta af fiskinum. Þessi vél hefur þrjá hluta, röntgen hluta, myndavéla hluta og skurð hluta. Meðan vélin er í rekstri fyllist hún af vatnsþoku frá vatnsskurðsrýminu sem dreifist um alla hluta vélarinnar. Þetta getur skapað vandræði í myndavélarýminu. Til að forðast að vatnþokan komist inn í myndavélarýmið voru settir tveir loftblásarar sem ýta þokunni frá fremri hluta vélarinnar til enda þar sem útflæðið á sér stað. Vatnsþokan er dregin út um strompinn, og svo beint í gegnum hólf til að losna við rakann og koma í veg fyrir að þokan komist inn í vinnslu rýmið. Í þessari ritgerð er loftflæðið gegnum kassann hermt og greint við mismunandi hraða, með úttak á mismunandi stöðum. Allir flæðisreikningar voru framkvæmdir með OpenFOAM. Hærri hraði sýndi betri niðurstöður í báðum tilfellum, þ.e.a.s minni þoka fór út um úttakið og varð frekar eftir í kassanum eða fór út um botninn. Tilraunagögn eru nauðsynleg til að bera saman við tölulegar niðurstöður og sannreyna þessar niðurstöður frekar.

Contents

List of Figures ix

List of Tables xiii

Nomenclature xv

Abbreviation xvii

Acknowledgements xix

1 Introduction 1

2 Background and Theory 3

2.1 Navier Stokes equation 3

2.1.1 Conservation of Mass 4

2.1.2 Conservation of Momentum 4

2.2 Turbulence modeling 5

2.3 Multi-phase flow 6

3 Method 9

3.1 Programs and systems 9

3.1.1 OpenFoam 9

3.1.2 Paraview 12

3.2 Meshing 13

3.3 Solver and boundary conditions 21

3.4 Numerical schemes 24

3.5 Simulation run 25

3.6 PostProcessing 26

4 Results and discussion 27

4.1 Original geometry at the velocity of 8.8m/s 27

4.2 Original geometry at the velocity of 12m/s 29

4.3 Original geometry at the velocity of 15m/s 31

4.4 Newer geometry at the velocity of 15m/s 33
4.5 Discussion 35

5 Conclusion **37**

6 References **39**

List of Figures

1.1 The Valka Cutter. 1

1.2 The box attached to the chimney of the Valka Cutter. 2

3.1 Domain and mesh visualization in paraview for 2D and 3D. 12

a 2D mesh in paraview 12

b 3D mesh in paraview 12

3.2 Original sketch of the domain. 13

3.3 Mesh of the 2D domain. 14

a domain in 2D. 14

b domain in 2D including the blocks. 14

3.4 Probes locations. 15

3.5 All the parts seen in Inventor as surfaces. 16

a Part body seen in Inventor 16

b Part pipe seen in Inventor 16

c Part bottom seen in Inventor 16

d Part inlet on the left outlet on the top right seen in Inventor 16

3.6 Complete view of the entire box along with a cross section view of the box as a surface in Inventor. 17

a Whole box with all the parts together 17

b Cross section view of the whole box with the parts all together 17

3.7 Mesh after running snappyHexMesh 18

3.8 Both geometries used in simulations. 18

a The original geometry 18

b The newer geometry 18

3.9 Mean value of velocity at probe locations over different time frames 20

a 10-20 seconds. 20

b 20-30 seconds. 20

c 10-30 seconds. 20

4.1 Final particles distribution for the original model with 8.8 m/s flow. 28

a 500 particles a second injected for 9 seconds at the speed of 8.8 m/s. 28

b 5000 particles a second injected for 9 seconds at the speed of 8.8 m/s. 28

4.2 Final particles distribution for the original model with 12 m/s flow. 30

a 500 particles a second injected for 9 seconds at the speed of 12 m/s. 30

b 5000 particles a second injected for 9 seconds at the speed of 12 m/s. 30

4.3 Final particles distribution for the original model with 15 m/s flow. 32

a 500 particles a second injected for 9 seconds at the speed of 15 m/s. 32

b 5000 particles a second injected for 9 seconds at the speed of 15 m/s. 32

4.4 Final particles distribution for the alternative model with 15 m/s flow. 34

a 5000 particles a second injected for 9 seconds at the speed of 15 m/s. 34

b 5000 particles a second injected for 9 seconds at the speed of 15 m/s. 34

List of Tables

3.1 Boundary conditions for k for the velocity of 8.8 m/s 22

3.2 Boundary conditions for ν_t 22

3.3 Boundary conditions for ω 23

3.4 Boundary conditions for p 23

3.5 Boundary conditions for U for the velocity of 8.8 m/s 23

4.1 Final particle distribution for the original model with 8.8 m/s flow. 29

4.2 Particle distribution for 12 m/s original geometry. 31

4.3 Particle distribution for 15 m/s original geometry. 33

4.4 Particle distribution for 15 m/s newer geometry. 34

4.5 Particle distribution for all cases. 35

4.6 Particle distribution in % for all cases. 36

Nomenclature

d_p	Pipe diameter (m).
g	The acceleration of gravity (m^2/s).
I	Turbulence intensity (%).
k	Turbulence kinetic energy (m^2/s^2).
l	Turbulent length scale (m).
ν_t	Turbulent viscosity (m^2/s).
ω	specific turbulent dissipation rate (1/s).
p	Pressure (Pa).
ρ	Density (kg/m^3).
μ	constant viscosity coefficient (Pa s).
$\overline{\overline{\tau}}$	viscous stress tensor (N/m^2).
U	Velocity (m/s).
$\overrightarrow{v}$	Velocity (m/s).
∇	gradient operator (1/m).

Abbreviation

ASCII	American Standard Code for Information Interchange
CFD	Computational Fluid Dynamics
LES	Large eddy simulation
LPT	Lagrangian particle tracking
MPPIC	Multi-Phase Particle In Cell
OpenFOAM	Open Source Field Operation and Manipulation
PISO	Pressure-Implicit with Splitting of Operators
RANS	Reynolds-averaged Navier-Stokes
SIMPLE	Semi-Implicit Method for Pressure Linked Equations
STL	Stereolithography
q8.8	Original model with 5000 particles a second for 9 second at the flow speed of 8.8 m/s
q8.8L	Original model with 500 particles a second for 9 second at the flow speed of 8.8 m/s
q8.12	Original model with 5000 particles a second for 9 second at the flow speed of 12 m/s
q8.12L	Original model with 500 particles a second for 9 second at the flow speed of 12 m/s

q15	Original model with 5000 particles a second for 9 second at the flow speed of 15 m/s
q15L	Original model with 500 particles a second for 9 second at the flow speed of 15 m/s
Y15	Newer model with 5000 particles a second for 9 second at the flow speed of 15 m/s
Y15L	Newer model with 500 particles a second for 9 second at the flow speed of 15 m/s

Acknowledgements

This thesis was only possible with the help of Valka and their representative, Dr. Elías Siggeirsson. I would like to express my deepest appreciation to Valka and Elías for his advice and input across all the stages of this project.

I would also like to extend my sincere thanks to my professors Dr. Ásdís Helgadóttir and Dr. Halldór Pálsson for their patience, support and guidance throughout the workings of this project. Lastly, I would like to thank my family for their continuous and endless support across my studies.

1 Introduction

Valka is an innovation company that develops and supplies cutting-edge machinery and automation products for the fish processing sector. It strives to achieve maximum efficiency, profitability, and competitiveness by collaborating closely with forward-thinking seafood businesses. Valka does this by designing, producing and servicing high-tech processing systems.[1]

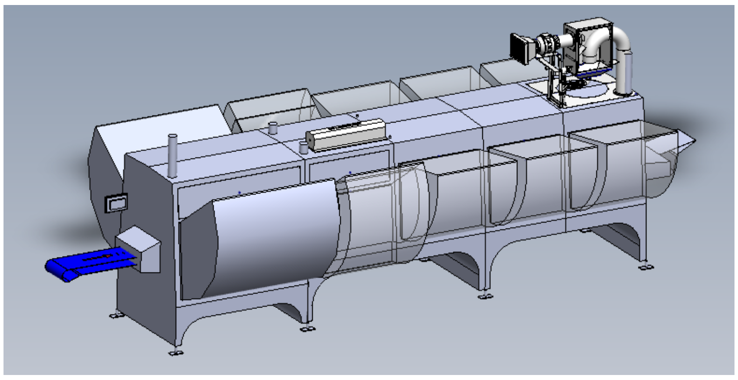


Figure 1.1: The Valka Cutter.

Valka developed a fish processing machine called the Valka Cutter. The Valka Cutter uses high pressure water to cut away bones and portion fish.
The machine consists of three stages. An X-ray is the first stage, which is used to detect and locate bones, mainly the pinbones and any bones not detected by manual labor. The X-ray detection takes place on a soft conveyor belt which has negligible effects on the functionality of the x-ray equipment. To be able to cut on the conveyor belt, it needs to be made of strong material. This is solved in the second stage by using a steel conveyor belt, however this requires a transfer from the soft belt to the steel belt. During this transfer the fish fillet might move or deform resulting in a different position of the bones after the transfer. To solve this problem, a 3D camera is used to locate the fillet on the steel belt and, if there are any changes, the fillet is relocated to calculate the correct position of the bones on the steel belt. The 3D camera is composed of a laser and a camera which detects

the 2D projection and the height of the fillet. Additionally, the information is used to calculate the weight of the fillet from an estimated density and the measured volume. In the third, and final stage the information from the first two stages is used to control the water cutter to optimally remove the bones and portion the fillet.
When the machine is operating, the water from the high pressure cutters dissolves into mere particles when it hits the steel belt. These particles create a mist within the whole machine. The camera in the second stage is sensitive to this water mist as it can get dense, resulting in fog on a protective glass in front of the camera or blocking the laser itself. To ensure precision and maximum productivity the camera section or the second stage must be as clear as possible. To avoid the water mist getting into the camera section a set of air blowers were placed to generate a overpressure in the camera section, pushing the water mist back to the third section and out through a chimney, located near the outlet of the machine. From the chimney the airflow is directed through a box to dispose of the majority of the water mist to prevent it from entering the surrounding working environment.

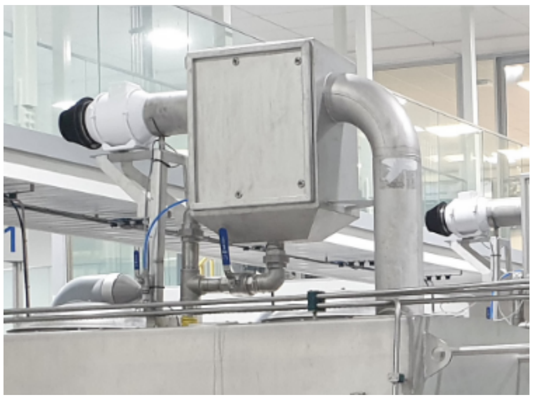


Figure 1.2: The box attached to the chimney of the Valka Cutter.

In this project the flow through the box connected to the chimney is modeled and analyzed with computational fluid dynamics, which is the simulation of fluid engineering systems using numerical methods.
The reason CFD was choosen here is because of its relative inexpense, in contrast to experimentation [9].

2 Background and Theory

Computational fluid dynamics (CFD) is the technique of mathematically simulating a fluid flow-related physical event and then numerically resolving it [5]. Computational fluid dynamics has become a more widely used technology for creating fluid flow solutions as a result of computers and their computational capacity. A CFD software study looks at fluid flow in relation to its physical characteristics, such as velocity and pressure. These features need to be taken into account simultaneously in order to practically produce an accurate solution for a physical phenomenon connected to fluid flow [5].

OpenFoam is the CFD software used in this case to conduct the fluid flow analysis. OpenFoam uses the Finite Volume Method [12].

2.1 Navier Stokes equation

The Navier-Stokes equations are based on the conservation law of physical properties of fluid [8]. Those equations are the three laws of conservation:

- Conservation of Mass - Continuity equation
- Conservation of momentum - Newton's Second Law (Momentum equation)
- Conservation of Energy - First Law of Thermodynamics (Energy equation)

The principle of conservational law is the change of properties Mass/momentum/energy in an object is decided by the input and output [5].

The system is isothermal so the temperature remains constant and therefore the energy equation was excluded in this project. By applying the mass and momentum conservation, the continuity equation and momentum equation can be derived as shown in following subsections.

2.1.1 Conservation of Mass

If the density is constant the flow is assumed to be incompressible and the continuity equation is defined as:

$$\nabla \cdot \overrightarrow{v} = 0 \tag{2.1}$$

2.1.2 Conservation of Momentum

The Navier-Stokes equations, that describe the motion of viscous fluid domains [9][5].

The balance of forces in the flow is described by:

$$\underbrace{\frac{\delta}{\delta t}(\rho \overrightarrow{v})}_{\text{I}} + \underbrace{\nabla \times (\rho \overrightarrow{v}\,\overrightarrow{v})}_{\text{II}} = \underbrace{-\nabla p}_{\text{III}} + \underbrace{\nabla \overline{\overline{\tau}}}_{\text{IV}} + \underbrace{\rho \overrightarrow{g}}_{\text{V}} \tag{2.2}$$

where p is the static pressure, $\overline{\overline{\tau}}$ is the viscous stress tensor and $\rho \overrightarrow{g}$ is the gravitation force per volume.

The stress tensor in incompressible flow is defined as:

$$\tau_{ij} = \mu \nabla^2 \overrightarrow{v} \tag{2.3}$$

however if the fluid is incompressible with constant viscosity coefficient μ is assumed constant the Navier-Stokes equations is simplified to:

$$\rho \frac{D\overrightarrow{v}}{Dt} = -\nabla p + \mu \nabla^2 \overrightarrow{v} + \rho \overrightarrow{g} \tag{2.4}$$

The momentum equation is split into terms marked with Roman letters underneath. The terms and what they represent is displayed below.

- I - Local change with time
- II - Momentum convection
- III - Pressure force
- IV - Molecular-dependent momentum exchange (Diffusion term)
- V - Mass force

2.2 Turbulence modeling

Turbulence is the chaotic movement of fluid flows. When the flow's speed or characteristic length is increased, the convective forces in the flow overcome the viscous forces of the fluid, and the previously laminar flow becomes turbulent. The Reynolds number is the ratio of convective to viscous forces. This number is used to classify flows, the higher the Reynolds number, the more turbulent the flow [6].

Three approaches for modeling turbulence are generally looked at, two of which apply to this thesis and will therefore be described below.

- Large eddy simulation (LES) - Models the details of the largest scales of turbulent movements directly, however the smaller scales are treated with a sub-grid model. This method does require substantial amount of computational power which makes it impractical in high Reynolds number problems.
- Reynolds average Navier Stokes (RANS) - Models the turbulence explicitly with the help of additional field variables. The computational complexity is similar to laminar problems, however the turbulence modeling is in no way uniform or reliable.

2.3 Multi-phase flow

There are several combinations of multi-phase flows possible, such as [12]:

- Gas and liquid
- Gas and solid particles
- Liquid and Solids
- Gas, liquid and solids

In this thesis gas-solid multi-phase flows are relevant as the water mist or droplets will be models as solids to preform lagrangian particle tracking on them and therefore those will be further described here. The particles are assumed to be rigid and spherical.

Gas-solid multi-phase flows are relatively common in numerous industrial processes as well as in various natural phenomena. To optimize the design and operation of industrial processes which involve gas-solid flows a thorough understanding of gas-solids flow is needed. There are many different types of CFD models available for the prediction of gas-solid flows. Every model has its own advantages and disadvantages. Therefore the user has to chose one solver over another depending on the prioritized factors. When dealing with modeling of gas-solids flows there are two frequently used approaches. Those are the Eulerian-Eulerian and the Eulerian-Lagrangian methods. In Eulerian-Eulerian approach all the phases are treated as continuous phases whilst in Eulerian-Lagrangian approach the fluid phase is treated as a continuous phase, however the solid phase is treated as a discrete phase [14].
In this study the Euler-Lagrangian method was chosen to be able to preform Lagrangian particle tracking (LPT) with a chosen solver to determine where the particles of the mist would exit the model. The Eulerian-Langrangain approach can provide analysis of flows with a wide range of particle types, size distribution, shapes and even velocities. However the traditional Lagrangain model does have some limitations when it comes to explicitly tracking particle-particle and particle-wall collisions. When a dense system with a large number of particles is tracked the calculation of particle-particle interactions is extremely complex. To improve particle-particle and particle-wall interactions some methods have been developed with improvements in calculations. [14]

3 Method

3.1 Programs and systems

In this project several programs and systems were used. Those programs are OpenFoam, ParaView and also Inventor.
As well as the subsystem for linux within windows called Ubuntu to run OpenFoam.

- Ubuntu - subsystem for Linux within Windows to run OpenFoam.
- Inventor - to draw the geometries in 3D.
- OpenFoam - To do CFD numerical computations.
- Paraview - To visually see the mesh and the numerical computations from OpenFoam.
- Matlab - to display the number of particles that exit through the outlet and bottom.

3.1.1 OpenFoam

OpenFOAM stands for Open-source Field Operation And Manipulation and is an open-source CFD software [3]. It uses C++ scripts to create specialized numerical solvers and pre-/post-processing tools to address continuum mechanics problems. OpenFOAM v2012 was used to mesh, model and simulate the problems in this project.

The first step is to create a mesh. This mesh can be created in blockMeshDict as was done initially when looking at a 2D version of the part. A surface mesh can also be made in another program such as Inventor and then exported and and used in OpenFoam. The mesh itself is the computational domain or the 3D volume of interest that is divided into small cells. Next the boundary conditions are set, which are necessary to solve the conservation equations using the built mesh mentioned above [10].

The solver MPPICFoam was utilized in this project. Lagrangian particle tracking (LPT) is a feature of the Lagrangian solver MPPICFoam, which can be used to represent particles in continuum. This solver represents collisions without resolving particle-particle interactions by simulating the solid phase as parcels. The diameter, density, restitution coefficient, and friction coefficient are used to describe the particles, which are considered to be rigid and spherical. This solver utilizes two time steps, the fluid flow time step, which is used for continuous phase solution, and the particle tracking time step, which determines the time step for particle injection [4].

To make a case three main directories are needed, the 0 directory, the constant directory and the system directory. These 3 directories will be described below along with their sub folders.

0 folder (Time directory) - In this file are the initial values of the solution variables along with the boundary conditions for each part.

- k - Turbulence kinetic energy (m^2/s^2).
- ν_t - Turbulent viscosity (m^2/s).
- ω - Specific turbulent dissipation rate (1/s).
- p - Pressure ($kg/m/s^2$).
- U - Velocity (m/s).

Specific turbulence dissipation rate **Constant (constant directory)** - Physical setup and data for meshing

- extendedFeatureEdgeMesh - a subdirectory which holds additional information about the feature edges, such as distinction between region edges and purely geometric features (only appears after running surfaceFeature-

Extract).

- PolyMesh - a subdirectory which holds the complete description of the case mesh (only appears after running blockMesh).
- triSurface - a subdirectory which holds the .stl files of the structure made in Inventor.
- kinematicCloudProperties - a subdirectory which holds the particle settings.
- g - gravitational acceleration.
- transportProperties - a subdirectory which holds the transport properties.
- turbulenceProperties - holds the turbulence modeling approach.

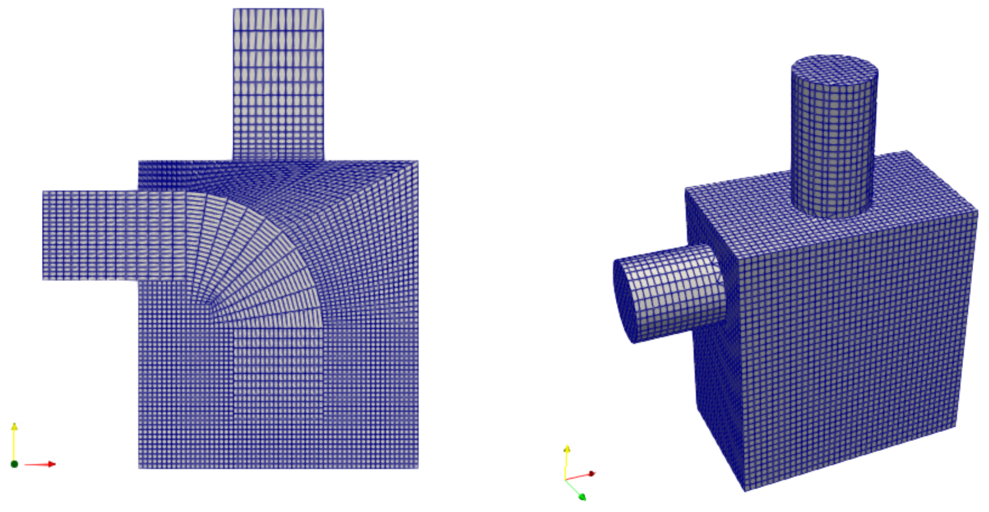
System (System directory) - for setting parameters associated with the solution procedure itself (System files and case manipulation).

- blockMeshDict - A hexagonal mesher (used in this project to create a mesh around the box so an internal mesh of the box could be constructed).
- controlDict - run control parameters are set such as start/end times and parameters for data output.
- decomposeParDict - used to decompose the problem between multiple cores and holds the core amount you are splitting the problem between (6 cpu cores were used in this project).
- fvSchemes - where discretisation schemes used in the solution may be selected at run-time.
- fvSolution - where the equation solvers, tolerances and other algorithm controls are set for the run.
- probes - location of the probes to see the U at varies places throughout the run (results shown in postprocessing after run).
- snappyHexMeshDict - Mesh controls are set in this folder.

- surfaceFeatureExtractDict - holds entries for each extraction process. The name of the individual dictionary is used to load the input surface and also the basename for the output.

3.1.2 Paraview

Paraview is an open-source, multi-platform data analysis and visualization application which user can quickly and effectively build visualizations to analyze their data using qualitative and quantitative techniques [2].
Paraview version 5.9.0 was used for all visualization of the problem, that includes the mesh itself pre-simulation as well as the post-processing of the CFD problem. This was done in both 2D and 3D versions. On figure 3.1 there is a demonstration of the visualization that can be seen of the mesh for both 2D and 3D.



(a) 2D mesh in paraview *(b)* 3D mesh in paraview

Figure 3.1: Domain and mesh visualization in paraview for 2D and 3D.

3.2 Meshing

The original drawing for the model of the domain was provided by Valka and is displayed below in figure 3.2. The black lines represent walls, green line is the inlet and the red line is the outlet of the domain. The dimensions of the box are as follows 500mm x 550mm x 300mm. The middle of the inlet pipe is located at 135mm from the top and runs from the inlet to the middle of the box to 70mm from the bottom. The outlet is located in the middle of the box and therefore has 170mm on both sides. Both the inlet and outlet pipes are 160mm in diameter. The domain was first modeled in 2D for simplicity and to gain a deeper understanding of the model and find a suitable solver for the project.

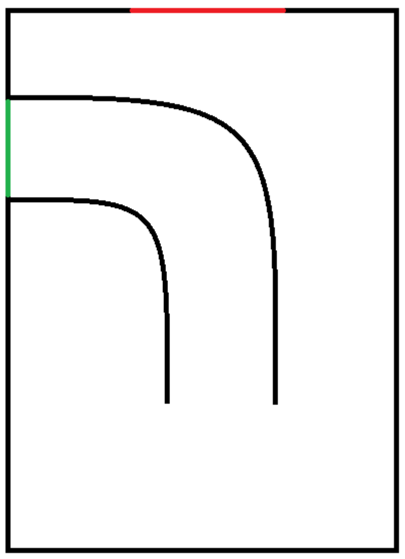
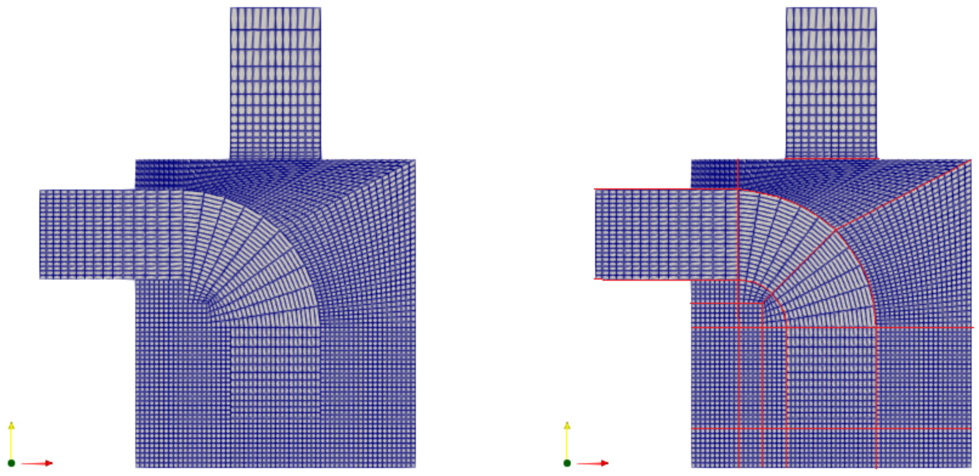


Figure 3.2: Original sketch of the domain.

The first steps when making a mesh is drawing up the domain. The the domain is split into squares or four corner field, with each corner marked with a number or referencing point. Then these points are set up in the blockMeshDict. This domain consisted of 22 blocks. However the pipe needs some curved edges. A reference point of (0.85 2.25 0) was chosen as the middle point, followed by the middle points in the curve which were found with the simple formulas of $0.85+r\cdot\cos 45°$ and $0.85+r\cdot\sin 45°$ for the inner curve and 2.45 for the outer curve from the radius of the curves. An inlet and outlet pipes outside of the box were added to the original sketch for a more realistic view of the boxs geometry. For the finite volume grid not to run into problems a few things had to be kept in mind such as large aspect ratios and abrupt changes needed to be refined with the mesh grading. This enables refinement of the mesh towards a specific location. Below in figure 3.3 the 2D domain can be seen normally as well as how the blocks were split within it.



(a) domain in 2D. *(b)* domain in 2D including the blocks.

Figure 3.3: Mesh of the 2D domain.

After the mesh was constructed four probes were placed in the mesh to gather data on the velocity at these certain points within the box. The first probe was located at the start of the outlet pipe, 2nd probe in a neutral position and probes 3 and 4 were located in the inlet pipe. Probe 3 in a straight section after the curve and probe 4 at the exit of the inlet pipe. Figure 3.4 below show the probes positions.

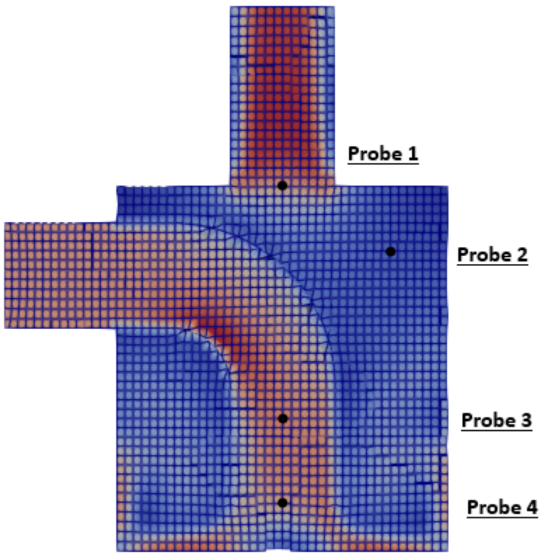
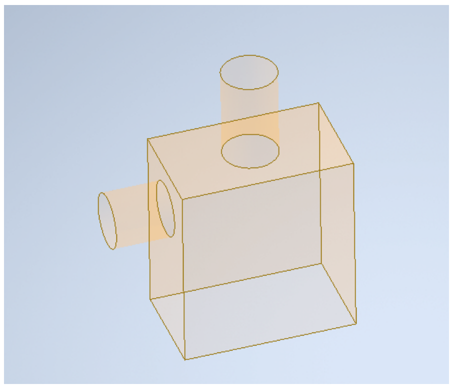
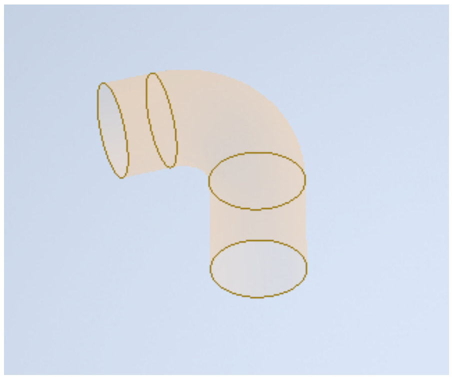
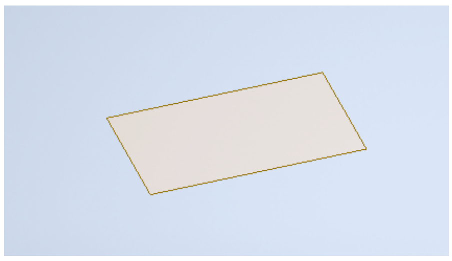


Figure 3.4: Probes locations.

After having seen how the model behaved in 2D and gaining a little more understanding of the model it was moved into 3D. Autodesk Inventor Professional 2023 was chosen to create the geometry due to the difficulties that would appear with trying to create the mesh in blockMeshDict, especially the pipe. First the box was drawn up, then the 2 holes made in to for the inlet and outlet. The extrude tool was used to create the outer inlet and outlet pipes. The internal pipe was done by creating a 3D sketch inside the box as a line then then use the sweep tool to make it into a pipe with the diameter of 160mm. To make this easier to place the half section view in Inventor came in handy so that the pipe would be in the correct position. Lastly the outer inlet and outlet pipes were closed with patches to make the system completely closed. All the parts were drawn as a surface i.e. the wall do not have any thickness. This is done so that the wall thickness does not interfere with the simulation in Openfoam. These surfaces were then split into 5 parts for boundary conditions. The box walls excluding the bottom along with the outer pipes were stitched together with the Stitch tool and renamed body, the inner pipe surface from the sweep tool was renamed to pipe and lastly the bottom, inlet and outlet patches were named respectfully bottom, inlet and outlet. These 5 parts were then exported from Inventor as .stl files in the ASCII format with the resolution on high. These files were then placed in the triSurface folder located in the constant folder. In figure 3.5 all the 5 parts can be seen before they were exported from Inventor.


(a) Part body seen in Inventor


(b) Part pipe seen in Inventor


(c) Part bottom seen in Inventor

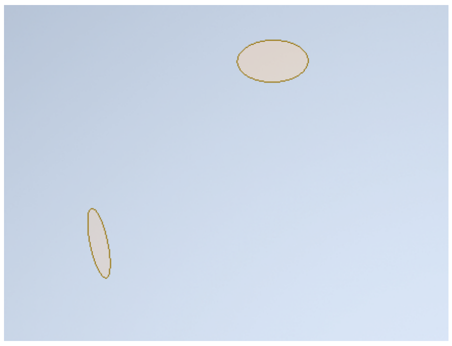
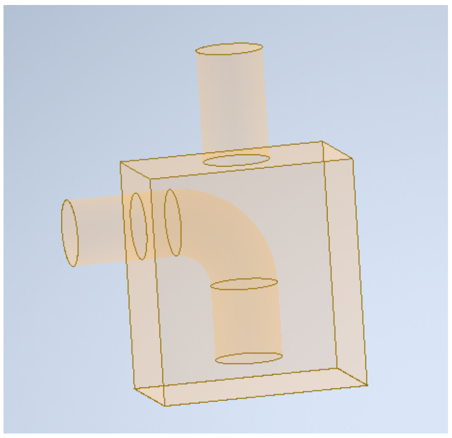

(d) Part inlet on the left outlet on the top right seen in Inventor

Figure 3.5: All the parts seen in Inventor as surfaces.

Figure 3.6 shows the box fully constructed with all the part together to make the whole geometry, along with a cross section view.


(a) Whole box with all the parts together

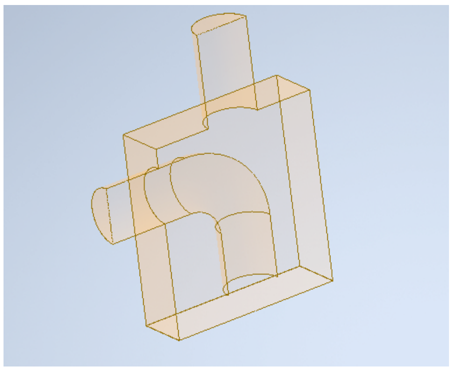

(b) Cross section view of the whole box with the parts all together

Figure 3.6: Complete view of the entire box along with a cross section view of the box as a surface in Inventor.

Meshing in 3D differs from 2D in the sense that instead of the domain being in the blockMeshDict the domain is in the triSurface folder as stated before and the blockMeshDict becomes a background mesh for snappyHexMesh. The overall domain bounding box is (-0.42 -5e-9 -0.15)(0.25 0.82 0.15) so the blockMeshDict must have a geometry that is larger and can fit the overall domain bounding box within it. In this project a 1200mm x 1120mm x 500mm box was constructed. The cells count was 48 cell in the x-axis, 40 cells in the y-axis and 20 cells in the z-axis in the coarsest mesh of the block. Now a point has to be chosen in the background mesh. The point can be chosen either outside the domain to see how things can effect it from the outside. This is often used to simulate wind effects on a race car for example or inside the domain like we want in this case to see the flow inside the box. This is selected in the snappyHexMeshDict in the castellatedMeshControls section which is the Dictionary to control the castellation functionality of snappyHexMesh. The locationInMesh has to be within the domain so that the internal mesh after running snappyHexMesh show the desired domain. In this case the location in mesh was selected at (0.17 0.28 0) which is a point within the box and therefore after running snappyHexMesh the domain changes from a simple box image to the complete geometry shown in figure 3.7.

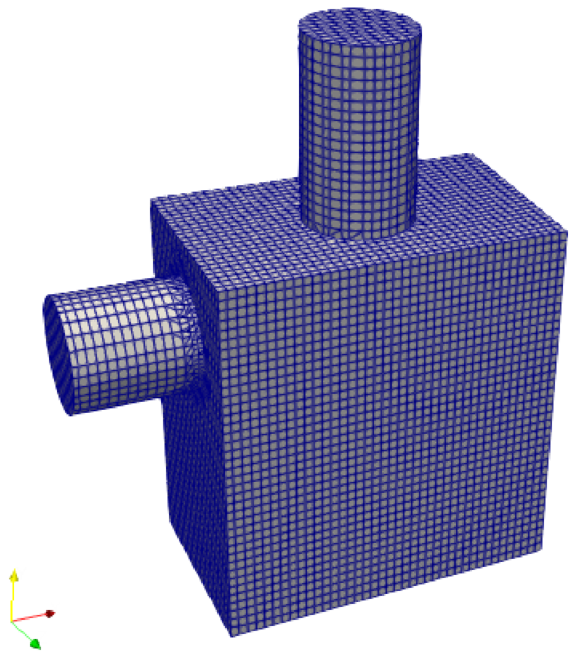
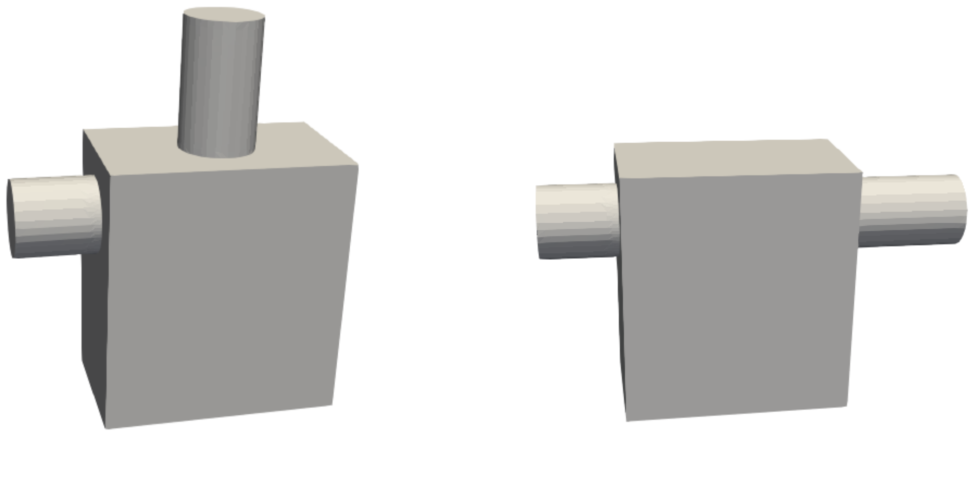


Figure 3.7: Mesh after running snappyHexMesh

In this project two different geometries, both of whom can be seen below in figure 3.8.



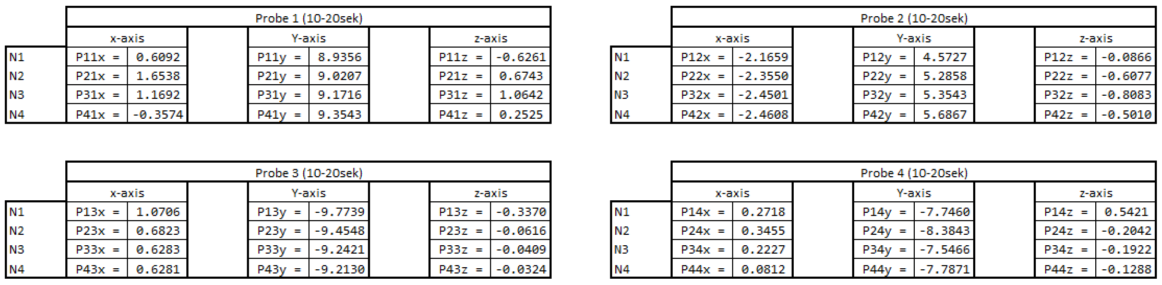
(a) The original geometry

(b) The newer geometry

Figure 3.8: Both geometries used in simulations.

Sensitivity study

To determine the fineness of the mesh, a mesh sensitivity study was performed. This study was done in the early stages of the 3D modeling and was first attempted with SimpleFoam and then PimpleFoam. PimpleFoam is a large time-step transient solver for incompressible flow, using the PIMPLE algorithm [11]. A case was set up to run PimpleFoam for a total of 30 seconds with only air in the system and use the probes placed in the system to track the velocity U to determine the quality of the mesh. Four mesh densities were made ranging from N=1 which was described earlier as the coarses mesh to N=2, N=3 and lastly the finest mesh N=4. Figure 3.9 below shows the results from the mean value of the velocity in all directions for 3 different times. Firstly 10 to 20 second, secondly 20 to 30 seconds and lastly the time of 10 to 30 seconds. The time from 0 to 10 seconds was excluded in the mean values since that's the time where the system was more unstable and including those values would skew the results.



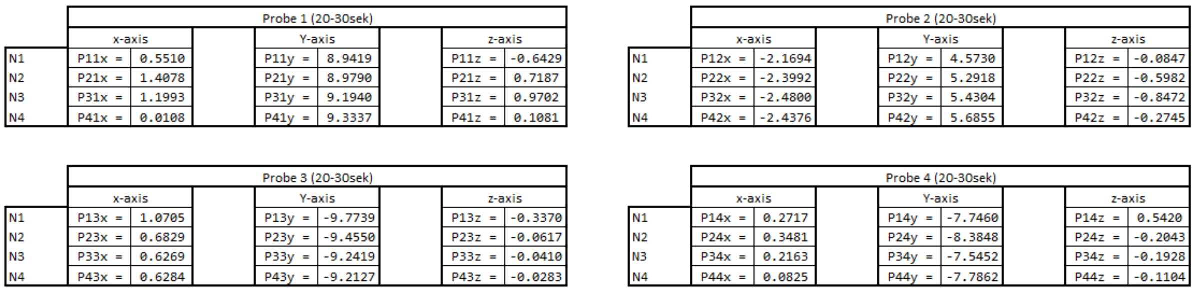
	Probe 1 (10-20sek)							
	x-axis			Y-axis			z-axis	
N1	P11x =	0.6092		P11y =	8.9356		P11z =	-0.6261
N2	P21x =	1.6538		P21y =	9.0207		P21z =	0.6743
N3	P31x =	1.1692		P31y =	9.1716		P31z =	1.0642
N4	P41x =	-0.3574		P41y =	9.3543		P41z =	0.2525

	Probe 2 (10-20sek)							
	x-axis			Y-axis			z-axis	
N1	P12x =	-2.1659		P12y =	4.5727		P12z =	-0.0866
N2	P22x =	-2.3550		P22y =	5.2858		P22z =	-0.6077
N3	P32x =	-2.4501		P32y =	5.3543		P32z =	-0.8083
N4	P42x =	-2.4608		P42y =	5.6867		P42z =	-0.5010

	Probe 3 (10-20sek)							
	x-axis			Y-axis			z-axis	
N1	P13x =	1.0706		P13y =	-9.7739		P13z =	-0.3370
N2	P23x =	0.6823		P23y =	-9.4548		P23z =	-0.0616
N3	P33x =	0.6283		P33y =	-9.2421		P33z =	-0.0409
N4	P43x =	0.6281		P43y =	-9.2130		P43z =	-0.0324

	Probe 4 (10-20sek)							
	x-axis			Y-axis			z-axis	
N1	P14x =	0.2718		P14y =	-7.7460		P14z =	0.5421
N2	P24x =	0.3455		P24y =	-8.3843		P24z =	-0.2042
N3	P34x =	0.2227		P34y =	-7.5466		P34z =	-0.1922
N4	P44x =	0.0812		P44y =	-7.7871		P44z =	-0.1288

(a) 10-20 seconds.



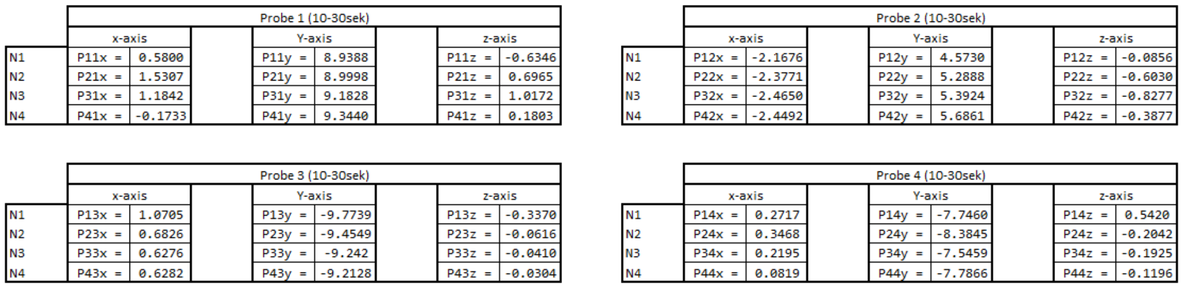
	Probe 1 (20-30sek)							
	x-axis			Y-axis			z-axis	
N1	P11x =	0.5510		P11y =	8.9419		P11z =	-0.6429
N2	P21x =	1.4078		P21y =	8.9790		P21z =	0.7187
N3	P31x =	1.1993		P31y =	9.1940		P31z =	0.9702
N4	P41x =	0.0108		P41y =	9.3337		P41z =	0.1081

	Probe 2 (20-30sek)							
	x-axis			Y-axis			z-axis	
N1	P12x =	-2.1694		P12y =	4.5730		P12z =	-0.0847
N2	P22x =	-2.3992		P22y =	5.2918		P22z =	-0.5982
N3	P32x =	-2.4800		P32y =	5.4304		P32z =	-0.8472
N4	P42x =	-2.4376		P42y =	5.6855		P42z =	-0.2745

	Probe 3 (20-30sek)							
	x-axis			Y-axis			z-axis	
N1	P13x =	1.0705		P13y =	-9.7739		P13z =	-0.3370
N2	P23x =	0.6829		P23y =	-9.4550		P23z =	-0.0617
N3	P33x =	0.6269		P33y =	-9.2419		P33z =	-0.0410
N4	P43x =	0.6284		P43y =	-9.2127		P43z =	-0.0283

	Probe 4 (20-30sek)							
	x-axis			Y-axis			z-axis	
N1	P14x =	0.2717		P14y =	-7.7460		P14z =	0.5420
N2	P24x =	0.3481		P24y =	-8.3848		P24z =	-0.2043
N3	P34x =	0.2163		P34y =	-7.5452		P34z =	-0.1928
N4	P44x =	0.0825		P44y =	-7.7862		P44z =	-0.1104

(b) 20-30 seconds.



	Probe 1 (10-30sek)							
	x-axis			Y-axis			z-axis	
N1	P11x =	0.5800		P11y =	8.9388		P11z =	-0.6346
N2	P21x =	1.5307		P21y =	8.9998		P21z =	0.6965
N3	P31x =	1.1842		P31y =	9.1828		P31z =	1.0172
N4	P41x =	-0.1733		P41y =	9.3440		P41z =	0.1803

	Probe 2 (10-30sek)							
	x-axis			Y-axis			z-axis	
N1	P12x =	-2.1676		P12y =	4.5730		P12z =	-0.0856
N2	P22x =	-2.3771		P22y =	5.2888		P22z =	-0.6030
N3	P32x =	-2.4650		P32y =	5.3924		P32z =	-0.8277
N4	P42x =	-2.4492		P42y =	5.6861		P42z =	-0.3877

	Probe 3 (10-30sek)							
	x-axis			Y-axis			z-axis	
N1	P13x =	1.0705		P13y =	-9.7739		P13z =	-0.3370
N2	P23x =	0.6826		P23y =	-9.4549		P23z =	-0.0616
N3	P33x =	0.6276		P33y =	-9.242		P33z =	-0.0410
N4	P43x =	0.6282		P43y =	-9.2128		P43z =	-0.0304

	Probe 4 (10-30sek)							
	x-axis			Y-axis			z-axis	
N1	P14x =	0.2717		P14y =	-7.7460		P14z =	0.5420
N2	P24x =	0.3468		P24y =	-8.3845		P24z =	-0.2042
N3	P34x =	0.2195		P34y =	-7.5459		P34z =	-0.1925
N4	P44x =	0.0819		P44y =	-7.7866		P44z =	-0.1196

(c) 10-30 seconds.

Figure 3.9: Mean value of velocity at probe locations over different time frames

When looking at the numbers in figure 3.9 it can be seen that for probe 2 and 3 the value between N=2 and N=4 are not so different, however when looking at probe 1 and 4 some noticeable differences can be seen for the x- and z-axis. These probes are placed in the middle of the start of the outlet pipe and middle of the end of the inlet pipe where there are more fluctuations in the flow. Probe 1 and 4 get caught in a velocity change field which does not occur in the coarser meshes, therefore making the difference remarkably high. When looking at the computational power and the time difference between N2 and N4 it was quite substantial. N4 takes around 5 times longer. When looking at the other probes and taking all this into consideration N2 looked like a sufficient mesh fineness and could be simulated in far shorter time with little drawback. Also considering the need to add particles into the system a lot of trial and error was yet to come so a shorter running mesh like N2 was the clear choice.

Now a new solver was need since PimpleFoam does not have lagrangian particle tracking capabilities and therefore the lagrangian solver MPPICFoam was chosen. MPPICFoam also runs the PIMPLE algorithm as well and therefore the mesh sensitivity study would have some weight even though it had a different turbulence model. To check if the difference between N2 and N4 behaved the same in MPPICFoam it resulted in N4 taking 6 times longer then N2 with the estimated times being 60000 seconds compared to 10500 seconds.

3.3 Solver and boundary conditions

MPPICFoam was the primary solver in this project. MPPICFoam is a transitory solver for the simultaneous transport of a solitary kinematic particle cloud that takes into account the impact of particle volume fraction on the continuous phase. Without resolving particle-particle interactions, collisions are represented using multi-phase particle in cell (MPPIC) models. [4]. This solver uses the PIMPLE algorithm, which is the combination of the PISO and SIMPLE algorithms. PISO stands for the Pressure Implicit with Splitting of Operators while SIMPLE stands for the Semi-Implicit Method for Pressure Linked Equations [3]. This solver uses parcels or particle clouds to be able to represent a large amount of particles without too much computational power however in this project the parcelBasisType was chosen as fixed and the nParticles was set as 1 so each parcel or particle cloud would only have 1 particle.
To model this case the cyclone tutorial case was altered to fit the needs for this project. The boundary conditions which are set in the 0 folder are used for the air in the solver. k was calculated with the following equation [15][7]:

$$k = \frac{3}{2}(UI)^2 \tag{3.1}$$

where U is the velocity and I is the turbulence intensity which was decided to be 10% in this project.

The specific turbulent dissipation rate was calculated with the following equation [7]:

$$\omega = C_\mu^{\frac{3}{4}} \frac{k^{\frac{1}{2}}}{l} \tag{3.2}$$

Where C_μ is the turbulence model constant which has the value 0.09, k is the turbulent energy and l is the turbulent length scale.

The turbulent length scale was calculated with the following equation [16][17]:

$$l = 0.07 d_p \tag{3.3}$$

Where d_p is the pipe diameter.

The boundary conditions can be seen in tables 3.1-3.4 below

Table 3.1: Boundary conditions for k for the velocity of 8.8 m/s

BoundaryField	Type	Value
Body	kqRWallFunction	1.1616
Bottom	kqRWallFunction	1.1616
Outlet	inletOutlet	1.1616
Inlet	fixedValue	1.1616
Pipe	kqRWallFunctions	1.1616
pipe_slave	kqRWallFunctions	1.1616

Table 3.2: Boundary conditions for ν_t

BoundaryField	Type	Value
Body	nutkWallFunction	uniform 0
Bottom	nutkWallFunction	uniform 0
Outlet	calculated	uniform 0
Inlet	calculated	uniform 0
Pipe	nutkWallFunctions	uniform 0
pipe_slave	nutkWallFunctions	uniform 0

Table 3.3: Boundary conditions for ω

BoundaryField	Type	Value
Body	omegakWallFunction	175.691
Bottom	omegakWallFunction	175.691
Outlet	inletOutlet	175.691
Inlet	fixedValue	175.691
Pipe	omegakWallFunction	175.691
pipe_slave	omegakWallFunction	175.691

Table 3.4: Boundary conditions for p

BoundaryField	Type	Value
Body	fixedFluxPressure	uniform 0
Bottom	fixedFluxPressure	uniform 0
Outlet	FixedValue	uniform 0
Inlet	fixedFluxPressure	uniform 0
Pipe	fixedFluxPressure	uniform 0
pipe_slave	fixedFluxPressure	uniform 0

Table 3.5: Boundary conditions for U for the velocity of 8.8 m/s

BoundaryField	Type	Value
Body	noSlip	
Bottom	noSlip	
Outlet	pressureInletOutletVelocity	uniform (0 0 0)
Inlet	fixedValue	uniform (8.8 0 0)
Pipe	noSlip	
pipe_slave	noSlip	

TransportProperties
The transportProperties for the continuousPhase air was rho.air = 1.2 and the transportModel Newtonian with nu set to 1.5e-5.

TurbulenceProperties
In this project two different turbulence models were used. For the mesh sensitivity study in PimpleFoam the RANS approach was selected since it has less needs for computational power then the other methods. For the main simulations in MPPICFoam the turbulenceProperties were of the type LES with the LESModel kEqn. The LES turbulence kinetic energy equation is given by [13]:

$$\frac{D}{Dt}(\rho k) = \nabla \times (\rho D_k \nabla k) + \rho G - \frac{2}{3}\rho k \nabla \times u - \frac{C_e \rho k^{1.5}}{\Delta} + S_k \tag{3.4}$$

where the default value for C_e is 1.048 and C_k 0.094.

Particle settnings
The particle settings are located in the kinematicCloudProperties folder within the constant directory. The particles were water and had the constantProperties of $\rho_0 = 1000$. ParticleForces effecting the particles were ErgunWenYDrag - alphac alpha.air and gravity. The particles were injected with patchInjection of the inlet with the parcelBasisType as fixed with nParticles = 1 with the amount of either 500 or 5000 particles a second for 9 seconds that started at 1 second into the simulation. The speed of the particles was set as the same as the inlet speed and the size distribution was of the type normal with a normalDistribution of expectation 10e-6, variance 5e-6, minValue 0.1e-6 and maxValue 50e-6, these size distributions were provided by Valka.

3.4 Numerical schemes

The fvSchemes folder is located in the systems directory and that is used to choose the numerical shemces for the simulations.

- ddtSchemes: Euler
- Gradient schemes (∇) : Gauss linear
- divSchemes ($\nabla\times$)
  - div(alphaPhi.air,U.air): Gauss linearUpwindV unlimited;
  - div(((alpha.air*nuEff.air)*dev2(T(grad(U.air))))): Gauss linear;
  - div(phiGByA,kinematicCloud:alpha): Gauss linear;
  - div(alphaPhi.air,epsilon.air): Gauss limitedLinear 1;
  - div(alphaPhi.air,k.air): Gauss limitedLinear 1;

- LaplacianScheme (∇^2) : Guass linear corrected
- interpolationSchemes: linear
- snGradSchemes: corrected

ControlDict

The controlDict dictionary is used to specify the main case controls. This includes the timing information, write format and optional libraries that can be loaded at run time such as the probes used in this project. In this dictionary the start time was set to 0 and end time at 10 seconds, with deltaT at 0.0002 and the writing interval at 0.1. The write precision and time precision were set at 6.

3.5 Simulation run

Now that the Mesh was ready and the boundary conditions set. The next step was to run the simulation in the Ubuntu terminal and the commands were as follows:

- **Rename 0 to 0.org** - Prevents snappyHexMesh from interfering with it.
- **blockMesh** - Creates background mesh for snappyHexMesh
- **surfaceFeatureExtract** - For the mesher to know where to snap to
- **decomposePar** - Divides the mesh into section per CPU core (in this case 6 CPU cores)
- **mpirun -np 6 snappyHexMesh -overwrite -parallel** - Euns the mesher in parallel on 6 CPU cores
- **reconstructParMesh - constant** - Puts the mesh back together
- **checkMesh** - Shows the quality of the mesh
- **delete all processor folders** - Clear old mesh data
- **rename folder 0.org to 0** - Reactivate the folder for the solver to use

- **decomposePar** - Puts the solution setup into a folder per CPU core
- **mpirun -n 6 renumberMesh -overwrite -parallel** - Optimises the mesh
- **mpirun -np 6 MPPICFoam -parallel** - Runs the solver in parallel
- **reconstructPar** - Puts the CPU results back together into one
- **ParaView** - Opens the results and displays it visually to investigate and analyze

3.6 PostProcessing

The post processing from the simulations was done in Paraview, python and matlab. Inside the kinematicCloudProperties under cloudFunctions two patchPostProcessing were added to track how many particles left the domain from the bottom and from the outlet, Those values were written down in text folders from 0.1-10 with a total of 100 text files that were combined using python and then uploaded into matlab to add the vectors together to get the total values. Lastly Paraview was used to see how many particles stayed in the system. This was vital to track the ratio of particles that were left in the system, that left through the outlet and lastly through the bottom. Since the main goal was to find the best solution so that the majority of the particles would leave through the bottom and not the outlet into the workspace.

For the mesh the number of levels was selected to be one meaning it would be at level 0, so that no cells during castellation were split, however during postprocessing it was noticed that the inlet and pipe had the level increased by one resulting in level 0 and level 1 with refinements around the pipewalls.

4 Results and discussion

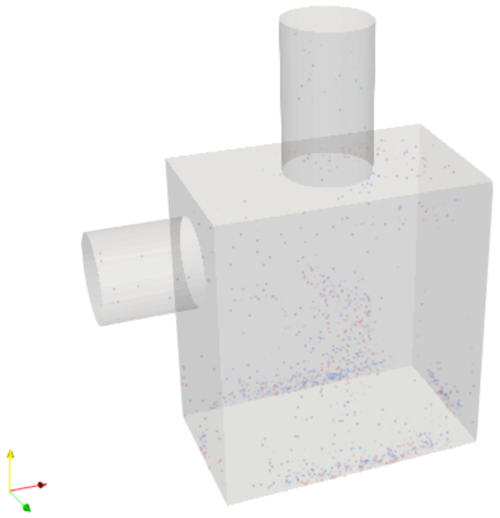
The results from the simulation on the box are displayed below both in figures and tables for all the speeds and models done in this project. As stated earlier in this project two different geometries were used and a total of 8 cases were set up and simulated. 6 cases with the first geometry and then 2 more added with the newer geometry. These cases were set up with different speed and different number of particles to see the ratio of where the particles would exit the system at different velocities:

- two cases at 8.8 m/s, one with 500 particles a second for 9 second and the other with 5000. Total of 4500 particles and 45000 particles.
- two cases at 12 m/s, one with 500 particles a second for 9 second and the other with 5000. Total of 4500 particles and 45000 particles.
- two cases at 15 m/s, one with 500 particles a second for 9 second and the other with 5000. Total of 4500 particles and 45000 particles.
- two cases at 15 m/s with the newer geometry, one with 500 particles a second for 9 second and the other with 5000. Total of 4500 particles and 45000 particles.

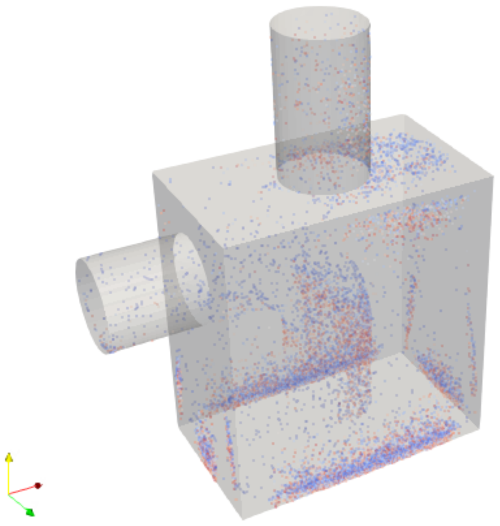
4.1 Original geometry at the velocity of 8.8m/s

Below in figure 4.1 the simulations from the 8.8 m/s speed case can be seen at the end of the simulation or at 10 seconds, with part (a) being for the 500 particles a second and part (b) for the 5000 particles per second. In part (a) there are not many particles to be seen as the total injected was only 4500 however we can see that the particles that remain in the system do seem to get stuck around the corners, just above the bottom and in the inside pipe. Same can be seen in part

(b) but since there were 10x more particles injected in that simulation it can be seen more clearly. There does seem to be a decent amount of particles still floating around the box, this might be due to the speed not being great enough and thus the particles have not settled fully yet.



(a) 500 particles a second injected for 9 seconds at the speed of 8.8 m/s.



(b) 5000 particles a second injected for 9 seconds at the speed of 8.8 m/s.

Figure 4.1: Final particles distribution for the original model with 8.8 m/s flow.

The results for where the particles ended up in the 8.8 m/s cases is displayed in table 4.1 below.

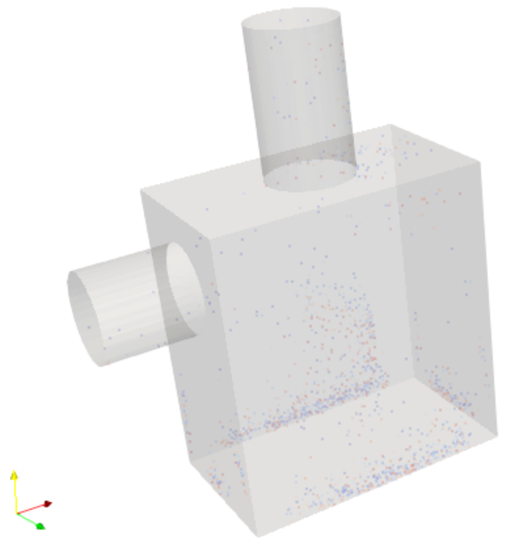
Table 4.1: Final particle distribution for the original model with 8.8 m/s flow.

Model	nParticles	Inside	Outlet	Bottom	Sum	Injected
q8.8L	500	1677	1017	1815	4509	4500
q8.8	5000	17181	10108	17709	44998	45000

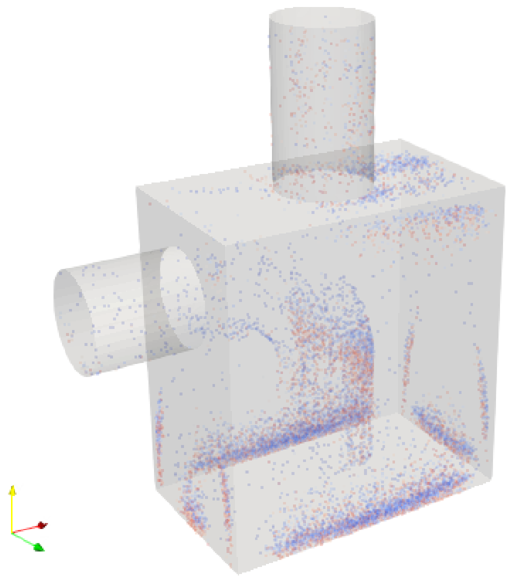
In table 4.1 the ratio between how many particles stay in the system and how many exit the system through either the outlet or the bottom are rather similar. The 5000 particle model does however seem to have around 1% more particles left in the system while that 1% seems to go into the bottom in the 500 particle version. The ratio of the particles that exit through the outlet is almost identical which does give a good indication that the differences are mostly due to the number of particle differences. It can be seen that the total particles or sum and the injected are not completely the same and that can be due to the fact that the case starts injection at 1 second from the start and does inject for 9 second and the simulation also stops at 10 sec so those 2 missing could come from not being registered in the system yet and there are "missing" from what would be expected. The same goes for the extra particles, they could come from the sudden stop of not having fully settled and therefore be registered as exiting the bottom/outlet and still be also registered in the system actively counting them twice. Since these extra/missing particles are such a small part of the whole this issue was deemed negligible.

4.2 Original geometry at the velocity of 12m/s

Below in figure 4.2 the number of particles left inside the system after 10 seconds for the 12 m/s cases can be seen, with part (a) being for the 500 particles a second and part (b) for the 5000 particles a second. Little can be seen in part (a) however in part (b) we can see it more clearly and the particles that are left in the box seem to get stuck around the edges from the wall to the bottom plate, with some getting stuck to the top and in the pipe as was seen before in the 8.8 cases. In this case the particles do seem to be more settled and not as many floating around in the box as in the previous case.



(a) 500 particles a second injected for 9 seconds at the speed of 12 m/s.



(b) 5000 particles a second injected for 9 seconds at the speed of 12 m/s.

Figure 4.2: Final particles distribution for the original model with 12 m/s flow.

The results for where the particles ended up in the 12 m/s cases is displayed in table 4.2 below.

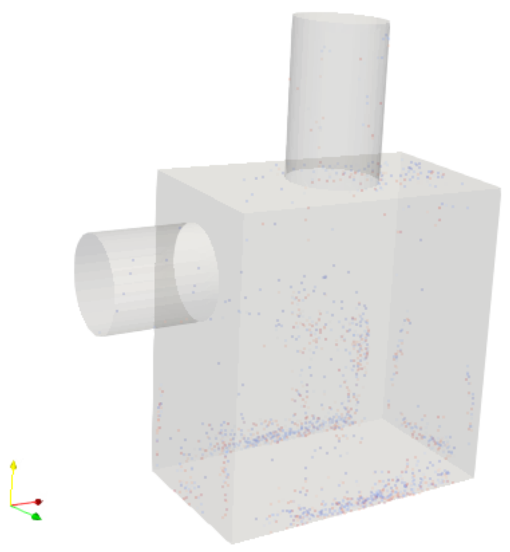
Table 4.2: Particle distribution for 12 m/s original geometry.

Model	nParticles	Inside	Outlet	Bottom	Sum	Injected
q12L	500	1744	840	1911	4495	4500
q12	5000	17566	8349	19083	44998	45000

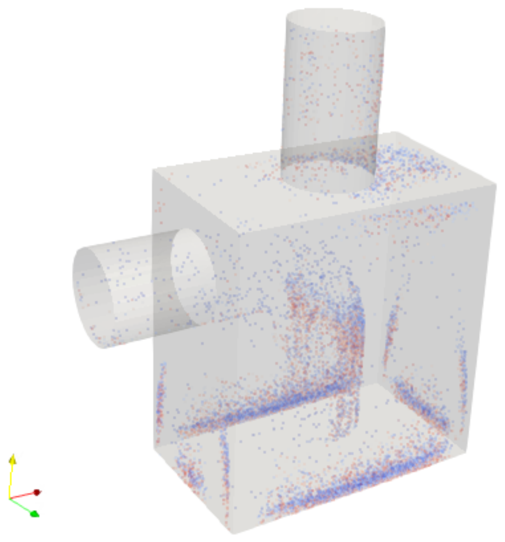
In table 4.2 the ratio between how many particles stay in the system and how many exit out from either the outlet or the bottom can be seen between the two 12 m/s cases and here the results are even more similar then the case before. These cases are the closest ones together out of all the cases, with only a 0.2% difference, where 0.2% more particles stay in the system in the 5000 particle version compared to the 500. Of those 0.2% extra particles that escape in the 500 particle version half or 0.1% leave through the outlet and the other half through the bottom. Here again there are a few particles missing and as was said before is likely due to the injection time and the total simulation run time.

4.3 Original geometry at the velocity of 15m/s

Below in figure 4.3 the number of particles left inside the system after 10 seconds for the 15 m/s original geometry cases can be seen, with part (a) being the 500 particles per second and part (b) being the 5000 particles per second. Again little can be seen in part (a) however in part (b) the particles do seem to distribute similar to the previous cases although now the particles do seem to be more in the lower end of the domain the previously. And in these cases the particles seem even more settled which is probably due to the higher speed pushing the particles faster and harder into place.



(a) 500 particles a second injected for 9 seconds at the speed of 15 m/s.



(b) 5000 particles a second injected for 9 seconds at the speed of 15 m/s.

Figure 4.3: Final particles distribution for the original model with 15 m/s flow.

The results for where the particles ended up in the 15 m/s original geometry cases is displayed in table 4.3 below.

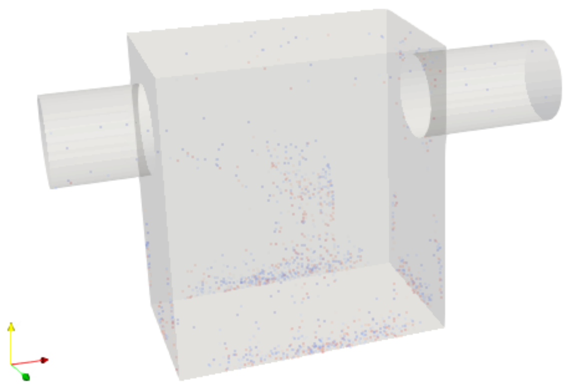
Table 4.3: Particle distribution for 15 m/s original geometry.

Model	nParticles	Inside	Outlet	Bottom	Sum	Injected
q15L	500	1699	754	2072	4525	4500
q15	5000	17391	7276	20331	44998	45000

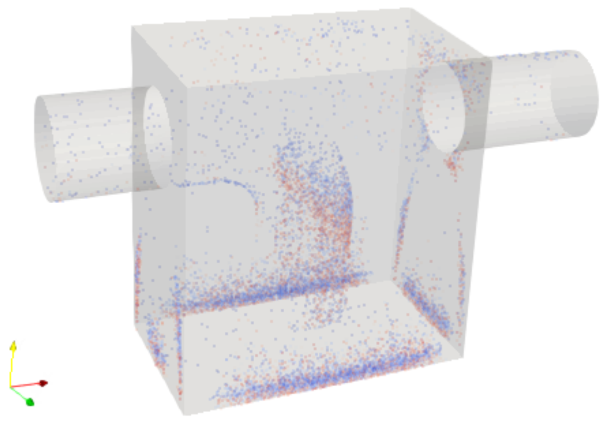
In table 4.3 the ratio between how many particles stay in the system and how many exit out from either the outlet or the bottom can be seen between the two 15 m/s original geometry cases, the results are similar to the 4.1 cases of q8.8 and q8.8L. Around 1.1% more of the total particles stay in the system in the 5000 particle version. Of those 1.1% more particles that leave q15L, about 0.6% leave through the bottom and the rest through the outlet. Here again there are a few particles missing and as was said before is likely due to the injection time and the total simulation run time.

4.4 Newer geometry at the velocity of 15m/s

Below in figure 4.4 the number of particles left inside the system after 10 seconds for the 15 m/s newer geometry cases can be seen with part (a) being the 500 particle per second and part (b) being the 5000 particles per second. The particles do seem to distribute almost identical to original cases at 15 m/s speed except if looked at really closely they do seem to be fewer particles left in the system.



(a) 5000 particles a second injected for 9 seconds at the speed of 15 m/s.



(b) 5000 particles a second injected for 9 seconds at the speed of 15 m/s.

Figure 4.4: Final particles distribution for the alternative model with 15 m/s flow.

The results for where the particles ended up in the 15 m/s newer geometry cases is displayed in table 4.4 below.

Table 4.4: Particle distribution for 15 m/s newer geometry.

Model	nParticles	Inside	Outlet	Bottom	Sum	Injected
Y15L	500	1636	813	2052	4501	4500
Y15	5000	16508	8207	20283	44998	45000

In table 4.4 the ratio between how many particles stay in the system and how many exit from either the outlet or the bottom can be seen between the two newer geometry 15 m/s cases. These results are slightly different from the earlier cases in the fact that there are less particles left in the system but also that in this geometry the cases are the second closest together out of all of the cases, only 0.4% difference in particles inside the geometry. Interesting to note that Y15 has 0.1% more particles escape through the outlet compared to Y15L, however Y15L has 0.5% more escape through the bottom resulting in a 0.4% difference. Again here there are a few particles missing or extra and as was said before is likely due to the injection time and the total simulation run time.

4.5 Discussion

Below in table 4.5 the result of where the particles ended up in every case is summarized and displayed numerically and in table 4.6 the same results are displayed in percentages.

Table 4.5: Particle distribution for all cases.

Model	Inside	Outlet	Bottom	Escape	Injected
q8.8L	1677	1017	1815	4509	4500
q12L	1744	840	1911	4495	4500
q15L	1699	754	2072	4525	4500
Y15L	1636	813	2052	4501	4500
q8.8	17181	10108	17709	44998	45000
q12	17566	8349	19083	44998	45000
q15	17391	7276	20331	44998	45000
Y15	16508	8207	20283	44998	45000

Table 4.6: Particle distribution in % for all cases.

Model	Inside	Outlet	Bottom	Escape	Injected
q8.8L	37.2	22.6	40.3	62.9	100.2
q12L	38.8	18.7	42.5	61.8	100.0
q15L	37.5	16.7	45.8	62.5	100.5
Y15L	36.3	18.1	45.6	63.7	100.0
q8.8	38.2	22.5	39.4	61.8	100.0
q12	39.0	18.6	42.4	61.0	100.0
q15	38.6	16.2	45.2	61.4	100.0
Y15	36.7	18.2	45.1	63.3	100.0

By studying the numbers in tables 4.5 and 4.6 it is clear that for the original geometry with higher speed, more particles seem to stay in the system but also the ones that do leave exit through the bottom rather then the outlet which is the desired results. The newer geometry does seem to have the least amount of particles still in the system and the second highest number of particle leave through the bottom with around the same number going out the outlet as the 12 m/s case in the original geometry. Having around 63.3% of the total injected particles escape and only 36.7% remain. Where 45.1% escape through the bottom and only 18.2% through the outlet. This is a good outcome since when the particles leave through the outlet they enter the atmosphere of the workplace and that is not desirable compared to exit through the bottom and get extracted complete from the system. The most desirable outcome is however from the q15 and q15L models or the original models with the speed of 15 m/s. These model have the highest number of particles leave through the bottom, the least leave through the outlet and rather stay in the system and therefore don't potentially harm the workers.

5 Conclusion

In this project CFD models were used to analyze the exit box on the chimney of the Valka cutter to dispose of the water mist generated in the machine. The purpose was to determine where the particles of the mist would exit based on different models and speed. The models solved the incompressible Navier-Stokes equations inside the box as well as particle tracking.

When comparing the results from the different speeds and the addition of the newer model for the box, it was determined that higher speed when entering the box resulted in a more desirable outcome. Both the simulations of the original model and the newer model with the speed of 15 m/s resulted in the most particles being disposed through the bottom of the box, the difference being that in the original model the particles rather stayed in the box compared to the newer model which more exited through the outlet. Looking at the goal of minimizing the amount of particles exiting through the outlet the original model with the speed of 15 m/s was the best choice.

These models were solely done with simulation so there was no experimental data available for comparison to further validate these outcomes to see if they would translate into real life.

The model did not run in it's coarsest state but rather in n=2 which was decided after a mesh sensitivity study and a thought to computation power and time. If there was more time and a stronger computational computer to run these simulation a finer mesh, possibly with more levels as well would likely produce a more accurate result compared to real life. Another thing that could be added is a model to make the particles slide down rather than get stuck to the walls of the box however that was deemed to be to complicated to include in this project.

6 References

[1] Valka. (2022). Retrieved from Valka: https://valka.is/about/ on 08/07/2022.

[2] ParaView. (2022). Retrived from ParaView: https://www.paraview.org/overview/ 10/07/2022.

[3] OpenFoam. (2022). Retrieved from OpenFoam: https://www.openfoam.com/ 10/07/2022.

[4] cfdyna. (2022). Multi-phase flow simulations in OpenFOAM - MULTI-PHASE FLOWS AND DISSCRETE PHASE MODELS. Retrieved from: http://www.cfdyna.com/Home/of _multiPhase.html on 02/08/2022.

[5] Simscale. (2021). What is CFD | Computational Fluid Dynamics? Retrieved from: https://www.simscale.com/docs/simwiki/cfd-computational-fluid-dynamics/what-is-cfd-computational-fluid-dynamics/ on 05/05/2022.

[6] idealsimulations. (2022). Turbulence Models in CFD. Retrieved from: https://www.idealsimulations.com/resources/turbulence-models-in-cfd/ on 10/05/2022.

[7] Simscale. (2021). K-Omega and K-Omega SST. Retrieved from: https://www.simscale.com/docs/simulation-setup/global-settings/k-omega-sst/ on 10/05/2022.

[8] Simscale. (2021). What Are the Navier-Stokes Equations? Retrieved from: https://www.simscale.com/docs/simwiki/numerics-background/what-are-the-navier-stokes-equations/ on 15/05/2022

[9] Wangda Zuo. Introduction of Computational Fluid Dynamics. Retrieved from: http://wwwmayr.informatik.tu-muenchen.de/konferenzen/Jass05/courses/2 /Zuo/Zuo_paper.pdf on 15/05/2022

[10] Ghione, A. (2012). Development and validation of a two-phase CFD model us-

ing OpenFOAM. Retrieved from: https://www.diva-portal.org/smash/get/diva2:579821 /FULLTEXT01.pdf on 17/05/2022

[11] swMATH. (2022). pimpleFOAM. Retrieved from: https://www.swmath.org/software /15529 on 14/09/2022.

[12] Pálsson, H. (2021). Two phase flow - concepts and definitions. Geothermal Well course, University of Iceland lecture.

[13] OpenFoam. (2022). Retrieved from: https://www.openfoam.com/documentation/guides /latest/doc/guide-turbulence-les-k-eqn.html on 10/09/2022

[14] Ariyaratne, W.K. & Manjula, E.V.P.J. & Ratnayake, Chandana & Melaaen, Morten. (2016). CFD Approaches for Modeling Gas-Solids Multiphase Flows – A Review. Retrived from https://ep.liu.se/ecp/142/099/ecp17142099.pdf on 30/06/2022

[15] CFD Online. (2022). Turbulence intensity. Retrieved from: https://www.cfd-online.com/Wiki/Turbulence _intensity on 25/09/2022

[16] CFD Online. (2012). Turbulence length scale. Retrieved from https://www.cfd-online.com/Wiki/Turbulence_length_scale on 15/11/2022

[17] Comsol. (2022). Inlet Values for the Turbulence Length Scale and Turbulent Intensity. Retrieved from https://doc.comsol.com/5.5/doc/com.comsol.help.cfd/cfd_ug _fluidflow_single.06.095.html on 15/11/2022