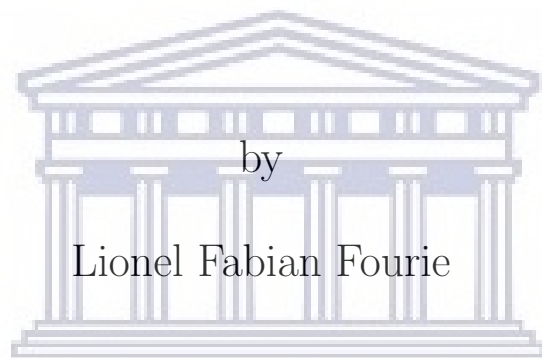


COMPUTATIONAL MODELLING OF A HOT-WIRE CHEMICAL  
VAPOUR DEPOSITION REACTOR CHAMBER



by

Lionel Fabian Fourie

UNIVERSITY *of the*  
WESTERN CAPE

A thesis submitted in fulfilment of the requirements for  
the degree of Magister Scientiae in the Department of Physics,  
University of the Western Cape.

Supervisor: Dr. L. C. Square

Co-Supervisor: Prof. C. J. Arendse

February 2020

<http://etd.uwc.ac.za/>

# Declaration of Authorship

I declare that this thesis titled, 'COMPUTATIONAL MODELLING OF A HOT-WIRE CHEMICAL VAPOUR DEPOSITION REACTOR CHAMBER ' and the work presented in it are my own, that it has not been submitted for any degree or examination in any other university, and that all the sources I have used or quoted have been indicated and acknowledged as complete references.

Full Name: Lionel Fabian Fourie

---

Signed:

---

Date: February 2020

---



UNIVERSITY of the  
WESTERN CAPE

*“I count religion but a childish toy, and hold there is no sin but ignorance”*

Christopher Marlowe



UNIVERSITY *of the*  
WESTERN CAPE

# *Abstract*

## COMPUTATIONAL MODELLING OF A HOT-WIRE CHEMICAL VAPOUR DEPOSITION REACTOR CHAMBER

Lionel Fabian Fourie

M.Sc. Thesis, Department of Physics and Astronomy, University of the Western Cape

**KEYWORDS:** HWCVD, Radiation, OpenFOAM, Heat Transfer,  
buoyantSimpleFoam, rhoSimpleFoam

In this thesis, I explore the subjects of fluid dynamics and the Hot-Wire Chemical Vapour Deposition (HWCVD) process. HWCVD, in its simplicity, is one of the more powerful and elegant deposition techniques available in thin film research which allows for both the growth and post deposition treatments of functional thin films. In the HWCVD process, the quality of the final films is determined by a fixed set of deposition parameters namely: temperature, pressure and the gas flow rate. Finding the optimal combination of these parameters is key to obtaining the desired film specifications during every deposition. Conducting multiple trial experiments to determine said parameters can be expensive and time consuming, this is where simulation methods come into play. One such simulation method is Computational Fluid Dynamics (CFD) modelling.

The aim of this work is to model the HWCVD reactor of the CADAR deposition system within the framework of the open-source CFD software OpenFOAM. The main focus of the work is to develop a model that can be used to predict the conditions inside the HWCVD reactor to determine the optimal set of parameters required for a specific deposition. Known parameters for the production of atomic hydrogen (H) from molecular hydrogen (H<sub>2</sub>) by the HWCVD process are used in the testing of the model, the results of which are used in the development of the generalized model.

At the time of this investigation, experimental results from the CADAR deposition system were unavailable for validation and therefore the results obtained are compared to previously published results that are relevant to the study. The results for pressure distribution inside the HWCVD reactor obtained from the model showed good agreement with the high-vacuum pressure meter attached to the reactor. Where the reading on the pressure meter fluctuated between 9 and 20 Pa, the model showed that the pressure stabilized at 20 Pa ( $2 \times 10^{-1}$  mbar). The temperature distribution both inside the reactor and in the deposition region of the reactor

was investigated as a function of filament temperature ( $T_f$ ) and filament geometry. The results of the temperature investigations were compared to experimental work done by Lee et al. into the effects of filament temperature on the HWCVD method. The model's temperature results showed a similar trend for temperature dissipation inside the reactor with all discrepancies between the two being attributed to the difference in the reactor sizes and the reactor geometries.

February 2020



# *Acknowledgements*

**I would like to thank the following people and organisations without whose assistance, advice and guidance, this thesis would not have been possible.**

Dr. Lynndle Square (Department of Physics, North-West University) for her excellent supervision of this thesis, her guidance, encouragement, understanding, support and her belief in my capabilities.

Prof. Christopher Arendse (Department of Physics, University of Western Cape) for the excellent co-supervision of this thesis and the wealth of experimental knowledge he was willing to share with me as it pertained to this work.

The staff of the Physics Department, University of Western the Cape, for their hospitality, support and encouragement.


My colleagues and fellow Physics Department Postgraduate students, Amy James, Siphelo Ngqoloda, Siphesihle Magubane, Bello Laden and Norman Bowers for the friendship, support and motivation.

My family; father, mother and brother for the understanding, encouragement and the emotional support throughout the course of this work.

To my fiancé, Corrine, THANK YOU for always being there for me when I was at my lowest points throughout the course of this work. I will forever be grateful for all the love and support you have given me during this period. This is as much your work as it is mine. I Love You.

The National Research Foundation (NRF), Armscor and University of the Western Cape for the financial support during this study.

# Contents

<b>Declaration of Authorship</b>	<b>i</b>
<b>Abstract</b>	<b>iii</b>
<b>Acknowledgements</b>	<b>v</b>
<b>List of Figures</b>	<b>ix</b>
<b>List of Tables</b>	<b>xi</b>
<b>Abbreviations</b>	<b>xii</b>
<b>Nomenclature</b>	<b>xiii</b>
 UNIVERSITY of the WESTERN CAPE	
<b>1 Introduction</b>	<b>1</b>
1.1 Background of HWCVD modelling	1
1.2 A brief history of Computational Fluid Dynamics	4
1.3 Objectives	5
1.4 Thesis Outline	6
<b>2 THEORY</b>	<b>7</b>
2.1 Introduction	7
2.2 Governing Equations	7
2.2.1 Conservation of Mass	7
2.2.2 Conservation of Momentum and Navier-Stokes Equations	8
2.2.3 Conservation of Energy	9
2.2.4 Equation of State	9
2.3 Heat Transfer	10
2.3.1 Conduction	10
2.3.2 Convection	11
2.3.3 Radiation	12
2.3.3.1 NoRadiation Model	13
2.3.3.2 Finite Volume Discrete Ordinates Model (fvDOM)	13
2.3.3.3 P1 model	13

<b>3</b>	<b>Computational Fluid Dynamics</b>	<b>15</b>
3.1	Introduction to Computational Fluid Dynamics (CFD)	15
3.2	OpenFOAM	16
3.2.1	Case Structure	17
3.3	Solvers	17
3.3.1	buoyantSimpleFoam	17
3.3.2	rhoSimpleFoam	19
3.4	Numerical Schemes	20
3.5	Solution and Algorithm Control	22
3.5.1	Residuals and Continuity Error	22
3.5.2	fvSolutions	22
3.6	Turbulence Modelling	24
3.6.1	$k - \epsilon$ model	25
<b>4</b>	<b>Modelling Approach</b>	<b>27</b>
4.1	Mesh	27
4.2	Boundary Conditions	31
4.2.1	Group A	31
4.2.1.1	Thermophysical modelling for buoyantSimpleFoam	33
4.2.2	Group B	34
4.2.2.1	Turbulence modelling	35
4.2.2.2	Thermophysical modelling for rhoSimpleFoam	39
4.3	Convergence Criterion	40
<b>5</b>	<b>Results and Discussion</b>	<b>44</b>
5.1	Introduction	44
5.2	Pressure Profile	44
5.2.1	Case 1A	44
5.2.2	Case 2A	46
5.2.3	Case 1B and 2B	47
5.3	Velocity Profile	48
5.4	Temperature Profile	50
5.4.1	Radiation Cases	50
5.4.1.1	Case 1A	50
5.4.1.2	Case 2A	51
5.4.2	Heat Transfer cases	52
5.4.2.1	Case 1B	52
5.4.2.2	Case 2B	54
5.5	Temperature Plots	56
<b>6</b>	<b>Conclusion</b>	<b>59</b>
6.1	Future Work	60
<b>A</b>	<b>checkMesh output</b>	<b>61</b>
A.1	checkMesh output for Straight wire mesh	61
A.2	checkMesh output for Coiled wire mesh	63



<b>B</b>	<b>buoyantSimpleFoam dictionaries</b>	<b>66</b>
B.1	0 folder . . . . .	66
B.1.1	U . . . . .	66
B.1.2	p . . . . .	67
B.1.3	T . . . . .	69
B.1.4	G . . . . .	70
B.2	constant folder . . . . .	72
B.2.1	g . . . . .	72
B.2.2	radiationProperties . . . . .	73
B.2.3	thermophysicalProperties . . . . .	74
B.2.4	turbulenceProperties . . . . .	75
B.3	system folder . . . . .	76
B.3.1	controlDict . . . . .	76
B.3.2	fvSchemes . . . . .	77
B.3.3	fvSolutions . . . . .	79
<b>C</b>	<b>rhoSimpleFoam dictionaries</b>	<b>82</b>
C.1	0 folder . . . . .	82
C.1.1	U . . . . .	82
C.1.2	p . . . . .	84
C.1.3	T . . . . .	85
C.1.4	k . . . . .	87
C.1.5	nut . . . . .	88
C.1.6	epsilon . . . . .	90
C.2	constant folder . . . . .	91
C.2.1	thermophysicalProperties . . . . .	91
C.2.2	turbulenceProperties . . . . .	93
C.3	system folder . . . . .	94
C.3.1	controlDict . . . . .	94
C.3.2	fvSchemes . . . . .	95
C.3.3	fvSolutions . . . . .	97



<b>Bibliography</b>	<b>100</b>
---------------------	------------

# List of Figures

3.1	Overview of OpenFOAM structure [1]	16
3.2	General OpenFOAM case structure [2]	17
3.3	Turbulent flow [3]	24
3.4	RANS, LES and DNS [4]	24
4.1	CADAR HWCVD reactor vessel	28
4.2	Side-view of the 3D CAD model for the coiled wire configuration (left) and straight wire configuration (right).	28
4.3	Top-view of the 3D CAD model for the coiled wire configuration (left) and straight wire configuration (right).	28
4.4	External surface mesh for the HWCVD reactor chamber	29
4.5	External surface mesh for the HWCVD reactor chamber including the inlet	29
4.6	Surface mesh for the substrate holder	30
4.7	Surface mesh for the four parallel wires	30
4.8	Patch Names	31
4.9	The calculation of the cell center of epsilon	38
4.10	A plot of the residuals for Case 1A	41
4.11	A plot of the residuals for Case 1B	41
4.12	A plot of the residuals for Case 2A	42
4.13	A plot of the residuals for Case 2B	42
5.1	Initial pressure for the straight wire configuration (Pa)	45
5.2	Convergence pressure for the straight wire configuration (Pa)	45
5.3	Initial pressure for the coiled wire configuration (Pa)	46
5.4	Convergence pressure for the coiled wire configuration (Pa)	46
5.5	Deposition pressure at Convergence for both sets of wire configurations (Pa)	47
5.6	Velocity and progression of gas through the reactor chamber ( $\text{ms}^{-1}$ )	48
5.7	Initial temperature for straight wire configuration (K)	50
5.8	Convergence temperature for straight wire configuration (K)	51
5.9	Initial temperature for coiled wire configuration (K)	51
5.10	Convergence radiation temperature for coiled wire configuration (K)	52
5.11	Convergence heat transfer temperature for straight wire configuration (K)	53
5.12	Top-view of temperature streamline plots for straight wire configuration (K)	53
5.13	Side-view of temperature streamline plots for straight wire configuration (K)	54
5.14	Convergence heat transfer temperature for coiled wire configuration (K)	54
5.15	Top-view of temperature streamline plots for coiled wire configuration (K)	55
5.16	Side-view of temperature streamline plots for coiled wire configuration (K)	55

5.17 The variation in glass temperature as a function of  $d_{f-s}$  distance and filament temperature[5]. . . . . 56

5.18 Substrate-to-Filament Temperature plots for  $T_f = 1600^{\circ}\text{C}$  . . . . . 57

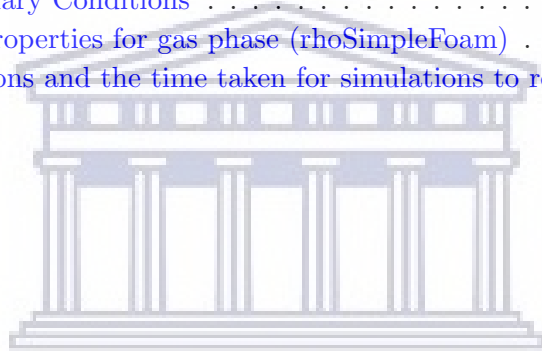
5.19 Substrate-to-Filament Temperature plots for various wire temperatures . . . . . 58



UNIVERSITY of the  
WESTERN CAPE

# List of Tables

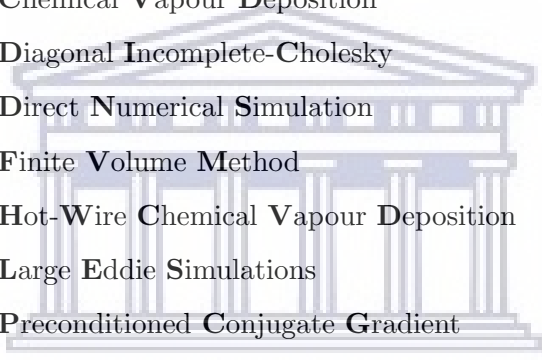
3.1	Group of terms. . . . .	21
3.2	Selected numerical discretization schemes in <i>fvSchemes</i> file . . . . .	21
3.3	Solvers selected in <i>fvSolution</i> . . . . .	23
4.1	Boundary Conditions for buoyantSimpleFoam . . . . .	32
4.2	ThermoPhysicalProperties for gas phase (buoyantSimpleFoam) . . . . .	33
4.3	Boundary Conditions for buoyantSimpleFoam . . . . .	35
4.4	Turbulence Boundary Conditions . . . . .	36
4.5	ThermoPhysicalProperties for gas phase (rhoSimpleFoam) . . . . .	39
4.6	Number of Iterations and the time taken for simulations to reach convergence . . . . .	43



UNIVERSITY *of the*  
WESTERN CAPE

# Abbreviations

<b>CAD</b>	<b>C</b> omputer- <b>A</b> ided <b>D</b> esign
<b>CADAR</b>	<b>C</b> luster <b>A</b> pparatus for <b>D</b> evice <b>A</b> pplications <b>R</b> esearch
<b>CFD</b>	<b>C</b> omputational <b>F</b> luid <b>D</b> ynamics
<b>CVD</b>	<b>C</b> hemical <b>V</b> apour <b>D</b> eposition
<b>DIC</b>	<b>D</b> iagonal <b>I</b> ncomplete- <b>C</b> holesky
<b>DNS</b>	<b>D</b> irect <b>N</b> umerical <b>S</b> imulation
<b>FVM</b>	<b>F</b> inite <b>V</b> olume <b>M</b> ethod
<b>HWCVD</b>	<b>H</b> ot- <b>W</b> ire <b>C</b> hemical <b>V</b> apour <b>D</b> eposition
<b>LES</b>	<b>L</b> arge <b>E</b> ddie <b>S</b> imulations
<b>PCG</b>	<b>P</b> reconditioned <b>C</b> onjugate <b>G</b> radient
<b>RANS</b>	<b>R</b> eynolds- <b>A</b> veraged <b>N</b> avier- <b>S</b> tokes

  
UNIVERSITY of the  
WESTERN CAPE

# Nomenclature

$A$	Cross sectional area	$[\text{m}^2]$
$A_s$	Surface area	$[\text{m}^2]$
$A_{su}$	Sutherland coefficient	
$c_p$	Specific heat capacity at constant pressure	$[\text{Jkg}^{-1}\text{K}^{-1}]$
$c_v$	Specific heat capacity at constant volume	$[\text{Jm}^{-3}\text{K}^{-1}]$
$C_\mu$	Constant	
$\left(\frac{\partial \hat{\mathbf{u}}}{\partial t}\right)_I$	Time derivative due to inviscid fluxes	
$\left(\frac{\partial(\rho \mathbf{u})}{\partial t}\right)_V$	Time derivative due to diffusion	
$d_h$	Hydraulic diameter	$[\text{m}]$
$\nabla \cdot$	Divergence operator	
$e$	Internal energy	$[\text{J}]$
$E$	Energy	$[\text{Jm}^3\text{kg}^{-1}]$
$\hat{E}$	Total energy density	$[\text{J}]$
$G$	Radiation intensity	
$\nabla$	Gradient operator	
$h$	Enthalpy	$[\text{J}]$
$h_c$	Convective heat transfer coefficient	$[\text{Wm}^{-2}\text{K}^{-1}]$
$h_r$	Radiation heat transfer coefficient	$[\text{Wm}^{-2}\text{K}^{-1}]$
$h_{tot}$	Total heat transfer coefficient	$[\text{Wm}^{-2}\text{K}^{-1}]$
$\mathbf{H}_f$	Enthalpy of fusion	$[\text{Jkg}^{-1}]$
$I$	Turbulent intensity	
$k$	Turbulent kinetic energy	$[\text{m}^2\text{s}^{-2}]$
$k_c$	Thermal conductivity	$[\text{Wm}^{-1}\text{K}^{-1}]$
$l$	Turbulent length scale	$[\text{m}]$
$l_T$	Turbulent characteristic length scale	$[\text{m}]$

$n$	Number of moles	
$p$	Pressure	[Pa]
$Q$	Mass flow rate	[m <sup>3</sup> s <sup>-1</sup> ]
$\dot{Q}_{cond}$	Heat flux due to conduction	[W]
$\dot{Q}_{conv}$	Heat flux due to convection	[W]
$\dot{Q}_{rad}$	Heat flux due to radiation	[W]
$r$	Radial coordinate	[m]
$R$	Universal gas constant	[JK <sup>-1</sup> mole <sup>-1</sup> ]
$S_i$	Source term for internal energy	
$S_{ij}$	Mean strain rate tensor	
$T$	Temperature	[K]
$T_c$	Critical temperature	[K]
$T_s$	Surface temperature	[K]
$T_{su}$	Sutherland temperature	[K]
$\mathbf{T}_{visc}$	Viscous stress tensor	[Nm <sup>-2</sup> ]
$T_0$	Reference temperature	[K]
$T_\infty$	Temperature in fluid far from surface	[K]
$u$	Velocity in x-direction	[ms <sup>-1</sup> ]
$u_{mean}$	Mean flow velocity	[ms <sup>-1</sup> ]
$u_T$	characteristic velocity of turbulence	[ms <sup>-1</sup> ]
$\mathbf{u}$	Velocity vector	[ms <sup>-1</sup> ]
$\hat{\mathbf{u}}$	Inviscid momentum density	[kgm <sup>-2</sup> s <sup>-1</sup> ]
$U$	Flow rate	[ms <sup>-1</sup> ]
$v$	Velocity in y-direction	[ms <sup>-1</sup> ]
$V$	Volume	[m <sup>3</sup> ]
$V_m$	Molar volume (volume of 1 mole gas)	[m <sup>3</sup> ]
$w$	Velocity in z-direction	[ms <sup>-1</sup> ]
$y^+$	Dimensionless wall coordinate	
$\alpha$	Thermal diffusivity	[m <sup>2</sup> s <sup>-1</sup> ]
$\epsilon$	Dissipation of turbulent kinetic energy	[m <sup>2</sup> s <sup>-3</sup> ]
$\mu$	Dynamic viscosity	[Pa·s]
$\mu_T$	Dynamic eddy viscosity	[Pa·s]

$\mu_0$	Dynamic viscosity at reference temperature $T_0$	[Pa·s]
$\nu$	Kinematic viscosity	[m <sup>2</sup> s <sup>-1</sup> ]
$\nu_t$	Turbulent kinematic viscosity	[m <sup>2</sup> s <sup>-1</sup> ]
$\rho$	Density	[kgm <sup>-3</sup> ]
$\tau_{ij}$	Reynold stress tensor	[Nm <sup>-2</sup> ]
$\tau_{xx}, \tau_{yx}, \tau_{zx}$	Viscous stress components	[Nm <sup>-2</sup> ]
$\tau_{xy}, \tau_{yy}, \tau_{zy}$		
$\tau_{xz}, \tau_{yz}, \tau_{zz}$		
$\Phi_d$	Dissipation function	
$\omega_a$	Acentric factor	



UNIVERSITY of the  
WESTERN CAPE





*To my late Grandmother:*

*Elizabeth Steyn*

UNIVERSITY *of the*  
WESTERN CAPE

# Chapter 1

## Introduction

### 1.1 Background of HWCVD modelling

Hot-wire Chemical Vapour Deposition (HWCVD), is an elegant low pressure deposition technique used for the deposition of functional thin films. Since its first patent in the early 1980s [6, 7], the HWCVD technique has been improved considerably and is currently a viable method for the deposition of many different functional films, both organic and inorganic.

The HWCVD technique is based on the decomposition of precursor gasses at a catalytic hot surface. The resulting radicals formed from this decomposition interact with the molecules at the surface of the substrate forming the desired film. In principle, the technique is simple and reliable. However, slight changes in any of the deposition parameters can lead to a large number of undesired defects appearing in the material severely compromising the quality of the films. Since repeatability is a key aspect of any experiment, adequate time should be taken to understand the layout and geometry of the HWCVD reaction chamber as well as how different deposition parameters will affect the quality of the resulting film. Unfortunately, conducting multiple trial experiments to achieve this end becomes expensive and time consuming. This is where Computational Fluid Dynamics (CFD) and material modelling becomes an invaluable tool. By developing a functional computer model of the system, the experimentalist obtains the ability to run multiple simulations, using different deposition parameters, to ascertain which would suit their film growth requirements best before conducting any experiment.

The main deposition parameters to consider in the HWCVD technique are: catalytic surface or filament temperature ( $T_f$ ), substrate temperature ( $T_{sub}$ ), number of filaments, or loops if a coil is used, distance between the filaments and substrate ( $d_{f-s}$ ), the overall temperature of the reaction chamber ( $T$ ) and the gas flow rate. The control of these parameters is key to understanding the HWCVD process as well as ensuring high quality films are grown during

every deposition. Numerous efforts, both experimental and computational, has gone towards modelling and, subsequently, understanding how these parameters affect the deposition process.

One of the objectives of this work is to investigate the optimal distance between the filaments and substrate. Pflüger et al.[8] used the Direct Simulation Monte Carlo method for simulations on the large-area deposition of amorphous silicon prepared by the HWCVD technique. In their work, they determined the optimal  $d_{f-s}$  distance using the model results for the production of uniform films which was later confirmed experimentally. Their simulations also confirmed the influence filament geometry has on the uniformity of the films. Based on this finding by Pflüger et al., two different filament geometries are used in the model for this study to deduce the major differences in the deposition parameters which arise from their use.

Sali et al.[9] conducted growth simulations for film thickness uniformity in the HWCVD deposition process. In their work, they considered five types of commonly used filament geometries to identify the best filament geometry for maximum thickness uniformity. From their results, they proposed the use of a parallel filament geometry for maximum film uniformity. Upon further investigation, relationships between  $d_{f-s}$  distance, minimum filament length as well as the number of parallel filaments used with the separation between them for the optimum thickness uniformity was obtained from the model. A notable observation from their work was that thickness uniformity increased with an increase in chamber pressure.

Swain and Dusane[10] investigated the effect of filament temperature on HWCVD deposited hydrogenated amorphous silicon (a-Si:H). In their work, the filament temperature was determined to be a critical parameter in a deposition of this kind. They, experimentally, demonstrated the changes in deposition rate, Carbon content, optical properties and Infra-red absorption of Si-C and C-H<sub>n</sub> for a-SiC:H alloy films deposited using filament temperature ranging from 1650°C to 2100°C.

Sharifi and Achenie[11] used a 2-D CFD model to investigate the effects of substrate geometry on the deposition rate in the CVD process. By solving the Momentum balance coupled with heat and mass conservation equations using the *COMSOL MULTIPHYSICS* software, their results indicated that certain substrate geometries change the flow patterns inside the reactor, creates recirculation, and increases the deposition rate. They deduced that changing the shape of the substrate creates a special flow pattern that could significantly affect the boundary layer thickness and mass transfer around the substrate.

Pflug et al.[12] modelled the gas flow and deposition profile in the HWCVD process. They utilized particle based simulation methods focused on solving the Boltzmann equations for pressures ranging from 0.1 Pa to 30 Pa. This selection was made with the intent to improve the accuracy of their results, compared to the CFD modelling results, near small gaps or openings where the characteristic geometry dimensions approach the order of the mean free path of the gas

molecules. They later developed a Direct Simulation Monte Carlo (DSMC) model, extended by a reactive wall chemistry model, to demonstrate the feasibility of three-dimensional simulations of the HWCVD process on realistic geometries. From their model, they were able to determine the absolute deposition rate and film thickness profiles. They thus concluded, since the results from the model and the experimental data were in agreement, that the model could be used to optimize the design of the HWCVD reactor with respect to gas conservation efficiencies and film thickness uniformity.

Song et al.[13] conducted CFD modelling research on the temperature parameter for the formation of diamond using the HWCVD process. In their work, they attempted to map the temperature distribution inside the system as a function of power applied to the filaments at constant pressure. Using the FLUENT CFD software, they were able to predict the filament temperature to be between 2512-2802 K for the applied power of 12, 14, 16 and 18 kW. Based on their simulation results, they conducted experiments in which they found the temperature to be in good agreement with the results from the model. The temperature was measured using a two-colour pyrometer.

Olivas-Martinez et al.[14] developed a two-dimensional CFD model of a HWCVD reactor. Their model solved the overall continuity, momentum, energy and specie continuity equations inside the reactor. The model incorporated the catalytic production of hydrogen radicals at the filament which were coupled with the expressions representing the recombination of the hydrogen radicals at surfaces and the growth rate of the diamond films on the substrate which were used in conjunction with the gas-phase transport equations. The results from their model showed that increasing the number of filaments used increased the concentration of  $\text{CH}_3$  in the reaction chamber as well as increases the growth rate of the diamond films. They also found that the shape and dimension of the reaction chamber, filaments and substrate significantly affected the model predictions.

Wahl et al.[15] experimentally modelled the gas temperature of  $\text{CH}_3$  in a HWCVD reactor to determine a safe  $d_{f-s}$  distance for the production of diamond films. In the study, they used the gas temperature and concentration at various  $d_{f-s}$  distances to the filament to determine the safe deposition distance. The gas concentration and filament temperature were kept constant throughout the experimental process whilst varying the  $d_{f-s}$  distance.

The effects of substrate and filament temperatures on the structural and electrical properties of SiC films grown by the HWCVD process was studied by Dasgupta et al.[16]. Their work showed that at constant filament temperature, the electrical conductivity of SiC films increased with an increase in substrate temperature. The increase in filament temperatures leads to unfavourable variations in the structure and conductivity of the material.

A combined experimental and computational study was conducted by Strengers et al.[17] on microcrystalline silicon solar cells prepared using the HWCVD technique. In their work, they combined experimental characterization with computer simulations. From the fitting of the solar cell curves, they found that the material has a density of dangling bonds and drift mobility that is comparable to that of amorphous silicon whereas the mobility band gap is closer to that of crystalline silicon. These fittings were done under the assumption of homogeneous electrical parameters in the intrinsic layers.

## 1.2 A brief history of Computational Fluid Dynamics

Computational Fluid Dynamics (CFD) is the branch of fluid dynamics providing a cost-effective means of simulating real flows by numerical solutions of the governing equations. The governing equations for Newtonian fluid dynamics, namely the Navier-Stokes equations, have been known for over 150 years[18]. CFD provides a qualitative and quantitative prediction of fluid flows by means of mathematical modelling (partial differential equations), numerical methods (discretization and solution techniques) and software tools (solvers, pre- and post-processing utilities). The history of CFD started in the early 1970's[19] when it widely became used to simulate fluid flows. The development of CFD was triggered by the availability of increasingly more powerful mainframe computers and advances in CFD are still closely linked to the evolution of computer technologies. The first application of the CFD method was in the simulation of transonic flows which later lead to the solution of the first two- and three-dimensional Euler equations[20].

In the mid 1980's, the focus shifted to the more demanding simulations of viscous flows governed by the Navier-Stokes equations[19]. These flow governing equations are particularly complicated to the extent where analytical solutions cannot be obtained for most practical applications. This lead to the development of reduced forms of these equations, like the Reynolds-averaged Navier-Stokes (RANS) equations, which are still an active field of study[18, 19]. Fast-forward to the present time, the solving of the RANS equations has become the bedrock upon which modern CFD software, both commercial and open-source, is built.

Numerous CFD codes exist today that range from commercially available packages, open-source codes or even problem specific codes. The use or choice of which depends greatly on the user's computational coding capabilities and what it is that they are modelling. There has been some debate as to whether commercial codes perform better than some of the open-source codes which are the two most commonly selected options when conducting CFD investigations. In work done by Rahman et al.[21], they attempted to compare the results and performance of what, at the time, was considered to be the two flagship packages from each category; FLUENT, belonging to the commercial codes class, and OpenFOAM, which is the open-source option. In

the work they found that, while OpenFOAM usage may require more computational ability on the part of the user, they gain the ability to modify and expand the codes without any of the restrictions that they would encounter when trying to do the same in the commercial/paid FLUENT package. They also concluded that the performance of OpenFOAM solvers was on par, if not better than, many other commercial codes of similar capability. This innate ability to modify codes to make the simulation process more problem specific that comes from the use of open-source CFD packages is the reason OpenFOAM was selected for the work done in this study. The methodology of OpenFOAM is expanded further in Chapter 3 of this thesis.

### 1.3 Objectives

Since there is no easy relation between process parameters and the resulting deposition profile, a substantial experimental effort is required to optimize the deposition process for a given film specification and the desired film uniformity. In order to obtain a better understanding of the underlying mechanisms and to enable an efficient way of process optimization, simulation methods come into play. Hence, the main objective of this thesis is to create a working model, using CFD modelling along with the open-source CFD software OpenFOAM, that can accurately predict the deposition conditions inside a HWCVD reactor. This is done by first computationally reproducing the HWCVD reactor of the CADAR deposition system under investigation, using Computer-aided design (CAD) software, which can later be used to study the deposition conditions inside the reactor vessel for different deposition parameters. The ultimate goal of this study is to provide the experimentalist with a predictive model of the HWCVD reactor that can be used to determine the optimum deposition parameters for their film requirements without having to conduct multiple depositions to achieve this end.

A secondary objective of this study is to compare the deposition condition for two of the most commonly used wire configurations namely: the straight or parallel wire configuration and the coiled wire configuration to test their performance in the HWCVD setup of the CADAR system. This will significantly add to the bank of insight into the HWCVD deposition parameters the model can provide.

A summary of the objectives:

- Create a CAD model of the HWCVD reactor under investigation.
- Create a CFD model of the problem in OpenFOAM 5.1
  - Create and evaluate the mesh.
  - Select the appropriate boundary conditions.
  - Select the appropriate Solver.

- Run simulations of HWCVD using test deposition parameters.
- Analyse the simulation results to identify usable deposition parameter data.
- Validate the obtained results where possible.

## 1.4 Thesis Outline

This thesis is divided into 6 chapters. Chapter 2 contains a comprehensive overview of the theory and methodology of CFD used in this investigation. This includes the theory of the governing equations for fluid flow and heat transfer with a description of all the heat transfer models available within the OpenFoam libraries.

Chapter 3 gives an overview of CFD as well as an overview of OpenFoam software. This chapter also includes a comprehensive description of the solvers used in this investigation, the numerical discretization schemes, the solution and algorithm control parameters and turbulence modelling theory.

The modelling approach is described in chapter 4. This includes information on how the mesh was obtained, boundary conditions as well as the modelling of transport properties and thermodynamics. Chapter 5 contains the results and discussion while chapter 6 provides the conclusions and prospects for future work.

This thesis also contains 3 appendices. Appendix A. contains the mesh quality output for the two wire configurations, also known as the checkMesh output. The OpenFOAM dictionaries for each solver are located in Appendix B. and Appendix C. respectively.

# Chapter 2

## THEORY

### 2.1 Introduction

This section intends to highlight the key aspects of the computational fluid dynamics, CFD, methodology as well as provide a mathematical overview of the equations that govern the flow of a fluid of a Newtonian nature.

### 2.2 Governing Equations

The pillar of CFD is the fundamental equations that govern fluid dynamics - the continuity, momentum and energy equations. They are the three mathematical statements of fundamental physics principles, namely, mass conservation,  $\mathbf{F} = m\mathbf{a}$  (Newton's second law) and energy conservation. Together these form the bases upon which fluid dynamics is built. To obtain the basic equations of fluid motion, the following approach is always followed[22]: Choose the appropriate fundamental physics principles from the laws of physics, apply these physical principles to a suitable model of the flow and lastly extract the mathematical equations which embody such physical principles. The sections that follow will give a brief overview of these governing equations[3].

#### 2.2.1 Conservation of Mass

The conservation of mass equation, also known as the continuity equation, is simply a mass balance of an arbitrary volume element and is derived by considering the mass flow in and out of a control volume. The rate of change of mass in the control volume must be equal to the rate of mass flowing in, minus the rate of mass flowing out.



$$\frac{\partial \rho}{\partial t} + \frac{\partial(\rho u)}{\partial x} + \frac{\partial(\rho v)}{\partial y} + \frac{\partial(\rho w)}{\partial z} = 0 \quad (2.1)$$

Equation (2.1) can be written in vector notation:

$$\frac{\partial \rho}{\partial t} = -\nabla \cdot \rho \mathbf{u} \quad (2.2)$$

where  $\rho$  is the density and  $\mathbf{u}$  is the velocity magnitude. If the fluid is incompressible,  $\rho =$  constant, independent of space and time such that  $\frac{\partial \rho}{\partial t} = 0$ , Equation (2.2) reduces to:

$$\nabla \cdot \mathbf{u} = 0 \quad (2.3)$$

### 2.2.2 Conservation of Momentum and Navier-Stokes Equations

Based on Newton's second law, we ascertain that the rate of change of momentum of a fluid particle equals the sum of the forces acting on the particle. This leads to the momentum equations in the x-, y-, and z-direction respectively [3]:

$$\rho \frac{Du}{Dt} = \frac{\partial(-p + \tau_{xx})}{\partial x} + \frac{\partial \tau_{yx}}{\partial y} + \frac{\partial \tau_{zx}}{\partial z} + S_{Mx} \quad (2.4)$$

$$\rho \frac{Dv}{Dt} = \frac{\partial \tau_{xy}}{\partial x} + \frac{\partial(-p + \tau_{yy})}{\partial y} + \frac{\partial \tau_{zy}}{\partial z} + S_{My} \quad (2.5)$$

$$\rho \frac{Dw}{Dt} = \frac{\partial \tau_{xz}}{\partial x} + \frac{\partial \tau_{yz}}{\partial y} + \frac{\partial(-p + \tau_{zz})}{\partial z} + S_{Mz} \quad (2.6)$$

where  $\tau_{xx}, \dots, \tau_{zz}$  are the viscous stress components and  $S_M$  are the body forces. To obtain a Newtonian fluid, we assume that the rate of deformation in the fluid is directly proportional to the shear stress. The directional momentum equation, Equations (2.4) - (2.6), thus reduces to the compressible Navier-Stokes equations:

$$\rho \frac{Du}{Dt} = -\frac{\partial p}{\partial x} + \nabla \cdot (\mu \nabla u) \quad (2.7)$$

$$\rho \frac{Dv}{Dt} = -\frac{\partial p}{\partial y} + \nabla \cdot (\mu \nabla v) \quad (2.8)$$

$$\rho \frac{Dw}{Dt} = -\frac{\partial p}{\partial z} + \nabla \cdot (\mu \nabla w) \quad (2.9)$$

where  $\mu$  is the dynamic viscosity. In Equations (2.7) - (2.9), the body forces are omitted but can be added easily if required.

### 2.2.3 Conservation of Energy

The first law of thermodynamics gives rise to the energy equation which states that the rate of change of energy of a fluid particle is equal to the net rate of heat added to the particle, plus the net rate of work done on the particle. Extracting the changes due to the kinetic energy and introducing the assumptions for a Newtonian fluid, we obtain the equation for internal energy:

$$\rho \frac{Di}{Dt} = -p \nabla \cdot (\mathbf{u}) + \nabla \cdot (k \nabla T) + \Phi_d + S_i \quad (2.10)$$

where  $p$  is pressure,  $i$  is the internal energy,  $k$  is the turbulent kinetic energy and  $S_i$  is the source term for internal energy. The term  $\Phi_d$  describes the effect of viscous stresses and is known as the dissipation function.

### 2.2.4 Equation of State

The equation of state provides the relationship between the various state variables of a fluid. These include; temperature, pressure and volume. The ideal gas law states:

$$pV = nRT \quad (2.11)$$

where  $p$  is pressure,  $V$  is volume,  $n$  is the number of moles,  $R$  is the universal gas constant and  $T$  is temperature[23].

The ideal gas law does not model the real gas effects. These can be taken into account by using the Peng-Robinson equation of state proposed by Peng et al[24].

$$p = \frac{RT}{V_m - b} - \frac{a\alpha}{V_m^2 - 2bV_m - b^2} \quad (2.12)$$

$$a = \frac{0.457235R^2T_c^2}{p_c} \quad (2.13)$$

$$b = \frac{0.077796RT_c}{p_c} \quad (2.14)$$

$$\alpha = (1 + \kappa(1 - \sqrt{T_r}))^2 \quad (2.15)$$

$$\kappa = 0.37464 + 1.5422\omega_a - 0.26992\omega_a^2 \quad (2.16)$$

$$T_r = \frac{T}{T_c} \quad (2.17)$$

where  $V_m$  is the volume of 1 mole of gas,  $\omega_a$  is an acentric factor,  $p_c$  and  $T_c$  are the critical pressure and temperature.

## 2.3 Heat Transfer

Heat is defined as the energy transfer due to temperature gradients. In general, three modes of heat transfer exist: conduction, convection and radiation. Thermodynamics only recognizes two of these modes, i.e. conduction and radiation. Convection is the transport of energy by the bulk motion of a medium[25].

Heat transfer problems can be solved using either transient or steady state algorithms. The steady state solutions only vary with location, while the transient solutions also vary with time. Subsequently, this means steady state heat transfer problems are easier to solve since all the time derivatives are equal to zero. It is for this reason that steady state simulations are used in this work.

### 2.3.1 Conduction

Heat transfer due to conduction occurs in both solids and dormant fluids. In this mode, heat is transferred by diffusion i.e. thermal diffusion, and collisions between particles without any mass flow. Heat diffuses from a high- to a low-temperature region due to the temperature gradient between those regions. The heat transfer rate varies depending on the material, geometry and the temperature gradient. The relationship between the heat flow and the temperature gradient is given by Fourier's Law. A one-dimensional example is shown below:

$$\dot{Q}_{cond} = -k_c A \frac{dT}{dx} \quad (2.18)$$

where  $k_c$  is the thermal conductivity, which is a measure of a materials ability to transfer heat by conduction,  $A$  is the cross-sectional area,  $\frac{dT}{dx}$  is the temperature gradient while  $\dot{Q}_{cond}$  is the

heat flux. The negative sign denotes that heat flows in the direction opposite to the temperature gradient. Fourier's law is thus the defining equation for thermal conductivity.

Thermal diffusivity,  $\alpha$ , is another important property of the material. It is defined as the thermal conductivity divided by density,  $\rho$ , multiplied by the specific heat capacity,  $c_p$  [26].

$$\alpha = \frac{k_c}{\rho c_p} \quad (2.19)$$

The quantity  $\rho c_p$  has been referred to as the volumetric heat capacity thus, the thermal conductivity is a measure of a material's ability to conduct heat relative to its volumetric heat capacity. The heat distribution due to conduction is described by a parabolic partial differential equation. For an isotropic material without internal heat generation, the one-dimensional heat equation becomes:

$$\frac{\partial T}{\partial t} = \frac{k_c}{\rho c_p} \left( \frac{\partial^2 T}{\partial x^2} \right) = \alpha \left( \frac{\partial^2 T}{\partial x^2} \right) \quad (2.20)$$

### 2.3.2 Convection

Heat is transferred through a fluid by convection in the presence of bulk fluid motion[25]. Convection is generally classified by two main types, forced and natural convection. Natural convection occurs when the flow is initiated by the buoyancy effect. If the fluid motion is caused by external forces, such as a pump or an extractor fan, we have forced convection.

The heat transfer rate in a fluid due to convection is higher than the rate due to conduction. This is due to the fluid motion bringing warmer and cooler portions of the fluid into contact which increases the heat transfer rate.

The rate of heat transfer depends on several fluid properties such as; dynamic viscosity  $\mu$ , thermal conductivity  $k_c$ , density  $\rho$ , specific heat capacity  $c_p$  and fluid velocity. Other variables to consider are; geometry, surface roughness and whether the flow is turbulent or laminar. Newton's law of cooling is used to express the rate of heat transfer due to convection, which shows that the rate of heat transfer due to convection is proportional to the temperature difference[27]:

$$\dot{Q}_{conv} = h_c A_s (T_s - T_\infty) \quad (2.21)$$

where  $h_c$  is the convective heat transfer coefficient,  $A_s$  is the surface area,  $T_s$  is the surface temperature and  $T_\infty$  is the temperature in the fluid sufficiently far from the surface. The expression looks relatively simple, however, the convective heat transfer coefficient is difficult to determine since it depends on many of the above-mentioned fluid properties.

### 2.3.3 Radiation

All bodies emit energy in the form of electromagnetic radiation across the entire electromagnetic spectrum, depending on the temperature of the body. Radiation differs from conduction and convection in that it may be transferred over larger distances and it is not medium dependent. For most heat transfer problems, the heat transfer by radiation can be calculated using Newton's law of cooling, as long as a radiation heat transfer coefficient is defined:

$$\dot{Q}_{rad} = h_r A_s (T_s - T_\infty) \quad (2.22)$$

where  $h_r$  is the radiation heat transfer coefficient and all the other symbols are as defined.  $h_r$  is a function of geometry, the emissivity of the surface and the temperature of the surface. In most cases, heat transfer by convection and radiation are active at the same time. If the temperature differences for the two modes are the same, the heat transfer coefficients for convection and radiation may be added to obtain a total heat transfer coefficient  $h_{tot}$ .

$$h_c + h_r = h_{tot} \quad (2.23)$$

There exists three modes of radiation in OpenFOAM: the NoRadiation model, the Finite Volume Discrete Ordinates model and the P1 model. These three modes are discussed in the sections that follow based on work reported by Chalmers[28].

In the abstract class for radiation models in OpenFOAM, the three most important member functions are  $Ru()$ ,  $Rp()$  and the one that generates a source term of the enthalpy equation -  $Sh()$ . These member functions are used in different radiation models and are done to redistribute source terms to make the main matrix more diagonally dominant which, subsequently, simplifies the solution process. The equation to be solved can thus be written as:

$$Sh() = Ru() - 4Rp() \frac{T^3 h}{c_p} - Rp() T^4 + 4Rp() \frac{T^3 h}{c_p} \quad (2.24)$$

$$Sh() = Ru() - Rp() T^4 \quad (2.25)$$

where  $h$  is the enthalpy.

### 2.3.3.1 NoRadiation Model

For testing purposes, and a quick way of disabling radiative heat transfer in the calculations, a *noRadiation* model is used. This sets the member functions  $Ru()$  and  $Rp()$  equal to zero so that the additional term for the enthalpy equation  $Sh()$  equals zero.

### 2.3.3.2 Finite Volume Discrete Ordinates Model (fvDOM)

fvDOM model is the second radiation model available in OpenFOAM. In it, the radiative transfer equation is solved for a discrete number of finite solid angles,  $\sigma_s$ , which are the angles in between two grid point in the mesh each associated with a vector direction.

The advantages of this model are:

- This is a conservative method that leads to heat balance for coarse discretization, with an accuracy that can be increased by using finer discretization.
- This is the most comprehensive radiation model as it accounts for scattering, semi-transparent media, specular surfaces and wavelength-dependent transmission using the banded-gray option.

The major limitation of the model:

- Solving a problem with a large number of ordinates is CPU-intensive.

### 2.3.3.3 P1 model

The most commonly implemented model for radiative heat transfer in OpenFOAM is the P1 model. The main assumption of this model is that the directional dependence in the radiative transfer equation is integrated out, resulting in a diffusion equation for incident radiation.

Advantages:

- Radiative transfer equation is easy to solve with little CPU demand.
- Includes effects for scattering, effects of particles, droplets and soot.
- Works well for applications where the optical thickness of the medium is large,  $\tau = a*L > 3$  where  $L$  is the distance between the objects in the given geometry and  $a$  is the absorbance.

Limitations:

- Assumes all surfaces are diffuse.
- May result in a loss of accuracy, depending on the complexity of the geometry or if the optical thickness of the medium is small.
- Tends to overestimate radiative fluxes from localized heat sources.



## Chapter 3

# Computational Fluid Dynamics

### 3.1 Introduction to Computational Fluid Dynamics (CFD)

CFD is used to analyze fluid flow, heat transfer, chemical reactions and other phenomena associated with fluid dynamics. CFD codes are based on numerical algorithms that can be used to solve the implicit differential equations of fluid flow. All CFD codes consist of three main components: a pre-processor, a solver and a post-processor.

The pre-processor is where the input parameters of the flow are defined. Here, the computational domain is specified, mesh generated and the boundary conditions, as well as the physical properties of the problem, are defined. Essentially, in the pre-processor phase, the user input data is transformed into a form that the solver can use.

Once the problem is adequately defined in the pre-processor, the solver uses the information to compute a solution. There are three well known numerical solution techniques that are used in the CFD analysis: finite element, finite difference and spectral methods. The most commonly used, in OpenFOAM and commercial CFD codes, is the finite volume method (FVM). The FVM algorithm consists of the following three main steps:

- Integration of the governing equations of fluid flow over all the control volumes inside the computational domain.
- Discretising the resulting integral equations into a system of algebraic equations.
- Solving the algebraic equations by iterative methods.

The integration of the governing equations provides an exact expression for the conservation of relevant properties for each cell in the domain. This provides a clear connection between



the conservation laws of physics and the numerical solution algorithm. Numerous discretization schemes are available and should be carefully selected based on the characteristics of the problem being dealt with.

The results are visualized in the post-processor. This is done by way of vector plots, contour plots, particle tracking or plots of variables over time or space.

When solving fluid flow problems, complex physics is involved. As such, it is important that the user has a complete understanding of the underlying physics involved in their problem. This is due to the fact that the results may be visually good, but might be physically incorrect. It is therefore of utmost importance that the results be compared to experimental data, if available, or to results from similar problems.

## 3.2 OpenFOAM

Open Source Field Operations And Manipulation (OpenFOAM) is an open source C++ library. It is primarily designed to create executables, known as applications. Users have access to a large repository of precompiled applications with the ability to create their own, or modify existing ones, with the users C++ knowledge being the only restriction. The applications are divided into two categories: utilities and solvers. Solvers are designed to solve the problems of a specific type in computational continuum mechanics, while utilities are used for pre-and post-processing tasks which include data manipulation and algebraic calculations.

OpenFOAM belongs to the family of Open source software that has, as of late, garnered a fair bit of attention from academics and the research field as a whole. Open source software provides unrestricted access to the codes/algorithms giving limitless customization possibilities to the user.

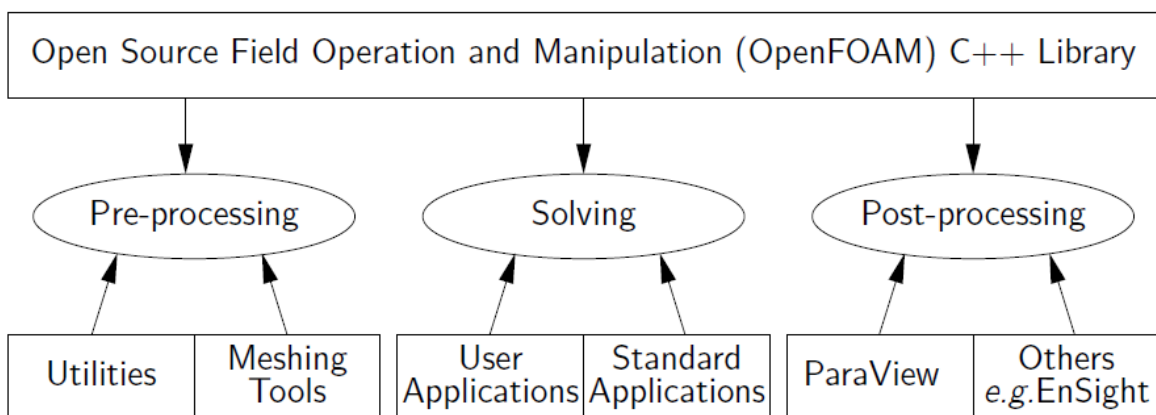


FIGURE 3.1: Overview of OpenFOAM structure [1]

### 3.2.1 Case Structure

In OpenFOAM, cases are built up in one main folder, called the directory. The directory contains three basic folders, namely: time (also named 0), constant and system.

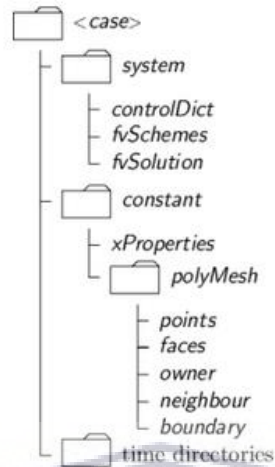


FIGURE 3.2: General OpenFOAM case structure [2]

In the constant directory, a subdirectory called *polyMesh* contains the description of the case mesh. Depending on the type of problem being dealt with, this directory also contains files, called *transportProperties* and *thermophysicalProperties*, that where the physical properties of the fluid flow is defined.

The parameters associated with the solution procedure are defined in files inside the system folder. The most important files are: *controlDict* in which the length of the time step, write intervals and write precision are defined, *fvSchemes* in which the discretization schemes for the different variables are selected and *fvSolution* which controls the solvers, tolerances and other algorithm control functions.

The 0/time directory contains the boundary conditions for the different fields. For each new time step, specified by the write interval in the *controlDict* file, the solver computes, a new time folder is created containing all the fields for that particular time step.

## 3.3 Solvers

### 3.3.1 buoyantSimpleFoam

buoyantSimpleFoam is a steady state solver for buoyant, turbulent flow of compressible fluids, including radiation, for ventilation and heat transfer. In this section, an overview of the "buoyant" class solver in OpenFOAM's treatment of the governing equations is given based on [29].

The continuity equation takes the familiar form:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0 \quad (3.1)$$

Conservation of momentum:

$$\frac{\partial(\rho \mathbf{u})}{\partial t} + \nabla \cdot (\rho \mathbf{u} \mathbf{u}) = -\nabla p + \rho \mathbf{g} + \nabla \cdot (2\mu_{eff} D(\mathbf{u})) - \nabla \left( \frac{2}{3} \mu_{eff} (\nabla \cdot \mathbf{u}) \right) \quad (3.2)$$

where  $\mathbf{u}$  is the velocity magnitude,  $\rho$  is the density,  $p$  is the static pressure and  $\mathbf{g}$  is the acceleration due to gravity. The effective viscosity,  $\mu_{eff}$ , is the sum of the molecular and turbulent viscosity, while  $D(\mathbf{u})$  is the deformation tensor and is defined as:

$$D(\mathbf{u}) = \frac{1}{2} (\nabla \mathbf{u} + (\nabla \mathbf{u})^T) \quad (3.3)$$

In OpenFOAM, the pressure gradient and gravity force terms are implemented as follows:

$$-\nabla p + \rho \mathbf{u} = -\nabla p_{rgh} - (\mathbf{g} \cdot \mathbf{r}) \nabla \rho \quad (3.4)$$

where  $p_{rgh} = p - \rho \mathbf{g} \cdot \mathbf{r}$  and  $\mathbf{r}$  is the position vector.

For the conservation of energy, two variables are selectable for the energy solution variable, namely: internal energy,  $\mathbf{e}$  or enthalpy,  $\mathbf{h}$ . This selection is made in accordance with the "energy" keyword in the *thermoPhysicalProperties*. If enthalpy is selected (selected in OpenFOAM using the keyword *sensibleEnthalpy*), the energy equation takes the form:

$$\frac{\partial(\rho \mathbf{h})}{\partial t} + \nabla \cdot (\rho \mathbf{u} \mathbf{h}) + \frac{\partial(\rho K)}{\partial t} + \nabla \cdot (\rho \mathbf{u} K) - \frac{\partial p}{\partial t} = \nabla \cdot (\alpha_{eff} \nabla \mathbf{h}) + \rho \mathbf{u} \cdot \mathbf{g} \quad (3.5)$$

whereas when internal energy is selected (selected in OpenFOAM using the keyword *sensibleInternalEnergy*), the equations has the form:

$$\frac{\partial(\rho \mathbf{e})}{\partial t} + \nabla \cdot (\rho \mathbf{u} \mathbf{e}) + \frac{\partial(\rho K)}{\partial t} + \nabla \cdot (\rho \mathbf{u} K) + \nabla \cdot (p \mathbf{u}) = \nabla \cdot (\alpha_{eff} \nabla \mathbf{e}) + \rho \mathbf{u} \cdot \mathbf{g} \quad (3.6)$$

where  $K \equiv \frac{|\mathbf{u}|^2}{2}$  is the kinetic energy per unit mass and  $\mathbf{h}$  is the sum of the internal energy per unit mass,  $\mathbf{e}$  and the kinematic pressure i.e.  $\mathbf{h} \equiv \mathbf{e} + \frac{p}{\rho}$ . We obtain the following relations from these definitions:

$$\frac{\partial(\rho \mathbf{e})}{\partial t} = \frac{\partial(\rho \mathbf{h})}{\partial t} - \frac{\partial p}{\partial t} \quad (3.7)$$

$$\nabla \cdot (\rho \mathbf{u} \mathbf{e}) = \nabla \cdot (\rho \mathbf{u} \mathbf{h}) - \nabla \cdot (p \mathbf{u}) \quad (3.8)$$

The effective thermal diffusivity  $\alpha_{eff}$  is the sum of the laminar and turbulent thermal diffusivities:

$$\alpha_{eff} = \frac{\rho \nu_t}{Pr_t} + \frac{k_c}{c_p} \quad (3.9)$$

where  $k_c$  is thermal conductivity,  $c_p$  is the specific heat at constant pressure,  $\nu_t$  is the kinematic viscosity and  $Pr_t$  is the turbulent Prandtl number. As specified earlier, `buoyantSimpleFoam` is a steady state solver which means that all time derivatives are omitted from the solution algorithm but it maintains the structure of the equations above.

### 3.3.2 rhoSimpleFoam

`rhoSimpleFoam` is a steady state solver for the turbulent flow of compressible fluids, based on the semi-discrete, non-staggered central schemes, developed by Greenshield et al. [30]. An overview of the "rho" class solver's solution algorithm is given below which is based on [31, 32].

In `rhoSimpleFoam`, each governing equation is solved separately. Firstly, density is calculated, from the continuity equation, using the velocity values from the preceding time step.

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0 \quad (3.10)$$

Secondly, the momentum equation is solved. In order to avoid an explicit solution procedure, which creates time step limitations, the momentum equation is solved in two steps. It starts off by calculating the inviscid momentum density,  $\hat{\mathbf{u}}$ .

$$\left( \frac{\partial \hat{\mathbf{u}}}{\partial t} \right)_I + \nabla \cdot (\mathbf{u} \hat{\mathbf{u}}) + \nabla \cdot p = 0 \quad (3.11)$$

Here,  $\left( \frac{\partial \hat{\mathbf{u}}}{\partial t} \right)_I$  is the time derivative due to inviscid fluxes only. Since  $\hat{\mathbf{u}} = \rho \mathbf{u}$ , a new value for the velocity can be calculated using the values calculated for  $\hat{\mathbf{u}}$  and  $\rho$ . The viscous forces are accounted for by solving a diffusion correction equation for the velocity.

$$\left( \frac{\partial (\rho \mathbf{u})}{\partial t} \right)_V - \nabla \cdot (\mu \nabla \mathbf{u}) - \nabla \cdot \mathbf{T}_{visc} = 0 \quad (3.12)$$

Here,  $\left(\frac{\partial(\rho\mathbf{u})}{\partial t}\right)_V$  is the time derivative related to diffusion and  $\mathbf{T}_{visc}$  is the viscous stress tensor. The energy equation is solved much in the same way. The process is started by calculating the total energy density,  $\hat{E}$ , neglecting the diffusive heat flux. Similar to the inviscid momentum density, the energy density is  $\hat{E} = \rho E$ .

$$\left(\frac{\partial\hat{E}}{\partial t}\right)_I + \nabla \cdot (\mathbf{u}(\hat{E} + p)) + \nabla \cdot (\mathbf{T} \cdot \mathbf{u}) = 0 \quad (3.13)$$

$$T = \frac{1}{c_v} \left( \frac{\hat{E}}{\rho} - \frac{|\mathbf{u}|^2}{2} \right) \quad (3.14)$$

In equation (3.13),  $\left(\frac{\partial\hat{E}}{\partial t}\right)_I$  is the time derivative solely due to inviscid fluxes. The temperature is calculated using equation (3.14) using the values previously calculated for  $\hat{E}$ ,  $\mathbf{u}$  and  $\rho$ .  $c_v$  is the specific heat capacity at constant volume. Lastly, a diffusion correction equation is solved for the temperature to include the diffusive heat flux.

$$\left(\frac{\partial(\rho c_v T)}{\partial t}\right)_V - \nabla \cdot (k_c \nabla T) = 0 \quad (3.15)$$

The thermal conductivity,  $k_c$ , and the dynamic viscosity,  $\mu$ , are functions of temperature and are updated during each iteration. Both  $k_c$  and  $\mu$  remain constant until the next iteration. The pressure is updated after each iteration, using the ideal gas law,  $p = \rho RT$ . With rhoSimpleFoam being a steady state solver, the time derivatives are once again omitted from the solution algorithm above.

### 3.4 Numerical Schemes

In the Finite Volume Method (FVM), the governing equations are integrated over the control volume. The resulting integral equations need to be discretized into algebraic equations. Different numerical discretization schemes are available in OpenFOAM and these are found in the *fvSchemes* file in the system directory. The terms that need to be specified by numerical schemes are divided into categories listed in Table 3.1 below [33].

The numerical schemes used for the solvers described in section 3.3 and in this thesis are shown in Table 3.2. The selection of the numerical schemes is based on the *fvSchemes* files from the rhoSimpleFoam and buoyantSimpleFoam tutorials that come with OpenFOAM.

TABLE 3.1: Group of terms.

Groups	Description
interpolationSchemes	Point-to-point interpolations of values
snGradeSchemes	Component of gradient normal to cell face
gradSchemes	Gradient $\nabla$ terms
divSchemes	Divergence $\nabla \cdot$ terms
laplacianSchemes	Laplacian $\nabla^2$ terms
timeSchemes	First and Second time derivatives
fluxRequired	Fields that require the generation of a flux

TABLE 3.2: Selected numerical discretization schemes in *fvSchemes* file

Groups	buoyantSimpleFoam	rhoSimpleFoam
ddtSchemes	steadyState	steadyState
gradSchemes	Gauss linear	Gauss linear
divSchemes	div(phi, u): bounded Gauss upwind div(phi, K): bounded Gauss upwind div(phi, h): bounded Gauss upwind div(phi, k): bounded Gauss upwind div(phi, epsilon): bounded Gauss upwind div(phi, Ekp): N/A div(phiid, p): N/A	div(phi, u): bounded Gauss upwind div(phi, K): N/A div(phi, h): N/A div(phi, k): bounded Gauss upwind div(phi, epsilon): bounded Gauss upwind div(phi, Ekp): bounded Gauss upwind div(phiid, p): Gauss upwind
laplacianSchemes	Gauss linear corrected	Gauss linear corrected
interpolationSchemes	linear	linear
snGradSchemes	corrected	corrected

Detailed descriptions of the discretization schemes will not be discussed in this work, however, it is important to note that the selection of different schemes can affect both the accuracy and efficiency of the simulation. An example of this would be, applying the hybrid difference scheme or the power law scheme to this work would require an extremely fine mesh, and subsequently more storage space and CPU-power, to produce more accurate shock wave predictions. The discontinuous changes in velocity, density and pressure, caused by shock waves, would introduce unphysical oscillations in the solutions. Hence, the first order upwind scheme is selected to capture the sharp shock waves on our fine mesh without introducing any unwanted oscillations in the solutions. This reduces computational time and helps to improve the correctness of the solutions. For further insight on how to select the appropriate numerical schemes for your problem, the reader is referred to lectures given by Guerrero[34].

## 3.5 Solution and Algorithm Control

During the process of running numerical simulations, it is always important to verify that convergence of the solution is reached. A converged solution is a good indicator used to identify if the simulation is physically correct [35]. The convergence of a simulation can be determined by monitoring the residuals and continuity error. This is done “on-the-fly” by plotting the residuals for each time step, or iteration in the case of steady state simulations, while the simulation is running. While the convergence of the solution is a good indicator of physical correctness, it is important to always validate results with experimental data where possible.

### 3.5.1 Residuals and Continuity Error

Residuals are a measure of imbalance, or error, that occur in the solution. As such, the smaller the residuals become, the more accurate the solution will be. The residuals are calculated by substituting the current solution for a time step or iteration into the equations and taking the absolute value of the difference between the left and right hand side. To make the result independent of the scale of the problem, the residuals are normalized [36].

The continuity error is a measure of the error in the solutions mass imbalance. The sum of the magnitude of the flux imbalance for all cells is a good measure of the continuity error [37]. This is found in the solver’s output from OpenFOAM, called “sum local” continuity error, as is also calculated after each time step or iteration.

### 3.5.2 fvSolutions

A solver needs to be specified for each of the discretized equations. The solvers used are defined in the *fvSolutions* file in the system directory. This file contains two directories, the solver control and the solution algorithm control. The controls for the solver are specified first followed by the controls for the solution algorithm. Other important solution parameters are defined in this file as well namely: tolerance, relative tolerance and a preconditioner if needed.

The equations are solved by reducing the residuals through a series of iterations. An initial residual is obtained, using the current values of the field, after which the residual is re-calculated after each iteration. If the residual falls below the specified tolerance or relative tolerance value, the solver stops. The relative tolerance is the ratio of the current to the initial residuals.

Table 3.3 shows the solvers that were used in this work, including tolerances and relative tolerances.

TABLE 3.3: Solvers selected in *fuSolution*

Equation	rhoSimpleFoam		buoyantSimpleFoam	
$p$	solver	GAMG;	N/A	
	smoother	GaussSeidel;		
	tolerance	1e-08;		
	relTol	0.1;		
$p_{rgh}$	N/A		solver	PCG;
			preconditioner	DIC;
			tolerance	1e-06;
			relTol	0.01;
$U k \epsilon$	solver	GAMG;	solver	PBiCGStab;
	smoother	GaussSeidel;	preconditioner	DILU;
	tolerance	1e-08;	tolerance	1e-05;
	relTol	0.1;	relTol	0.1;
$e$	solver	GAMG;	N/A	
	smoother	GaussSeidel;		
	tolerance	1e-08;		
	relTol	0.1;		
$h$	N/A		solver	PBiCGStab;
			preconditioner	DILU;
			tolerance	1e-05;
			relTol	0.1;
$G$	N/A		$\$ p_{rgh}$ ;	
			tolerance	1e-05;
			relTol	0.1;

rhoSimpleFoam uses the Geometric agglomerated Algebraic Multi-Grid (GAMG) solvers with a Gauss Seidel smoother, which is the most reliable combination [34]. buoyantSimpleFoam uses two different solver and preconditioner combinations for its equations. For the  $p_{rgh}$  equation, it uses a Preconditioned Conjugate Gradient (PCG) solver with the Diagonal Incomplete-Cholesky (DIC) preconditioner. All the other equations use the Preconditioned bi-conjugate gradient (PBiCGStab) solver with a Simplified Diagonal-based Incomplete LU (DILU) preconditioner. The preconditioner multiplies both sides of the equation with a new matrix in an attempt to make the solution process easier [38].



### 3.6 Turbulence Modelling

The flow in our case is clearly turbulent. The turbulence arises from the large change in diameter from the inlet pipe to the reactor chamber. In turbulent flow, all the variables concerning flow vary in a random way. Figure 3.3 shows how the velocity in a turbulent flow varies with time at different points.

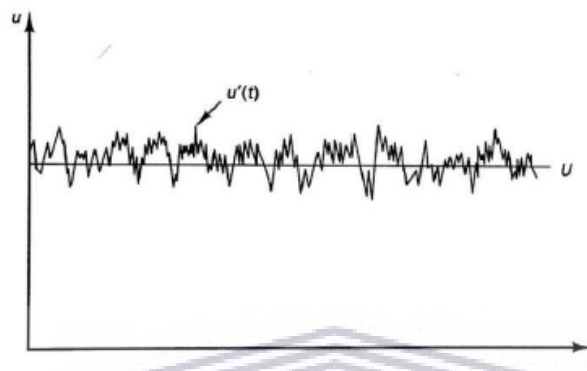


FIGURE 3.3: Turbulent flow [3]

Turbulent flow contains rotational flow structures, called turbulent eddies[4], with a wide range of length scales. The largest eddies are dominated by inertia forces and extract energy from the mean flow by a process called vortex stretching[39]. This process leads to eddies with smaller length and time scales. There are three basic approaches to turbulence modelling. Reynolds Averaged Navier-Stokes (RANS) models, Large Eddy Simulations (LES) and Direct Numerical Simulations (DNS). Figure 3.4 shows the accuracy of the different models.

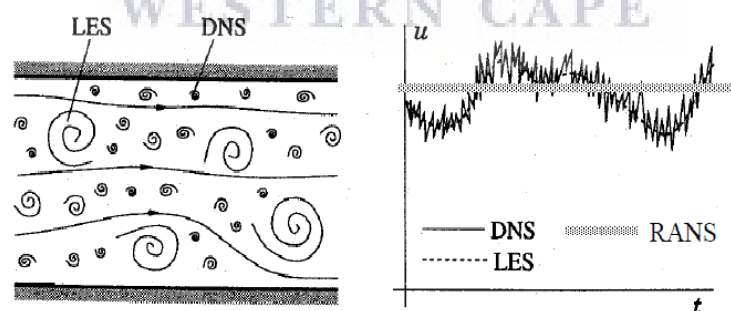


FIGURE 3.4: RANS, LES and DNS [4]

DNS resolves all the details of the flow and is the method that is the most computationally taxing. LES resolves only the largest eddies while the smaller ones are merely approximated. The RANS-models requires the least computational power and for this reason, is utilized in this thesis. The flow variables are decomposed into a steady mean value,  $\Phi$ , and a fluctuating component  $\varphi'(t)$ . This is known as the Reynolds decomposition.

The introduction of Reynolds decomposition into the governing equations results in the appearance of new terms. The new terms in the momentum equations are referred to as Reynolds stresses. An example of these new terms would be; in the scalar transport equations, the terms can represent the heat or mass flux due to turbulence. Governing equations for the new terms can be derived but this would introduce new unknowns into the equations. It is, therefore, necessary to find a relation between the known and the new terms. Introducing a turbulence model achieves this end.

A turbulence model is a set of equations that express the relation between the unknown Reynolds stresses and known flow quantities. Most RANS models are based on the concept of eddy viscosity, more commonly known as turbulent viscosity, proposed by Boussinesq in 1877 [40]. The Reynolds stress tensor is assumed to be proportional to the mean rate of deformation[40]:

$$\tau_{ij} = 2\mu_T S_{ij} - \frac{2}{3}\rho k\delta_{ij} \quad (3.16)$$

where  $\tau_{ij}$  is the Reynolds stress tensor,  $\mu_T$  is the eddy viscosity,  $S_{ij}$  is the mean strain rate tensor,  $k$  is the turbulent kinetic energy and  $\delta_{ij}$  is the Kronecker-delta.  $\mu_T$  is not a property of the flow and must be estimated for the turbulence model. From dimensional considerations, it can be shown that the eddy viscosity is proportional to the characteristic velocity of the turbulence,  $\mathbf{u}_T$ , and the turbulent characteristic length scale,  $l_T$ :

$$\mu_T = \rho \mathbf{u}_T l_T. \quad (3.17)$$

This led to the development of the two-equation turbulence models. These models use two extra equations to calculate  $\mathbf{u}_T$  and  $l_T$  which is then used to calculate  $\mu_T$ . The  $k - \epsilon$  model, used in this work, is an example of a two-equation turbulence model.

### 3.6.1 $k - \epsilon$ model

The  $k - \epsilon$  model for turbulence is the most commonly used to simulate average flow characteristics for turbulent flow conditions. As previously mentioned, this is a two-equation model which gives a general description of turbulence by means of two partial differential transport equations. These account for historical effects like convection and diffusion of turbulent energy. The two transported variables are turbulent kinetic energy  $k$ , which determines the energy in turbulence, and the turbulent dissipation  $\epsilon$ , which determines the rate of dissipation of the turbulent kinetic energy. The  $k - \epsilon$  model is shown to be applicable for free-shear flows, such as the ones with relatively small pressure gradients [41], but might not be suitable for problems involving large adverse pressure gradients [42]. Given the nature of the problem, the  $k - \epsilon$  model is deemed to be suitable for the work done here. The turbulent kinetic energy is given by:

$$k = \frac{3}{2}(\mathbf{u}\mathbf{I})^2 \quad (3.18)$$

where  $\mathbf{u}$  is the mean flow velocity and  $\mathbf{I}$  is the turbulent intensity. The turbulent intensity gives the level of turbulence and is defined as[31]

$$\mathbf{I} \equiv \frac{\mathbf{u}'}{\mathbf{u}} \quad (3.19)$$

where  $\mathbf{u}'$  is the root-mean-square of the turbulent velocity. The root-mean-square of the turbulent velocity fluctuation is given as:

$$\mathbf{u}' = \sqrt{\frac{1}{3}(u'^2 + v'^2 + w'^2)} = \sqrt{\frac{2}{3}k} \quad (3.20)$$

It follows that the mean velocity can be calculated using:

$$\mathbf{u} = \sqrt{u^2 + v^2 + w^2} \quad (3.21)$$

The turbulent intensity can be estimated using:

$$\mathbf{I} = 0.16Re_{d_h}^{-\frac{1}{8}} \quad (3.22)$$

where  $Re_{d_h}$  is the Reynolds number for a pipe of hydraulic diameter  $d_h$ . The turbulent dissipation rate is calculated using:

$$\epsilon = C_\mu^{\frac{3}{4}} \frac{k^{\frac{3}{2}}}{l} \quad (3.23)$$

where  $C_\mu$  is the turbulent model constant which usually takes the value 0.09 as part of the set of closure coefficients for the  $k - \epsilon$  model quoted by David Wilcox[42]. The turbulent length scale describes the size of the large energy containing eddies in a turbulent flow. For a fully developed flow, the turbulent length scale is given by:

$$l = 0.07d_h \quad (3.24)$$

Equations (3.18) - (3.24) are used in this investigation to determine the initial boundary condition values of the  $k - \epsilon$  model the specifics of which is given in chapter 4.

## Chapter 4

# Modelling Approach

In this chapter, the case setup for the simulations will be defined. This includes a description of the mesh, boundary conditions and phase properties used for the different cases.

### 4.1 Mesh

A computational geometry, that replicated the CADAR system, was created from measurements taken of the CADAR deposition system at the University of the Western Cape Department of Physics and Astronomy. The replicated geometry was subsequently then used to generate the computational domain or mesh. The CADAR is a multi-chamber ultra-high vacuum deposition system used for the growth and refinement of high quality thin films. The system comprises of two separate Plasma-Enhanced CVD reactor chambers, two HWCVD reactor chambers, a thermal evaporation chamber and a transfer chamber all separated by valves that allow each chamber to be kept at different pressures. For this study, the focus was solely on the HWCVD chamber of the system that is connected to one of the Plasma reaction chambers as shown in Figure 4.1. No valves are separating the two reactor chambers so they are considered to be a single vessel with two separate sources for deposition.

The system in Figure 4.1 is reproduced using the Computer Aided Design (CAD) software FreeCAD, based on the physical measurements taken, using an assortment of measuring instruments, of the systems internal and external geometry. The 3D mesh is then created from the CAD model using one of the built-in pre-processing tools of OpenFOAM called snappyHexMesh to ensure only hexahedral cells are used. This results in a more structured mesh which subsequently makes the solution process easier. During the process of creating the mesh, boundary conditions are added to the different components of the system namely, the filaments and its holder, the substrate holder or stage as well as the gas inlet and outlet. The 3D CAD for the

2 wire configurations are presented in Figures 4.2 and 4.3, followed by the generated mesh in Figures 4.4 to 4.7.

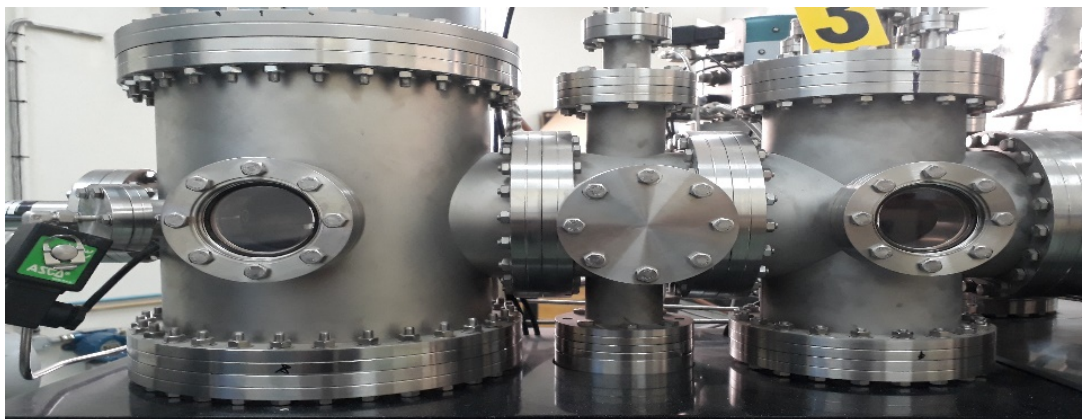


FIGURE 4.1: CADAR HWCVD reactor vessel

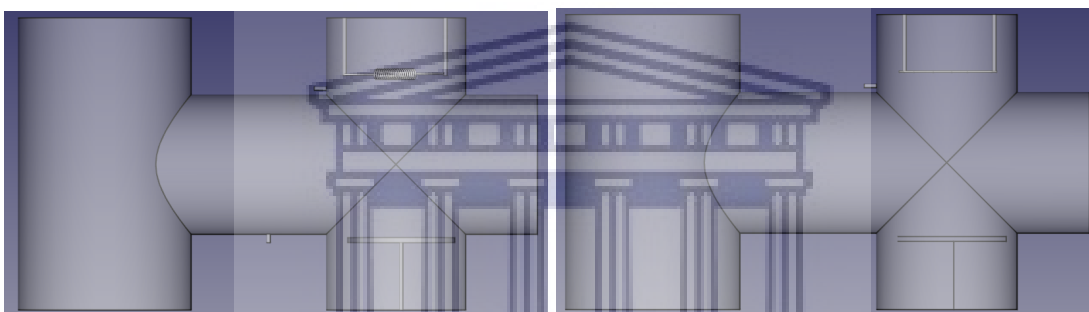


FIGURE 4.2: Side-view of the 3D CAD model for the coiled wire configuration (left) and straight wire configuration (right).

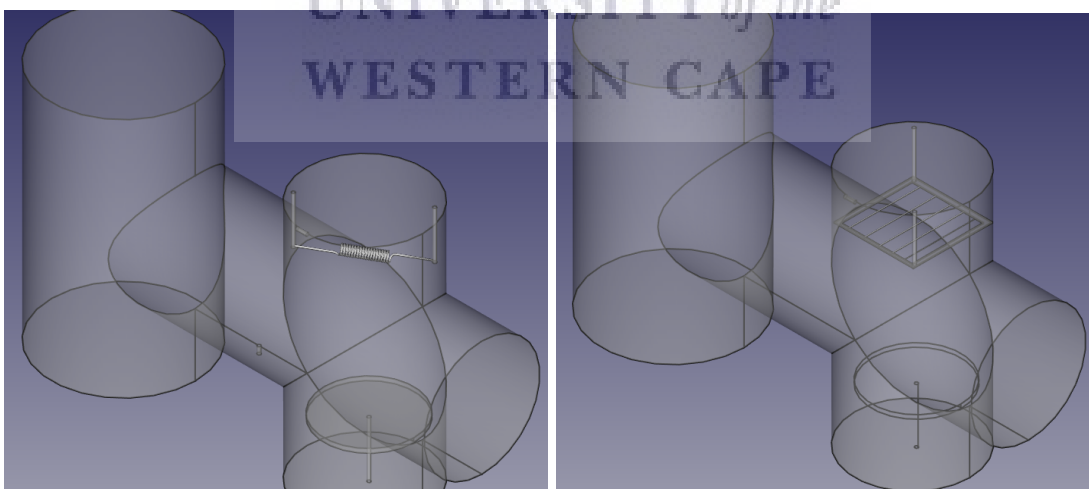


FIGURE 4.3: Top-view of the 3D CAD model for the coiled wire configuration (left) and straight wire configuration (right).

As seen from Figures 4.4 to 4.7, the mesh is built up from 1 mm hexahedrons distributed evenly throughout the geometry of the reactor. For the full mesh statistics for both wire configurations, the reader is referred to Appendix A.

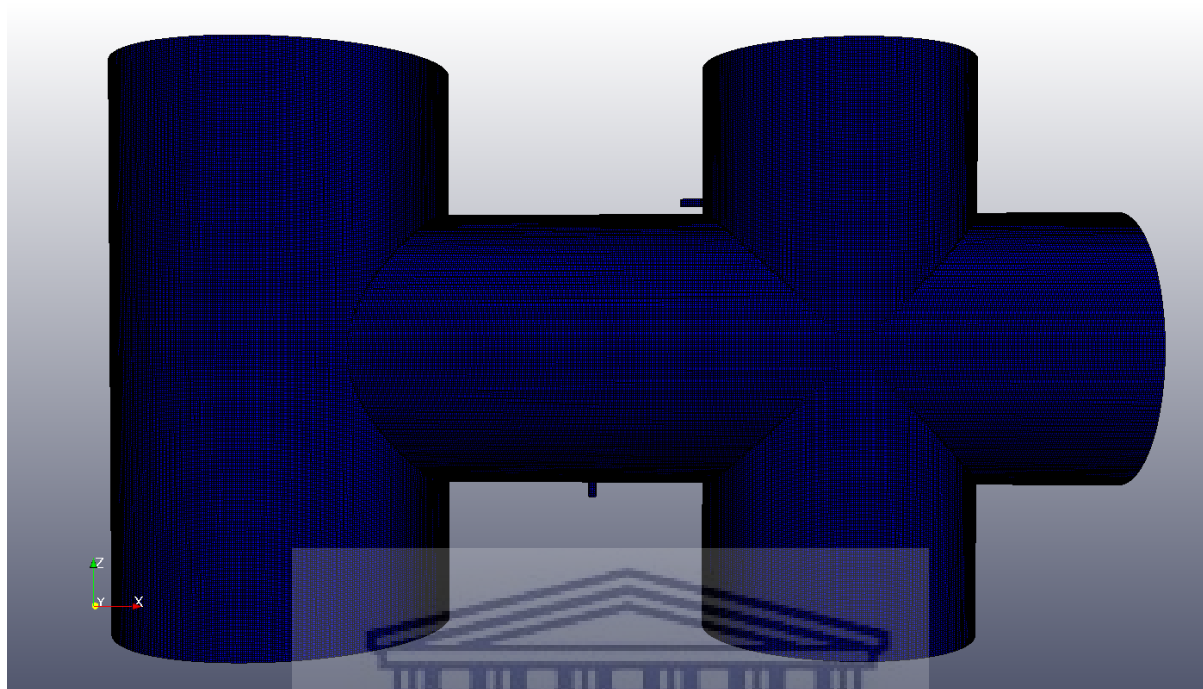


FIGURE 4.4: External surface mesh for the HWCVD reactor chamber

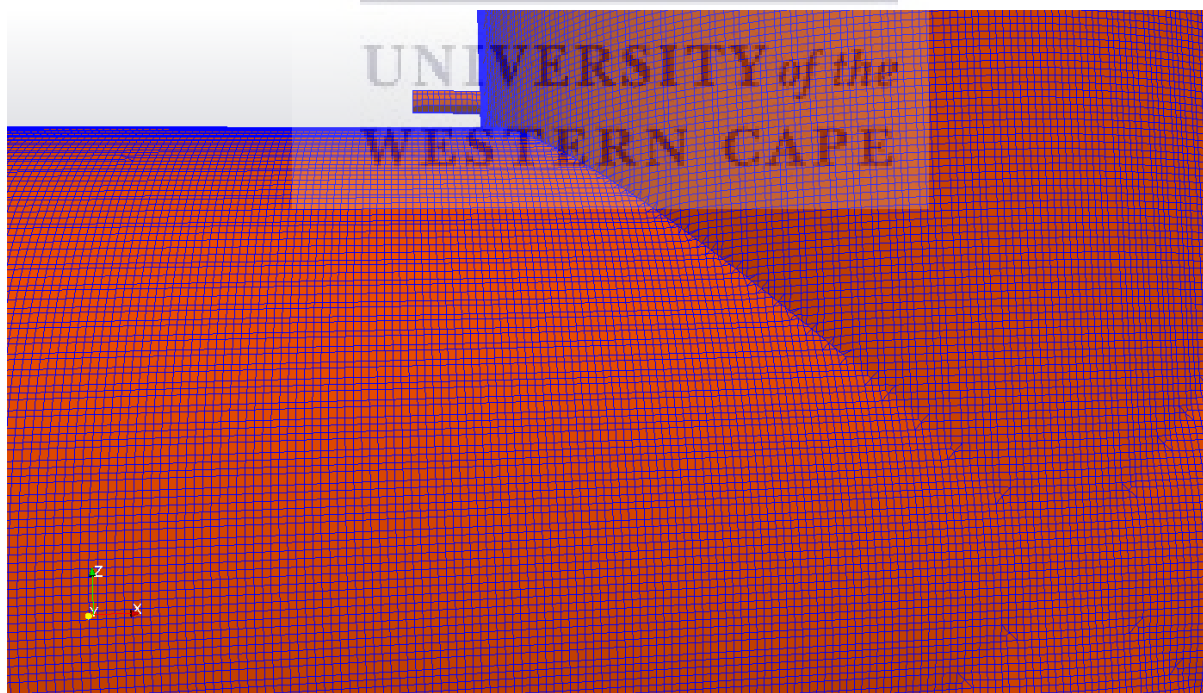


FIGURE 4.5: External surface mesh for the HWCVD reactor chamber including the inlet

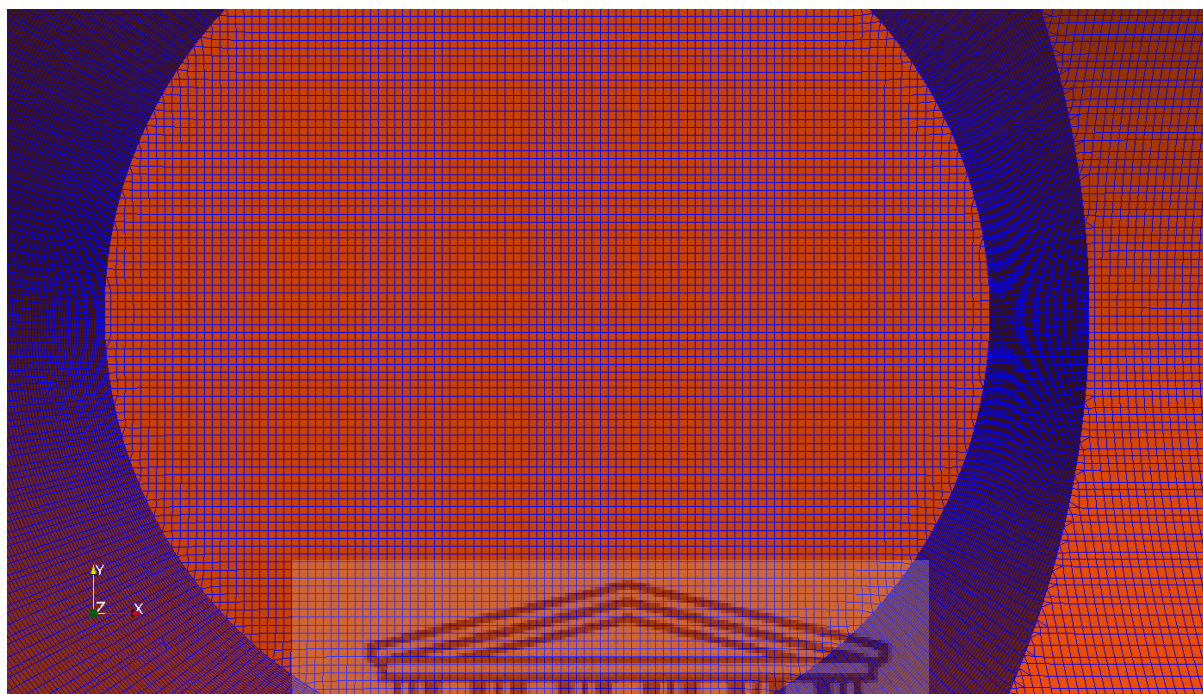


FIGURE 4.6: Surface mesh for the substrate holder



FIGURE 4.7: Surface mesh for the four parallel wires

## 4.2 Boundary Conditions

To enable the specification of boundary conditions, the mesh surface must be divided into different regions, also known as patches. Figure 4.8 shows the patch names for both the fluid and solid regions of the mesh used in this investigation.

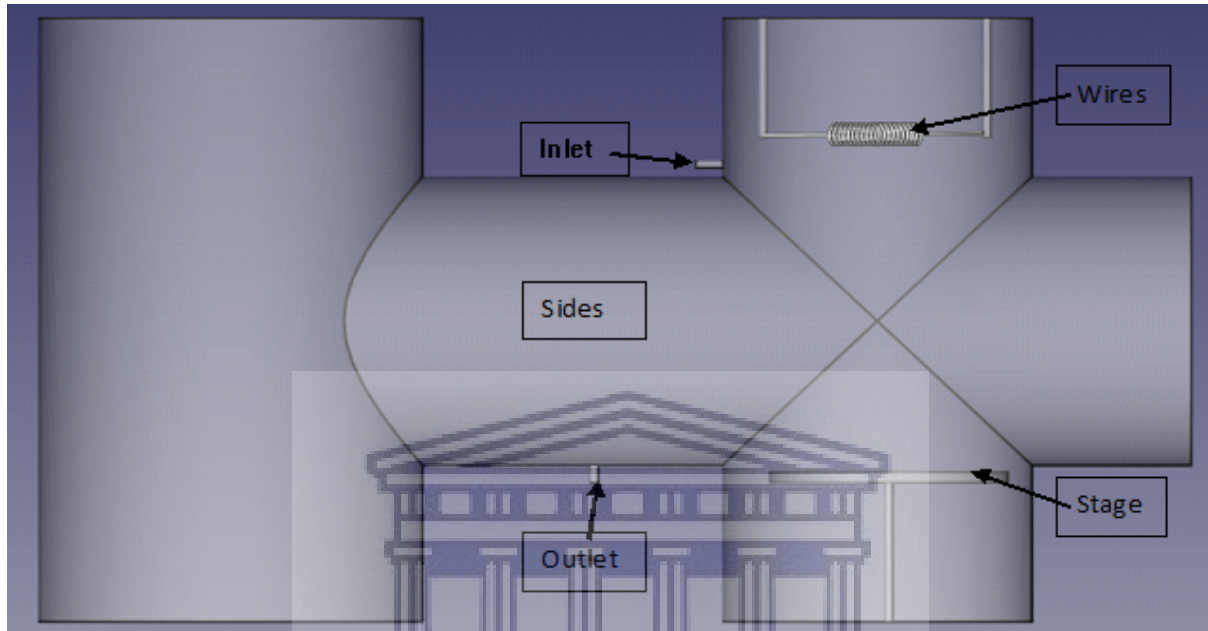


FIGURE 4.8: Patch Names

To distinguish between the four case setups, they are grouped based on the solver that each used. Group A, containing Case 1A and 2A, is for the buoyantSimpleFoam solver while Group B, containing Case 1B and 2B, is for the rhoSimpleFoam solver.

### 4.2.1 Group A

Group A, which uses the buoyantSimpleFoam solver, contains the two cases used in this investigation to map the conditions in the reactor vessel under high vacuum before the introduction of the hydrogen,  $H_2$ , precursor gas. As such, the internal pressure of the reactor is set to  $6 \times 10^{-6}$  mbar ( $6 \times 10^{-4}$  Pa) with an internal velocity of 0 m/s. This is done to ensure that the molecules that remain in the reactor at this low pressure can only move under the effects of gravity and thermal radiation felt from the heated wires. At these high vacuum pressures, heat transfer due to conduction and convection can be neglected [25], this is due to both modes of heat transfer being medium dependent, making radiation the dominant mode of heat transfer. This is reflected in the selection of boundary conditions for the different patches used in these cases, a summary of which is given in the table below. The boundary conditions are defined in the 0 folder of the case setup. These cases focus on four boundary conditions namely: velocity (U), pressure (p), temperature (T) and incident radiation intensity (G).



TABLE 4.1: Boundary Conditions for buoyantSimpleFoam

Patch	Boundary Condition			
	U	p	T	G
inlet	noSlip;	type calculated; value \$internalField;	type zeroGradient;	type MarshakRadiation; emissivityMode lookup; emissivity uniform 0.85;
outlet	noSlip;	type calculated; value \$internalField;	type zeroGradient;	type MarshakRadiation; emissivityMode lookup; emissivity uniform 0.85;
wires	noSlip;	type calculated; value \$internalField;	type fixedValue; value uniform 1873;	type MarshakRadiation; emissivityMode lookup; emissivity uniform 0.35;
sides	noSlip;	type calculated; value \$internalField;	type zeroGradient;	type MarshakRadiation; emissivityMode lookup; emissivity uniform 0.85;
stage	noSlip;	type calculated; value \$internalField;	type zeroGradient;	type MarshakRadiation; emissivityMode lookup; emissivity uniform 0.85;
internalField	uniform (0 0 0);	uniform 6e-4;	uniform 293;	uniform 0;

For the velocity field, the no-slip (noSlip) boundary condition is applied to all patches. With this, it is assumed that the fluid will have zero velocity relative to the boundary as to ensure that the fluid velocity at all fluid-solid boundaries are equal to that of the solid boundary. This is deemed to be a good approximation since there is no flow emanating from the reactor's inlet valve.

The pressure field uses the calculated boundary condition which fixes the pressure value at the patches to the assigned value for the internal field of the reactor. This is done to reproduce the effects of the turbo pump, which constantly tries to reduce the pressure inside the reactor, maintaining the pressure at a fixed value.

The temperature field uses two boundary conditions namely: zeroGradient and fixedValue. The zeroGradient boundary condition applies a zero-gradient condition from the patch to the internal field patch for the reactor. This is applied to all patches except the wires patch. The wires patch uses the fixedValue boundary condition which supplies a fixed gradient such that the patch values are calculated using the assigned value for the patch. The wires are set to a fixed temperature of 1600°C (1873 K) with the rest of the reactor, internalField, set to the ambient temperature of 20°C (293 K).

The last boundary condition used is that for the incident radiation intensity field. The MarshakRadiation boundary condition is used for all patches which use the calculated temperature from the case as a temperature for calculating the radiation intensity [28]. With radiation being material dependent, the MarshakRadiation boundary condition requires the specification of the material's emissivity to improve the accuracy of the radiation intensity calculation. The system

comprises of various parts fabricated from weathered stainless steel with an emissivity of 0.85 [43]. The wires are 99.9% Tantalum, 0.25 mm in diameter, with an emissivity of 0.35 at 1600°C [44].

Once all the boundary conditions are defined in the 0 folder of the case setup directory, the next step is to define the model constants for the simulation. This is done in the constant folder of the case setup directory. For buoyantSimpleFoam, the constant folder contains four files namely: g, where the gravitational acceleration is defined, radiationProperties where the radiation model is selected and its constants defined, turbulenceProperties where the turbulence model is selected and lastly the thermoPhysicalProperties file where the physical properties of the fluid are defined. In the turbulenceProperties file, the simulation type is set to laminar with the assumption that, prior to there being any flow inside the reactor, turbulent effects can be neglected. The P1 radiation model is selected in the radiationProperties file for the reasons outlined in section 2.3.3.3. The selections for the thermoPhysicalProperties file are discussed in the section that follows.

#### 4.2.1.1 Thermophysical modelling for buoyantSimpleFoam

The selections made in the thermoPhysicalProperties file are to ensure that the gas molecules in the simulation behave as realistically as possible. Coupled with this, the selection of the appropriate thermophysical models are of importance as it affects the accuracy of the results. The models selected for this investigation are given in Table 4.2.

TABLE 4.2: ThermoPhysicalProperties for gas phase (buoyantSimpleFoam)

Keyword	Selected Model	Description
type	heRhoThermo	Thermophysical model for fixed composition, based on density.
mixture	pureMixture	Calculation for passive gas mixtures
transport	const	Constant $\mu$ and Prandtl number
equationOfState	perfectGas	Ideal gas equation of state
thermo	hConst	Constant specific heat capacity and enthalpy of fusion
energy	sensibleEnthalpy	All heat leads to changes in temperature.

In these simulations, the gas that remains in the system at the high vacuum pressure was modelled to be trace amounts of air. This is a reasonable deduction as a result of air venting between depositions. The viscosity ( $\mu$ ) and Prandtl number (Pr) of air have the values of  $1.8 \times 10^{-5}$  Pa·s and 0.7 respectively [45]. The transport properties are modelled as constant. For Newtonian fluids, the viscosity is assumed to be independent of stress and subsequently, independent of pressure [46]. The thermal diffusivity of the gas is implicitly specified using Pr

which is the ratio between the momentum diffusivity and the thermal diffusivity [47].

$$Pr = \frac{\nu}{\alpha} = \frac{c_p \mu}{k_c} \quad (4.1)$$

$\nu$  is the kinematic viscosity and  $\alpha$  is the thermal diffusivity. The specific heat capacity ( $c_p$ ) was set to 1000 J/kg·K, which is the value of the specific heat capacity of air at room temperature as quoted from [48]. In the model,  $c_p$  and the enthalpy of fusion ( $H_f$ ) are set to constant values using the thermodynamic model hConst.

With all the boundary conditions defined and transport properties specified, the last step in the case setup would be to define the convergence criteria for each case. This is discussed in detail in section 4.3. For more detail on the boundary selection for Group A, the reader is referred to Appendix B.

### 4.2.2 Group B

Group B, which uses the rhoSimpleFoam solver, contains the two heat transfer cases used in this investigation. It models the conditions in the reactor vessel during the deposition process in which molecular hydrogen, H<sub>2</sub>, is converted into atomic hydrogen, H, by catalytic decomposition. The H<sub>2</sub> precursor gas enters the reactor vessel at the inlet patch at a constant volumetric flow rate of 60 standard cubic centimetres per minute (sccm). The outlet uses a pressure regulator to control the amount of H<sub>2</sub> gas in the system. At the outlet, the pressure is set to 0.1 mbar (10 Pa) to maintain a deposition pressure between 0.08 mbar (8 Pa) and 0.2 mbar (20 Pa). This allows for the relatively free movement of the H<sub>2</sub> gas inside the reactor consequently increasing the mean free path in an attempt to avoid any recombinations of the H molecules before it has reached the substrates. With the introduction of a medium into the reactor, all three modes of heat transfer need to be considered with the heat transfer from conduction and convection now emanating from the movement of the H<sub>2</sub> gas. The methodology of selecting the boundary conditions that encompass all these parameters is similar to that described for Group A with one key difference, the turbulent effects stemming from the flow can not be neglected for these cases. The turbulence boundary conditions used in this investigation are discussed later in the section that follows. Table 4.3 gives the boundary conditions used in the case setup for the rhoSimpleFoam solver.

The velocity field boundary condition for the stage, sides and wire is the same one used for the buoyantSimpleFoam case i.e. no-slip boundary condition. The inlet patch uses the boundary condition type flow Rate Inlet Velocity (flowRateInletVelocity) with a constant mass flow rate of  $8.988 \times 10^{-8}$  kg/s, which is equivalent to 60 sccm. The outlet patch uses the boundary condition

TABLE 4.3: Boundary Conditions for buoyantSimpleFoam

Patch	Boundary Condition		
	U	p	T
inlet	type flowRateInletVelocity; massFlowRate constant 8.988e-8;	type zeroGradient;	type fixedValue; value uniform 293;
outlet	type inletOutlet; value uniform (0 0 0); inletValue uniform (0 0 0);	type fixedValue; value uniform 1;	type inletOutlet; value uniform 293;
wires	type noSlip;	type zeroGradient;	type fixedValue; value uniform 1873;
sides	type noSlip;	type zeroGradient;	type zeroGradient;
stage	type noSlip;	type zeroGradient;	type zeroGradient;
internalField	uniform (0 0 0);	uniform 6e-4;	uniform 293;

type inlet-outlet (inletOutlet) which provides the outflow condition based on the value at the inlet and the internal field value.

The pressure field uses the zero gradient boundary condition for all patches, except the outlet patch, and its use in the simulations were outlined in Group A. The CADAR system uses a pressure regulator connected to the outlet valve, which regulates the opening and closing of the valve, to control the amount of precursor gas in the chamber which subsequently prevents the build-up of dangerous gasses inside the reactor vessel. During a deposition, the pressure regulator is set to 0.1 mbar (1 Pa) to maintain a deposition pressure between 0.08 mbar and 0.2 mbar. To duplicate this in the simulations, a constant pressure outflow of 1 Pa is applied to the outlet patch using the fixed-value (fixedValue) boundary condition. The internal field value remains at  $6 \times 10^{-4}$  Pa to ensure we start with high vacuum pressures inside the reactor.

The temperature field uses the same boundary conditions as was outlined in Group A.

#### 4.2.2.1 Turbulence modelling

The turbulence model,  $k - \epsilon$ , used in this investigation requires two input parameters, namely  $k$  and  $\epsilon$ . In this section, the approach on how to estimate these parameters for the case is given as well as an overview of how the turbulent boundary conditions are applied to each patch. Table 4.4 contains the list of turbulent boundary conditions used in the Group B case setup.

The turbulent viscosity (nut) field forms part of the set of boundary conditions required for the RANS turbulence modelling. The inlet and outlet patches use the inlet-outlet (inletOutlet) boundary condition with an inlet value initialised to the internal field value. The remaining patches use the nutkWallFunction boundary condition with a value initialised to the internal field value as well. The nutkWallFunction forms part of the nutWallFunction abstract class directory which calculates the turbulent viscosity at the boundary patches. The nutkWallFunction wall

TABLE 4.4: Turbulence Boundary Conditions

Patch	Boundary Condition		
	nut	k	epsilon
inlet	type inletOutlet; inletValue internalField; value internalField;	type inletOutlet; inletValue internalField; value internalField;	type inletOutlet; inletValue internalField; value internalField;
outlet	type inletOutlet; inletValue internalField; value internalField;	type inletOutlet; inletValue internalField; value internalField;	type inletOutlet; inletValue internalField; value internalField;
wires	type nutkWallFunction; value internalField;	type kqRWallFunction; value internalField;	type epsilonWallFunction; value internalField;
sides	type nutkWallFunction; value internalField;	type kqRWallFunction; value internalField;	type epsilonWallFunction; value internalField;
stage	type nutkWallFunction; value internalField;	type kqRWallFunction; value internalField;	type epsilonWallFunction; value internalField;
internalField	uniform 0;	uniform 3.1e-8;	uniform 5.9e-9;

function provides the turbulent viscosity,  $\nu_t$ , based on the turbulent kinetic energy,  $k$ , at the different patches [49]. Using this boundary condition, the turbulent viscosity,  $\nu_t$ , is calculated using:

$$\nu_t = \nu \cdot \left( \frac{\kappa y^+}{\ln(Ey^+)} - 1 \right) \quad (4.2)$$

where  $\kappa$  is the thermal conductivity,  $\nu$  is the kinematic viscosity of the fluid,  $E$  is energy and  $y^+$  is known as the dimensionless wall distance.  $y^+$  is one of the more commonly used parameters when judging the acceptability of wall functions and is defined as:

$$y^+ = \frac{yu_\tau}{\nu} \quad (4.3)$$

where  $u_\tau$  is the friction velocity and  $y$  is the absolute distance to the wall. OpenFOAM uses the  $y^+$  parameter in the solution algorithm for  $\nu_t$  to determine the form of the equation used to solve for  $\nu_t$  at each boundary face and cell center. At the start of each iteration,  $\nu_t$  is initialized with a value of 0. The nutkWallFunction then provides two conditions, based on the value of  $y^+$ , to solve for  $\nu_t$ .

When  $y^+ > yPlusLam$ :

$$\nu_t = \nu \cdot \left( \frac{\kappa y^+}{\ln(Ey^+)} - 1 \right) \quad (4.4)$$

When  $y^+ < yPlusLam$ :

$$\nu_t = 0 \quad (4.5)$$

where `yPlusLam` is simply the value of  $y^+$  at the edge of the laminar sublayer.

The second boundary field required for the RANS turbulence modelling is the turbulent kinetic energy ( $k$ ) field. Similar to the nut field, the inlet and outlet patches use the inlet-outlet boundary condition with the value initialised to the internal field value. All other patches use the `kqRWallFunction` boundary condition. This boundary condition is a sub-directory that forms part of the `KqRWallFunctions` abstract class used to calculate  $k$  at each boundary patch. Unlike the nut field,  $k$  is not initialised to zero at the start of each new iteration but is rather calculated using the calculated value from the previous iteration. This means that an initial approximation for the value of  $k$  needs to be provided in the field patch. An initial approximation for  $k$  at the inlet can be obtained using Equation (3.18) and this value is applied to the internal field value. To solve for the new value of  $k$ , the algorithm creates a zero-Gradient boundary condition across all patches and calculates the new value of  $k$ , using Equation (3.18), for each face and cell center during every new iteration. To estimate the initial value for  $k$ , we begin by estimating the turbulent intensity,  $\mathbf{I}$ . According to the SimScale documentation [50], for high speed flow into complex geometry, the typical turbulence intensity ranges between 5 % and 20 %. A turbulent intensity of 7% was used at the inlet for this investigation as this is the default value in OpenFOAM. Using this, the turbulent length scale can be calculated using Equation (3.24):

$$l = \mathbf{I}d_h = (0.07)d_h \quad (4.6)$$

where

$$d_h = 4 \frac{\text{Area}}{\text{Perimeter}} = 4 \frac{\pi r^2}{2\pi r} = 2r = d \quad (4.7)$$

where  $d$  is the diameter of the inlet pipe. Where  $d$  for the inlet pipe of the CADAR HWCVD chamber is equal to 4 mm, substituting this back into Equation (4.6), we obtain:

$$l = (0.07)(4 \times 10^{-3}) = 0.28\text{mm} \quad (4.8)$$

The next step would be to calculate the mean velocity,  $u_{mean}$ , at the inlet pipe, using:

$$u_{mean} = \frac{Q}{A} \quad (4.9)$$

where  $Q$  is the mass flow rate,  $Q = 8.9 \times 10^{-8} \text{m}^3/\text{s}$ , and  $A$  is the cross sectional area of the pipe,  $A = \pi r^2$ . Using this we obtain:

$$u_{mean} = \frac{Q}{A} = \frac{8.9 \times 10^{-8}}{\pi(2 \times 10^{-3})^2} = 7.15 \times 10^{-3} m/s \quad (4.10)$$

The value from Equation (4.9) and  $\mathbf{I}$  can now be used in conjunction with Equation (3.18) to estimate the value for  $k$ .

$$k = \frac{3}{2}(7.15 \times 10^{-3} \cdot 0.07)^2 = 3 \times 10^{-8} m^2 s^{-2} \quad (4.11)$$

This value is then applied to the internal field value for the  $k$  boundary condition.

The last boundary condition required for the RANS turbulence modelling is the epsilon,  $\epsilon$ , field. Similar to the previous boundary condition described above, wall functions are used to calculate the value for  $\epsilon$  throughout the solution process namely, the epsilonWallFunction boundary condition. Unlike the  $k$  and  $\nu_t$  wall functions, in the epsilon wall function the value for the cell center, rather than the value for the cell face, is calculated [49]. This is done by using a “weights” parameter in the calculation of  $\epsilon$ . An example of this would be; if three faces of one cell use the epsilonWallFunction boundary condition, the “weights” of this cell is three. The “weights” is calculated to obtain the value of epsilon at the cell center,  $\epsilon_{center}$ .

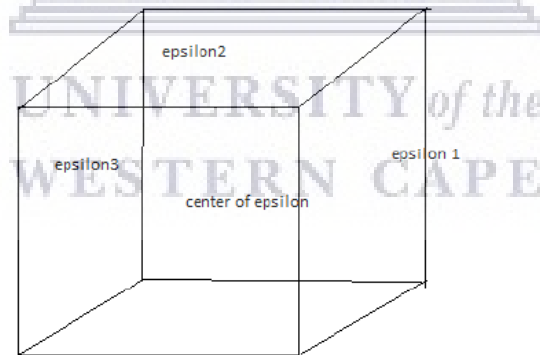


FIGURE 4.9: The calculation of the cell center of epsilon

In the figure above, the epsilon value for each face is calculated. The value at the cell center is then obtained using:

$$\epsilon_{center} = \frac{1}{3}\epsilon_1 + \frac{1}{3}\epsilon_2 + \frac{1}{3}\epsilon_3 \quad (4.12)$$

where the values for  $\epsilon_1$ ,  $\epsilon_2$  and  $\epsilon_3$  are calculated using Equation (3.23). The epsilonWallFunction boundary condition also requires an initial estimation for the value of  $\epsilon$  to be applied to the internal field. The estimated value is obtained using Equation (3.23) at the inlet patch. For the calculation, the only requirements are the values for  $k$  and  $l$  which were previously calculated in Equations (4.11) and (4.8) respectively. Substituting these values into the equation yields:

$$\epsilon = (0.09)^{\frac{3}{4}} \frac{(3 \times 10^{-8})^{\frac{3}{2}}}{0.28 \times 10^{-3}} = 5.9 \times 10^{-9} m^2 s^{-3} \quad (4.13)$$

In summary, these boundary conditions and field values are used by the  $k - \epsilon$  model in the calculation of all the turbulent parameters during each iteration in the solution process. The final step in the setup of the boundary conditions for the Group B cases would be to define the thermophysical properties for the cases. These selections are discussed in the section that follows.

#### 4.2.2.2 Thermophysical modelling for rhoSimpleFoam

The selections for the thermo-physical properties used in this investigation for these cases are given in Table (4.5) below.

TABLE 4.5: ThermoPhysicalProperties for gas phase (rhoSimpleFoam)

Keyword	Selected Model	Description
type	hePsiThermo;	Thermophysical model for fixed composition, based on compressibility
mixture	pureMixture;	Calculation for passive gas mixtures
transport	sutherland;	Uses Sutherland's Law to calculate viscosity
equationOfState	perfectGas;	Ideal gas equation of state
thermo	hConst;	Constant specific heat capacity and enthalpy of fusion
energy	sensibleInternalEnergy;	Specifies that internal energy is the form used

Many of the selections made for the thermo-physical properties in these cases are similar to that made for the cases in Group A with the only difference being in the selections made for the type, transport and energy parameters. rhoSimpleFoam uses the type hePsiThermo model, which operates on the compressibility of the gas, Sutherland's equations as the transport model and the sensible internal energy model for the energy equations. The internal gas is modelled the same way as for the gas in the cases for Group A with two additional input parameters for the specific heat capacity,  $c_p$ , and the heat of fusion,  $H_f$ . The values used for  $c_p$  and  $H_f$  were 14130 J/Kg.K and 587 J/mol respectively, which are the values for these constants for the hydrogen gases entering the system according to the online databases[51, 52]. The final selection



made for these cases was the use of Sutherland's Law to calculate the dynamic viscosity,  $\mu$ , as a function of temperature [46, 53]. In OpenFOAM,  $\mu$  is calculated using:

$$\mu = \frac{A_{su} T^{3/2}}{T + T_{su}} \quad (4.14)$$

where  $T$  is the temperature,  $T_{su}$  is the Sutherland temperature and  $A_{su}$  is the Sutherland coefficient which is calculated using:

$$A_{su} = \frac{\mu_0 (T_0 + T_{su})}{T_{su}^{3/2}} \quad (4.15)$$

where  $\mu_0$  is the dynamic viscosity at the reference temperature  $T_0$ . Sutherland's law allows you to calculate the viscosity,  $\mu$ , for gas at a given temperature,  $T$ , using information about the gas at a reference temperature,  $T_{su}$ , for which the value  $\mu_0$  is known. As such,  $T_{su}$  and  $A_{su}$  in Equation (4.14) are used as coefficients in Sutherland's equation to store the known values of  $\mu_0$  and  $T_0$  needed to determine the new value of  $\mu$  at the calculated temperature  $T$  in the model. For hydrogen,  $T_{su}$  equals 96.67 K with  $\mu_0$  equalling  $8.441 \times 10^{-6}$  Pa·s at  $T_0$  of 273.11 K [54, 55]. Using these values along with Equation (4.15),  $A_{su}$  is calculated to equal  $6.89 \times 10^{-7}$  kg/msK<sup>1/2</sup>. The values for  $A_{su}$  and  $T_{su}$  are then updated in the thermoPhysicalProperties file to be used by the transport model.

Once all the boundary conditions have been defined, the last step in the case setup procedure would be to define the convergence criteria for each case. For more detail on the boundary selection for Group B, the reader is referred to Appendix C.

### 4.3 Convergence Criterion

As mentioned in section 3.5, residuals can be used to monitor the convergence of the simulation. In the *fvSolutions* file inside the system folder, the residual control function can be used to monitor the residuals for each iteration. In this function, the convergence criteria are defined with a specific value assigned to each parameter in question. For the cases of Group A, the radiation study, the parameters  $\mathbf{u}$ ,  $p_{rgh}$  and the radiation intensity,  $G$ , are used as the indicators of convergence with assigned values of  $1 \times 10^{-3}$ ,  $1 \times 10^{-2}$  and  $1 \times 10^{-3}$  respectively. Once the error between two successive iterations falls below the specified value for all the parameters, the simulation is deemed to have converged and the algorithm will stop running. These values are chosen to ensure that the absolute error in the calculations is below 1% for each parameter which increases the accuracy of the calculated value. The cases of Group B, the deposition study, use the parameter  $\mathbf{u}$  and the internal energy,  $e$ , as the indicators of convergence with an

assigned value of  $1 \times 10^{-3}$  for both parameters. The figures below show the residuals of the four simulations at the point of convergence.

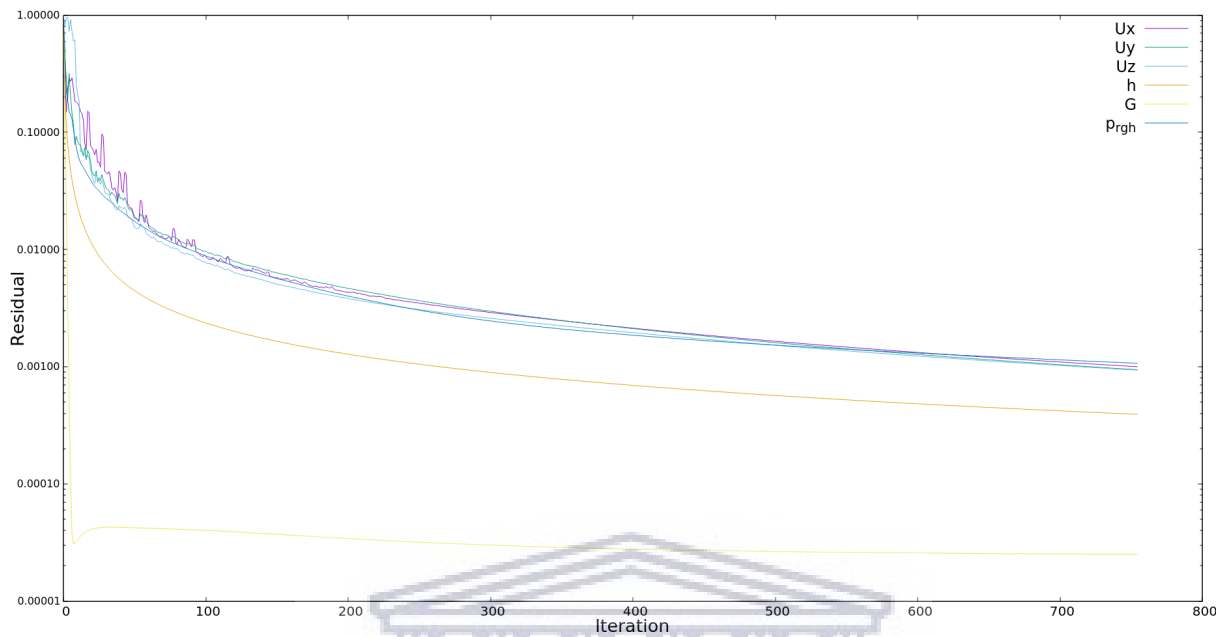


FIGURE 4.10: A plot of the residuals for Case 1A

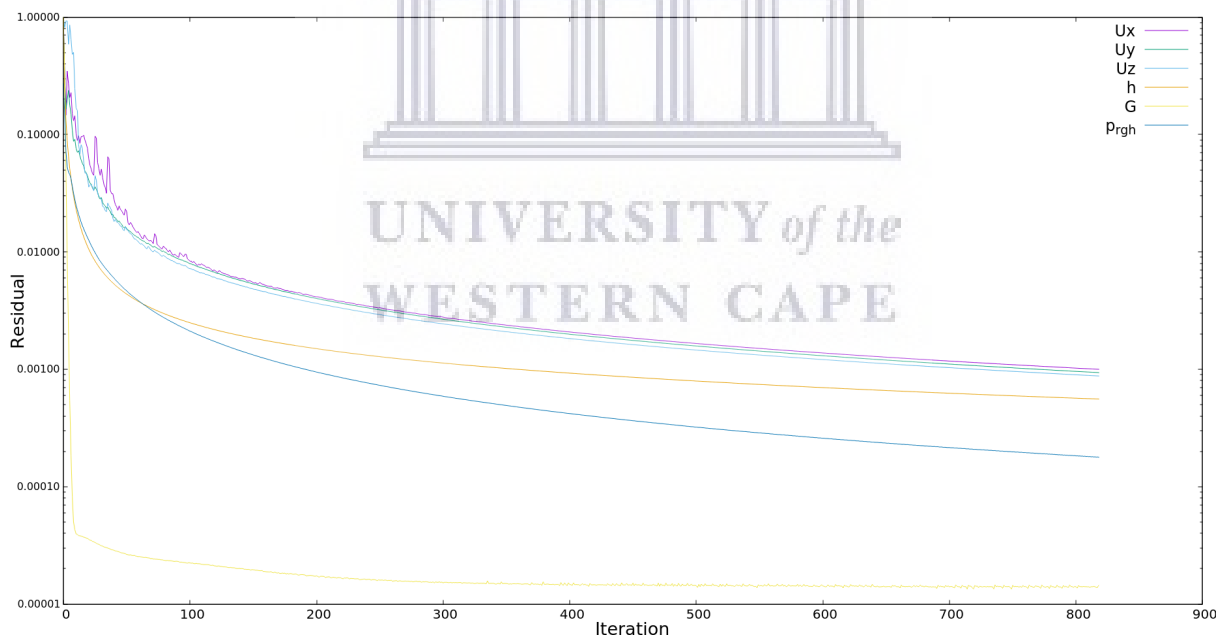


FIGURE 4.11: A plot of the residuals for Case 1B

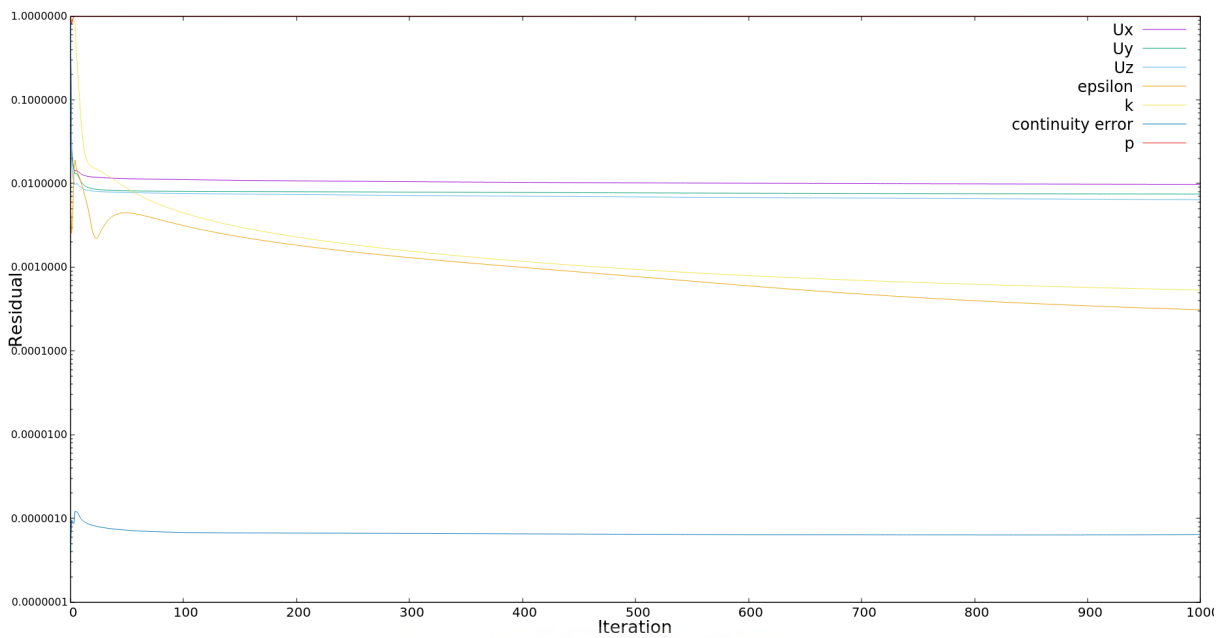


FIGURE 4.12: A plot of the residuals for Case 2A

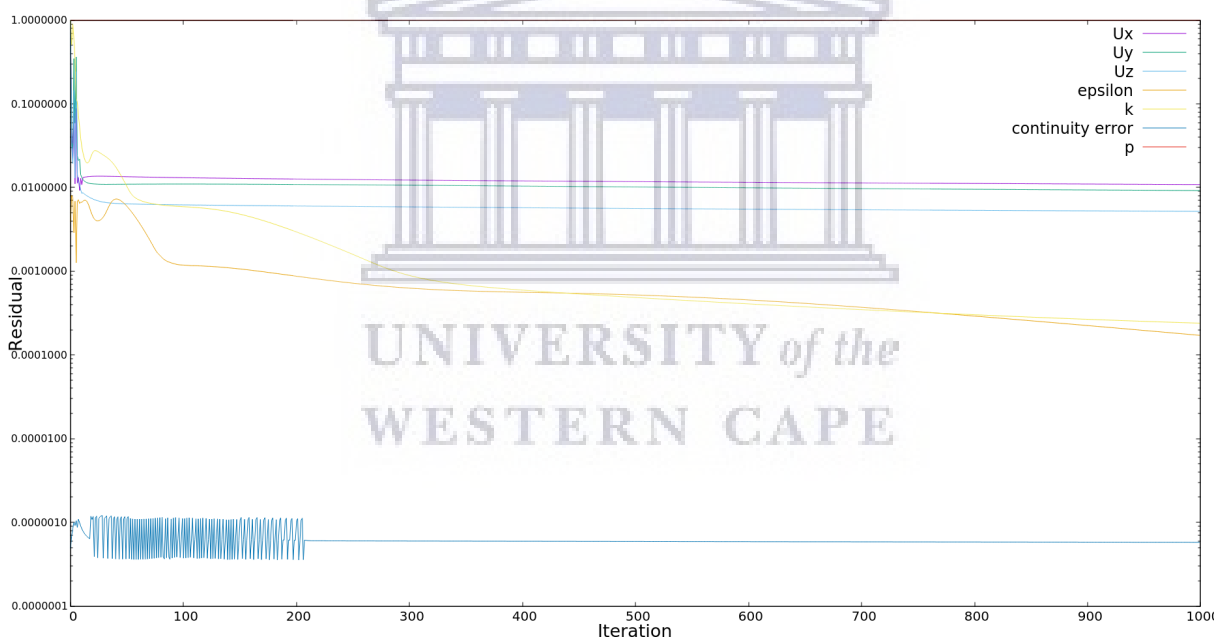


FIGURE 4.13: A plot of the residuals for Case 2B

The residual plots show the exact point when convergence was reached with the time taken to reach convergence stored in the log file for each case. The table below shows the number of iteration as well as the computational time taken for the simulations to converge.

To reduce the computational time for each simulation, relaxation factors were applied to each parameter. Relaxation factors are defined to help guide the simulations towards convergence. In general, heat transfer simulations converge slowly, so it is recommended by Bakker [56] to use under-relaxation factors of above 0.9 for enthalpy. The optimal combination of relaxation

TABLE 4.6: Number of Iterations and the time taken for simulations to reach convergence

Case	Number of Iterations	Computation Time (Hours)
1A	764	~36
1B	819	~28
2A	1000	~7
1B	1000	~7

factors used in these cases were found by the trial and error approach to achieve a balance between computational time and results accuracy.



# Chapter 5

## Results and Discussion

### 5.1 Introduction

In this chapter, the sets of simulation results for the two different wire configurations are presented. These results include that for temperature, pressure and gas velocity where applicable. For the visual representation of the results, the post-processing tool, Paraview, is used to visually map the contours of the parameters under investigation inside the reactor vessel.

For most of the results displayed in this section, a cross sectional bisection was made, along the plane normal to the y-axis, on the reactor vessel to expose its internal geometry as well as to reveal the simulation results of the fluid domain which otherwise would not be visible on the boundary layers of our domain.

### 5.2 Pressure Profile

#### 5.2.1 Case 1A

As previously mentioned in section (4.2), the base pressure of the reactor chamber is  $6 \times 10^{-4}$  Pa. This is set as the initial pressure of the system. Figure 5.1 illustrates the model results for the initial pressure profile inside the reactor vessel. This is displayed to identify whether the pressure gradient inside the reactor is uniform when the system reaches its base pressure. The pressure profile in Figure 5.1 is taken before the wire is turned on so the temperature is kept constant at this point in the simulation so that its effects on the pressure profile can be neglected.

In Figure 5.2, the pressure variations in the reactor vessel at the point of convergence are illustrated to ascertain the effects of the wire temperature on the pressure profile. From Figure



FIGURE 5.1: Initial pressure for the straight wire configuration (Pa)

5.2, we see a slight increase in pressure in the area surrounding wires. This is most likely due to the increase in temperature at this point. The increase in pressure, however, is so small that its effect on the overall pressure can be neglected. This, in essence, implies that the system is still at its base vacuum pressure and the heating of the wires does not disturb the pressure in the reactor vessel.

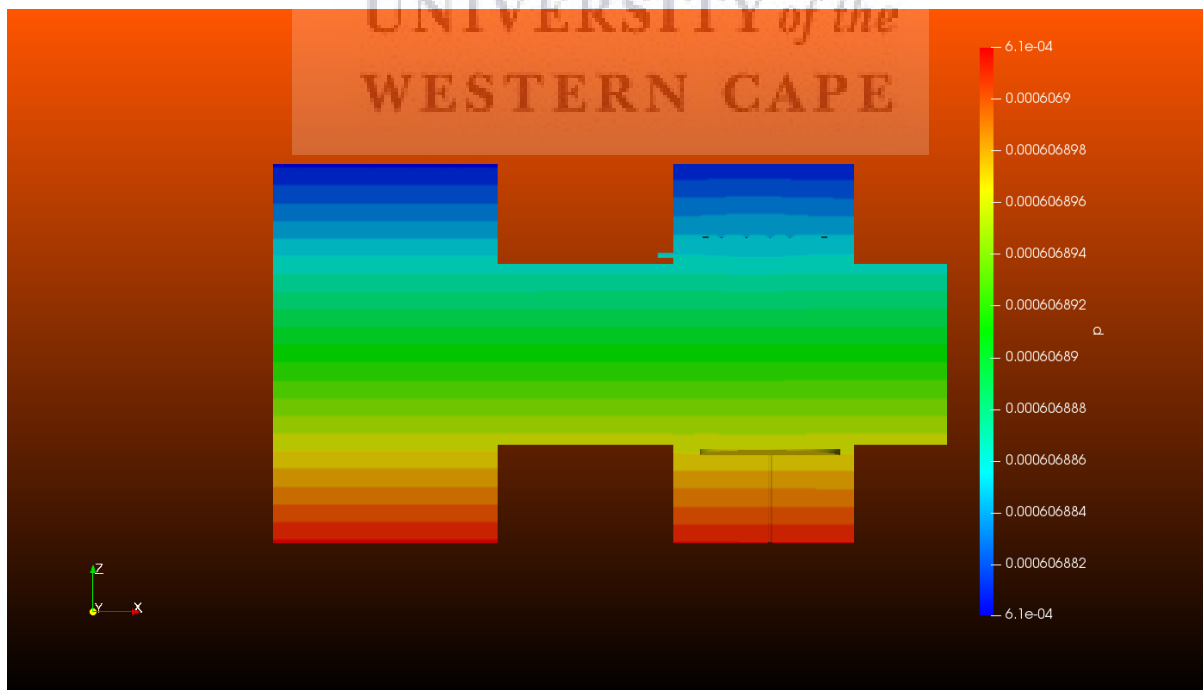


FIGURE 5.2: Convergence pressure for the straight wire configuration (Pa)

### 5.2.2 Case 2A

Similar to Case 1A, for this case using the coiled wire configuration, the initial pressure in the model reactor is set to the base pressure of the system, this can be seen in Figure 5.3. Figure 5.4 shows the pressure at the point of convergence for this case.



FIGURE 5.3: Initial pressure for the coiled wire configuration (Pa)

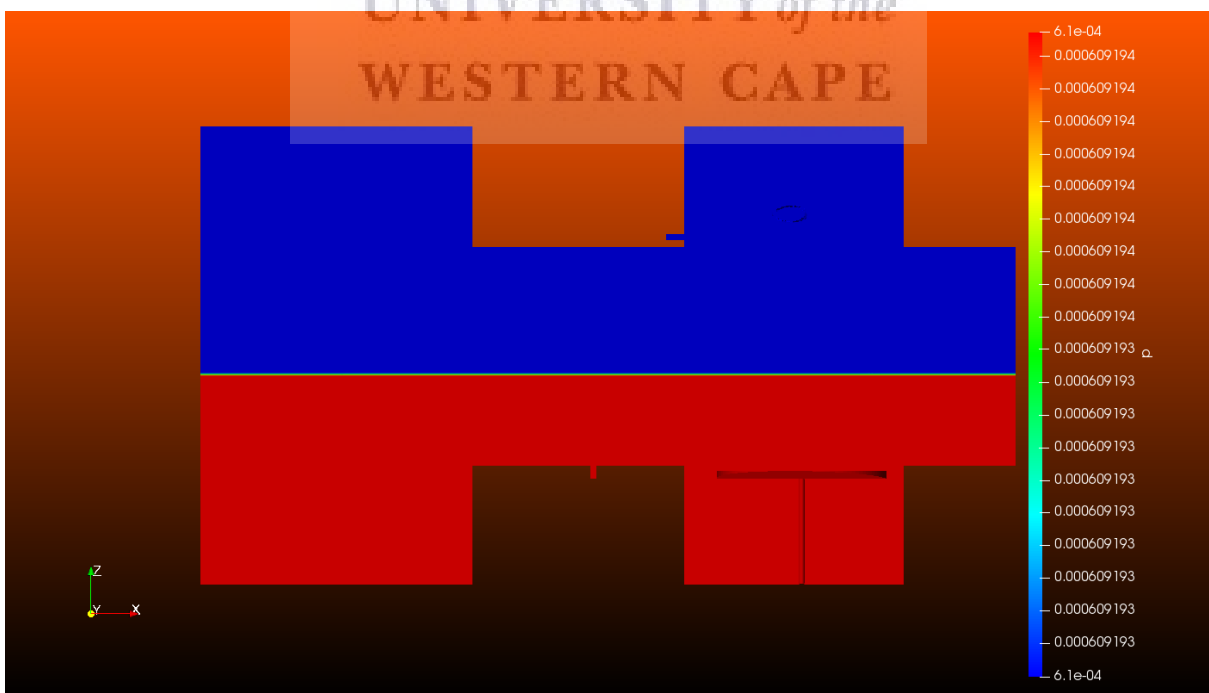


FIGURE 5.4: Convergence pressure for the coiled wire configuration (Pa)

Figure 5.4 shows that the pressure in the reactor remains at the base pressure even with the increase in temperature inside the system. Although the pressure profile in Figure 5.4 shows two very distinct colours, blue and red for the top and bottom halves of the reactor respectively, the magnitude of these two regions is to the order  $6 \times 10^{-4}$  Pa which is the base pressure for the reactor. The green line running through the center of the system indicates a diffuse region. In this region, the mesh is too coarse to accurately calculate the pressure creating a pressure boundary. Given that the width of the boundary relative to the system is so small, coupled with how fine the mesh used in this investigation was, the effects of the boundary layer can safely be neglected.

### 5.2.3 Case 1B and 2B

In these cases, the main goal is to determine whether the system can maintain the desired deposition pressure, as listed in section 4.2.2, with a constant flow of the precursor gas into the reactor vessel. Given that both cases use the same pressure and inlet flow conditions, the result for only Case 1B is presented. Figure 5.5 shows the deposition pressure at the point of convergence for both cases.

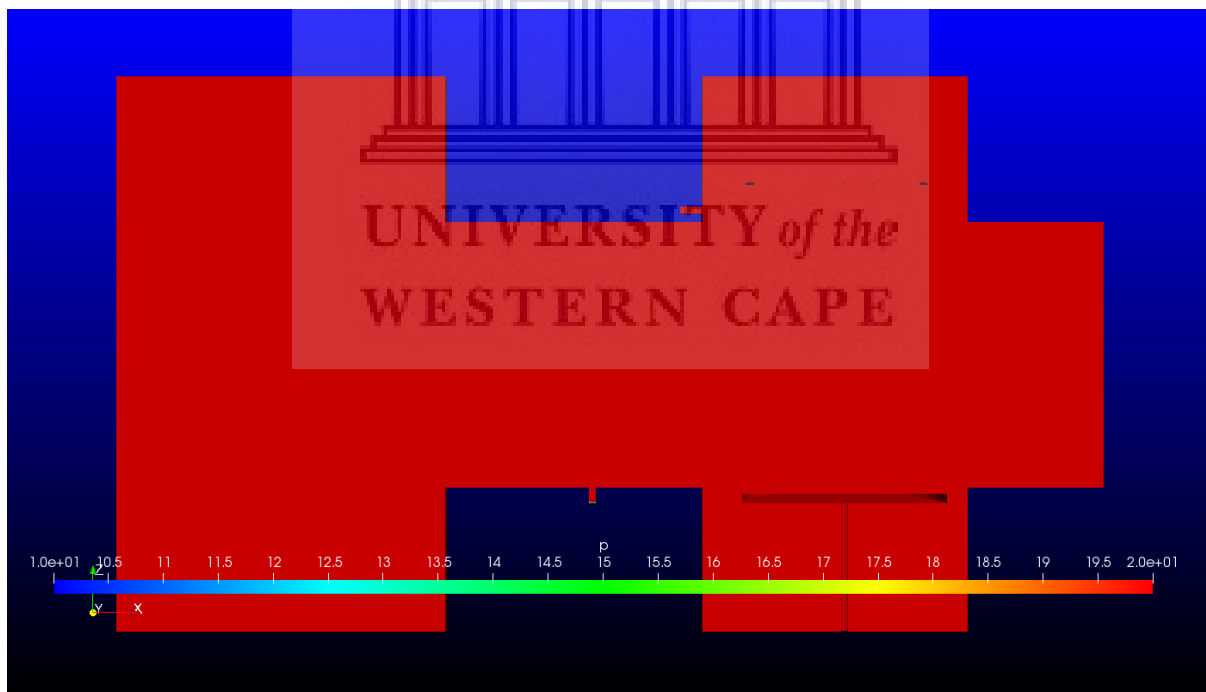


FIGURE 5.5: Deposition pressure at Convergence for both sets of wire configurations (Pa)

Figure 5.5 shows that at convergence, the pressure inside the reactor stabilizes to 20 Pa which is maintained by a constant, 10 Pa, outflow of pressure at the opening of the turbo pump i.e. the outlet patch. During a test deposition, using the same parameters as in the model, the pressure inside the reactor was found to be oscillating between 0.09 mbar (9 Pa) and 0.2 mbar (20 Pa).



This thus suggests that the model gives a reasonable approximation for the deposition pressure inside the reactor. The oscillations in the pressure stem from the precision and location of the pressure gauge which is positioned directly above the turbo pump valve opening. The pressure gauge takes a new pressure reading that is used to control the opening and closing of the outlet valve. The oscillations are thus caused by the variation in the time taken to open and close the outlet valve.

### 5.3 Velocity Profile

In this section, the intent is to map both the velocity and the position of the precursor gas during its progression through the reactor chamber. This is done to identify areas, if any, of gas build-up as well as identify the magnitude of the gas velocity as it flows over and or through the different components of the reactor chamber. To achieve this, streamline plots, which are a built-in function of the Paraview post-processing tool, are used to highlight the flow pattern of the gas from the inlet valve to the outlet valve of the reactor chamber where the colour of the streamlines indicates the magnitude of the parameter in question. The streamline plots allow for a specified number of points to be plotted, up to a maximum of 10 000 points, for one complete cycle in the given geometry i.e. one cycle refers to the movement of the gas from the inlet to the outlet of the geometry. To avoid convolution in the images, 1000 points are used to create the streamline plots in this study, the results of which can be seen in Figure 5.6 below.

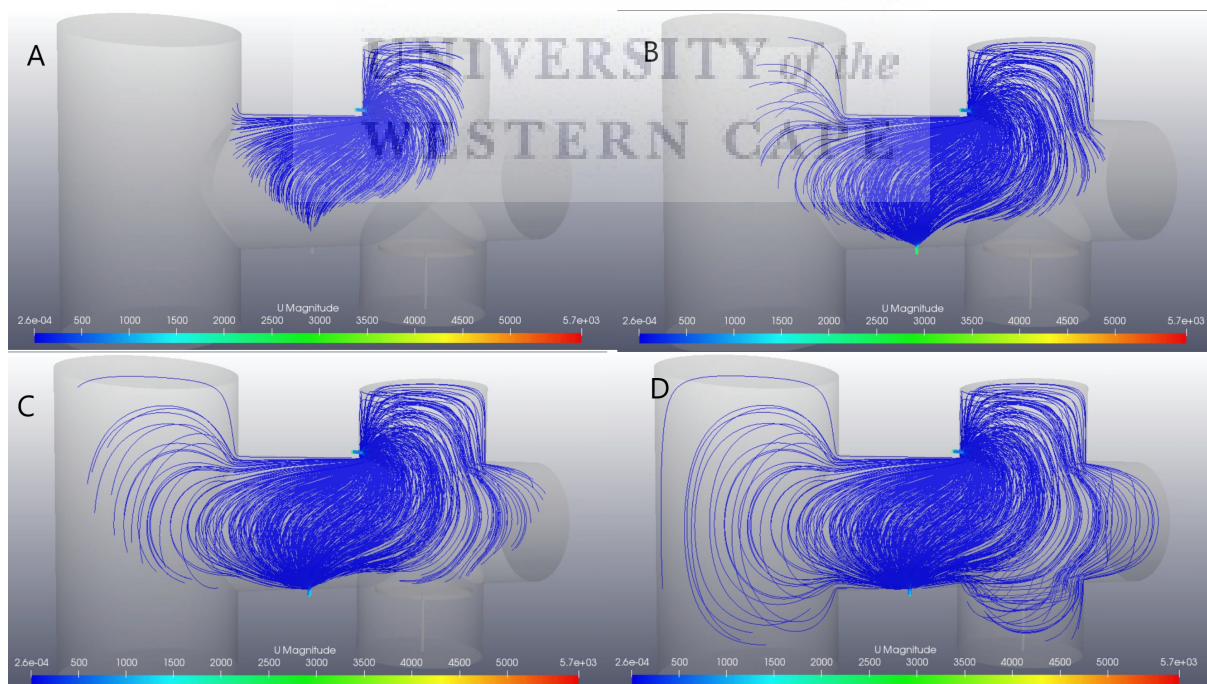


FIGURE 5.6: Velocity and progression of gas through the reactor chamber ( $\text{ms}^{-1}$ )

From Figure 5.6, we observe a reduction in the velocity magnitude as the gas enters the main reactor vessel when compared to the calculated value, in Equation 4.10, for the inlet. This is due to the drastic change in volume, from the inlet pipe to the reactor vessel, experienced by the gas as it enters the deposition chamber, illustrated in Figure 5.6 as the colour goes from light blue at the inlet to a darker blue as the gas goes to the rest of the chamber. In section 4.2.2.1, the mean velocity inside the inlet pipe was calculated to be  $7.15 \times 10^{-3} \text{ ms}^{-1}$  which reduces to  $2.6 \times 10^{-4} \text{ ms}^{-1}$  as the gas enters the main deposition chamber and remains constant throughout the progression of the gas through the vessel. This constitutes approximately a 10% reduction in the velocity magnitude. The results also show large increases in velocity, in the order of  $10^3 \text{ ms}^{-1}$ , at the inlet and the outlet. Pflug et al.[8], in their work on, concluded that CFD is unable to model results near small gaps in the geometry accurately and this is deemed to be an appropriate conclusion for this scenario. An alternative approach to solving the velocity for these areas would be to use Boltzmann's equation.

There are two dominant forces affecting the flow of the gas in the reactor chamber. These are, a pushing force from the pressure inside the narrow inlet pipe and a pulling force from the suction created by the opening of the outlet valve. Gravity also affects the motion of the gas but to a lesser extent than the two aforementioned forces. This is due to the precursor gas,  $\text{H}_2$ , molecules being lighter than air causing them to rise to the top the chamber rather than sink to the bottom. Knowing this, the results indicate that the suction created at the outlet dominates over the force created at the inlet as the gas seems to congregate in the region above the outlet as it awaits extraction. The congregation of the gas away from the deposition region would suggest that the chance of having collisions and reactions between decomposed and undecomposed gas would decrease significantly. This result can be considered as the first step in determining the optimal placement of the substrate in the chamber. The optimal placement of the substrate in the reactor is known as the  $d_{f-s}$  distance and is the separation distance between the filament and the substrate. An optimized  $d_{f-s}$  distance allows you to have better control over which created species form on the surface of the substrates during the deposition process. The idea of determining the optimal  $d_{f-s}$  distance is discussed further in section 5.5.

In summary, the results for both the gas velocity and position in the deposition region suggests that the current setup would be favourable as there is enough gas supply in the deposition region for the replenishment of the decomposed precursor gas while there's also no build-up of gas in this region that could lead to undesired species being created on the substrates.

## 5.4 Temperature Profile

### 5.4.1 Radiation Cases

In this section, the model's temperature results for the cases with no gas flow inside the reactor are presented for the two wire configurations.

#### 5.4.1.1 Case 1A

In the figure below, the initial temperature profile for the straight wire configuration is illustrated. Figure 5.7 shows that the entire system is set to a starting temperature of 293.15 K with the wire set to 1873.15 K as per the boundary conditions. The intent of the simulation was to see the effect of the wire temperature on the overall temperature of the chamber.

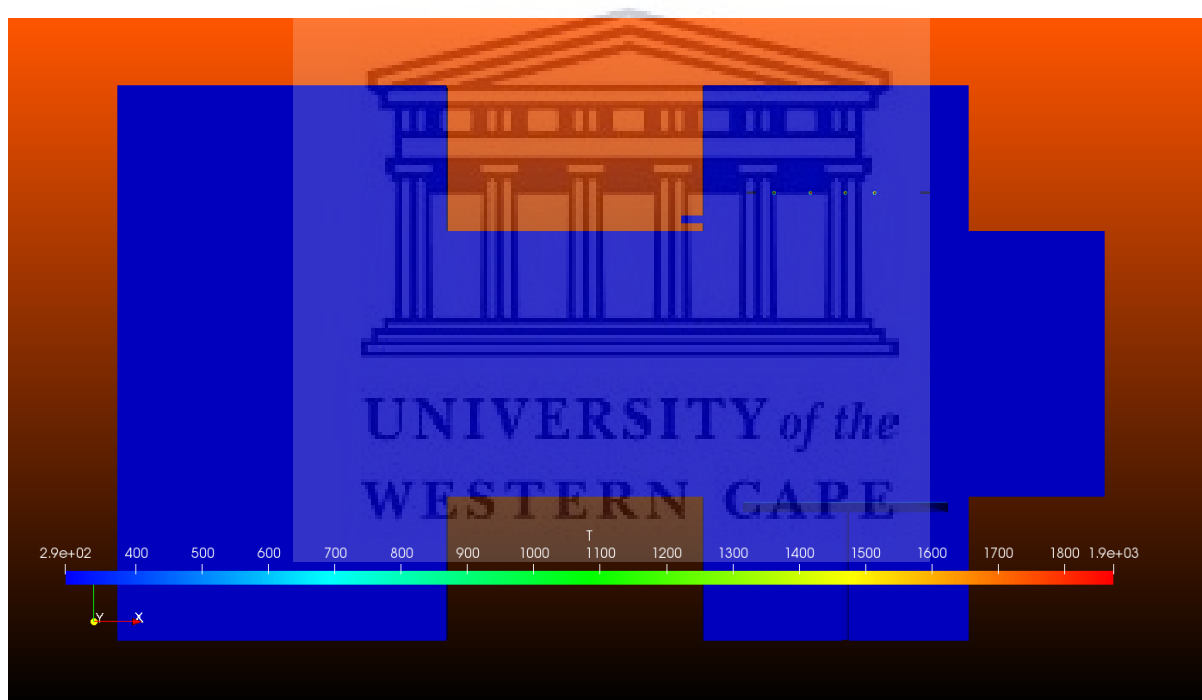


FIGURE 5.7: Initial temperature for straight wire configuration (K)

The temperature profile inside the reactor vessel at the point of convergence is illustrated in Figure 5.8. From Figure 5.8, we see that the wires, with a diameter of 0.25 mm, has only managed to heat its immediate surroundings while the rest of the reactor remains at room temperature. This is a consequence of the thickness of the wire. Even at this high temperature, only small amounts of radiative energy leaves the surface of the wire[57]. The distribution of heat in the area around the wire is even, producing concentric rings around each filament. These rings vary in temperature depending on the distance from the filament.



FIGURE 5.8: Convergence temperature for straight wire configuration (K)

#### 5.4.1.2 Case 2A

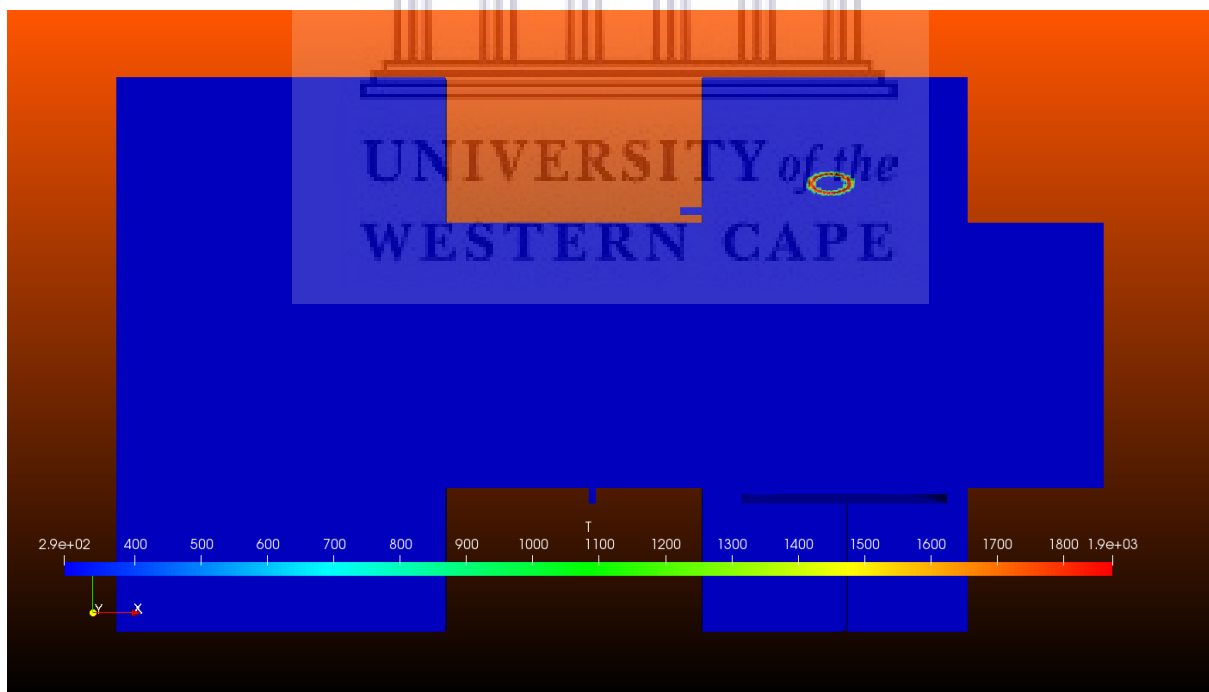


FIGURE 5.9: Initial temperature for coiled wire configuration (K)

In the second study for the Group A cases, the coiled wire configuration is considered. Figure 5.9 illustrates the initial temperature profile of the reactor for the coiled wire configuration. Again, 293.15 K is used for the internal temperature with the coil set to 1873.15 K. Only a cross

section of the coil is visible in Figure 5.9. To view the true orientation of the coil in the reactor, the reader is referred to Figure 4.3. The temperature at the point of convergence is given in Figure 5.10.

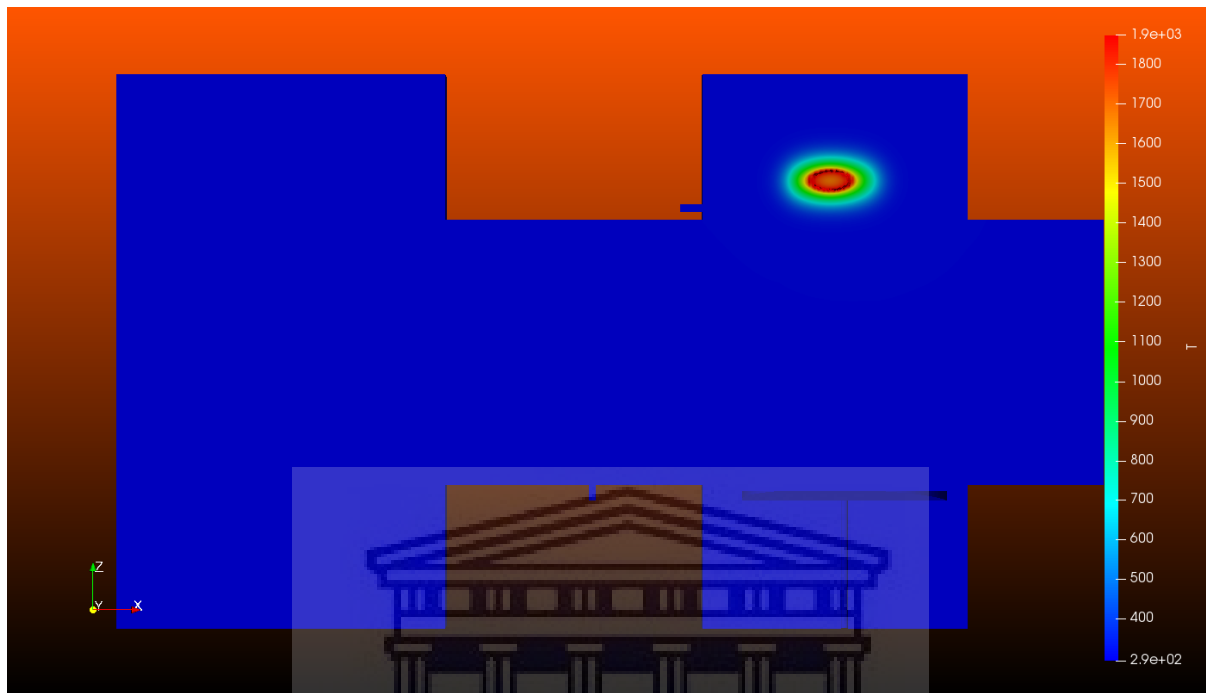


FIGURE 5.10: Convergence radiation temperature for coiled wire configuration (K)

To create the coiled filament, the wire, with a diameter of 0.25 mm, is looped around a coiling device to produce a coiled filament with a diameter of 5.19 mm. This increases the total surface area of the filament. This thus increases the area of heating at the top of the reactor. As was the case for the straight wires, the coiled wire heats only the immediate area of its surroundings, producing concentric rings that vary in temperature. The distribution of the rings are even around the surface of the coil and vary in temperature from 700 K to 1873.15 K at the surface of the coil. Again, the rest of the chamber remains unaffected.

#### 5.4.2 Heat Transfer cases

In this section, the model's temperature results for the cases with a gas flow into the reactor are presented for the two wire configurations. The cases presented here start off at the initial temperature shown in Figures 5.7 and 5.9 respectively.

##### 5.4.2.1 Case 1B

Figure 5.11 shows a large increase in the heating area surrounding the wires when compared to Figure 5.8. This increase in temperature stems from the transfer of heat from the surface

of the wires to the  $H_2$  gas emanating from the inlet. The area around the wires shows an even distribution of heating of the precursor gas. The even distribution of heating stems from the even spacing of the wires inside the wire holder which, in theory, means that the gas is decomposed uniformly across the face of the substrate leading to the growth of uniform thin films. A key observation from Figure 5.11 is that the acquired heat from the gas dissipates relatively quickly resulting in the rest of the reactor vessel remaining relatively unaffected by changes in the gas temperature. The heated area around the wires can thus be considered to be the location where it is most likely for all primary and secondary reactions to occur during the deposition.



FIGURE 5.11: Convergence heat transfer temperature for straight wire configuration (K)

This is the optimal result for this wire configuration which makes it the favoured configuration for the growth of thin films [9]. To better visualise the heat transfer from the wires to the gas, streamlines, similar to the ones used for velocity, are used to map the temperature evolution of the gas as it moves through the reactor vessel. Figure 5.12 and 5.13 shows the top- and side-view of the temperature streamlines respectively.

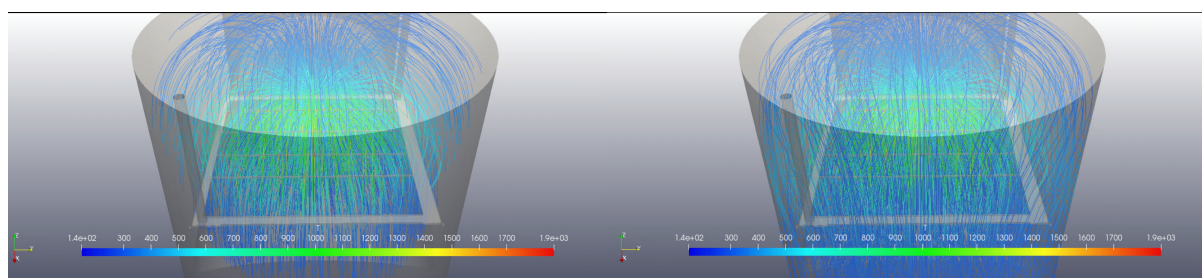


FIGURE 5.12: Top-view of temperature streamline plots for straight wire configuration (K)

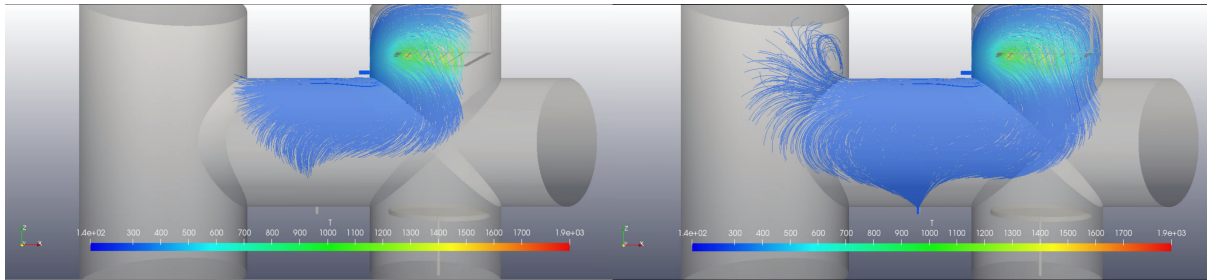


FIGURE 5.13: Side-view of temperature streamline plots for straight wire configuration (K)

The streamline plots reaffirm the results shown in Figure 5.11. They show a definite increase in the temperature of the gas around the heated wire while showing the progression of the heat transfer from the wires to the gas. Given that the streamlines show the gas increases in temperature before making direct contact with the wire, one can deduce that it is likely, assuming the temperature is high enough, a few initial gas reactions can occur in this area.

#### 5.4.2.2 Case 2B

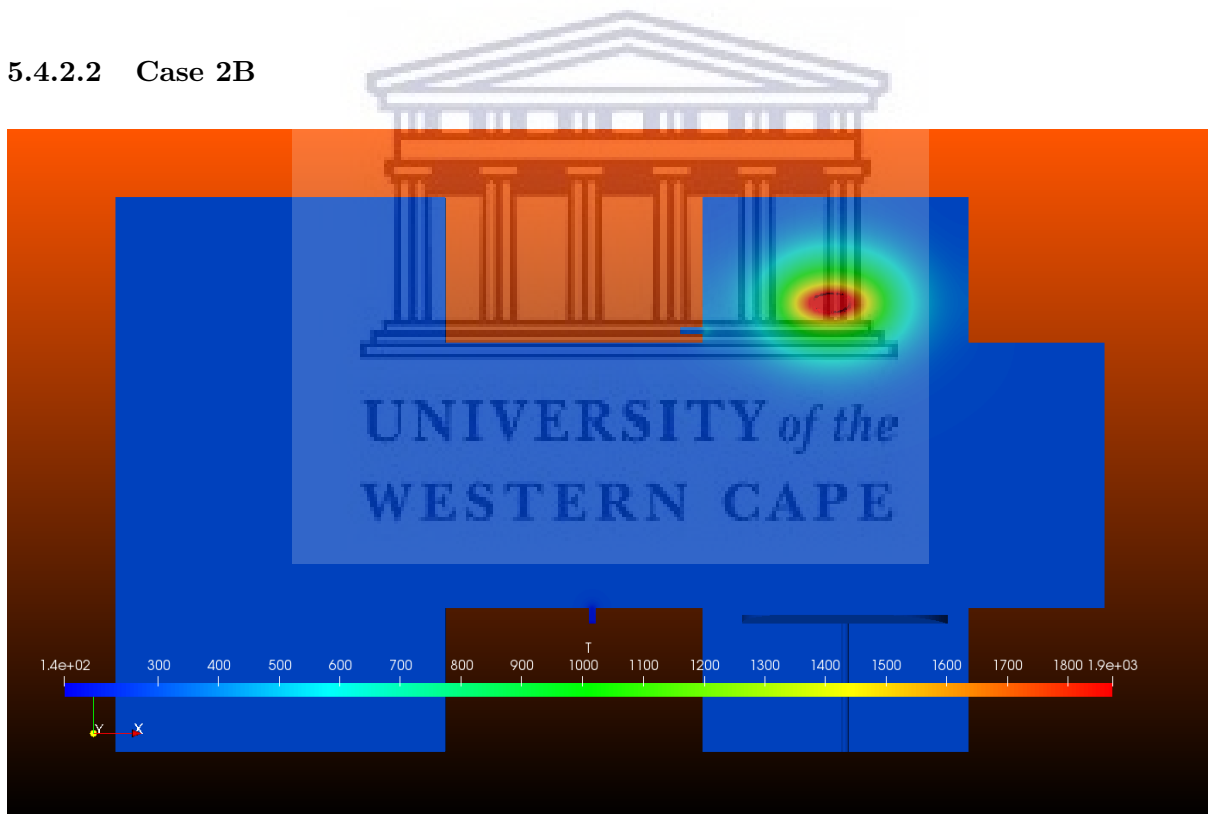


FIGURE 5.14: Convergence heat transfer temperature for coiled wire configuration (K)

Figure 5.14 shows a vastly different heat transfer profile from that shown in Figure 5.11 with the biggest difference being that the coiled wire now has a more of an effect on the overall temperature of the reactor chamber. The results for this wire configuration shows an increase in chamber temperature of approximately 100 K both at the top of the chamber, above the coil, and the area below the coil approximately 4 cm away from the coil. Another notable observation from Figure 5.14 is that the heat transfer is more concentrated for the coiled wire

configuration and is not as evenly distributed as in the case for the straight wire configuration seen in Figure 5.11. This stems from the shape and placement of the coil at the top of the reactor chamber. This concentration of heated precursor gas is why the coiled wire setup is unfavourable for growth purposes but favourable for the post deposition treatments of thin films[9]. Streamline plots are once again used to three-dimensionally map the heat transfer for this wire configuration. The streamline heat transfer plots for the top- and side-view are presented in Figures 5.15 and 5.16 respectively.

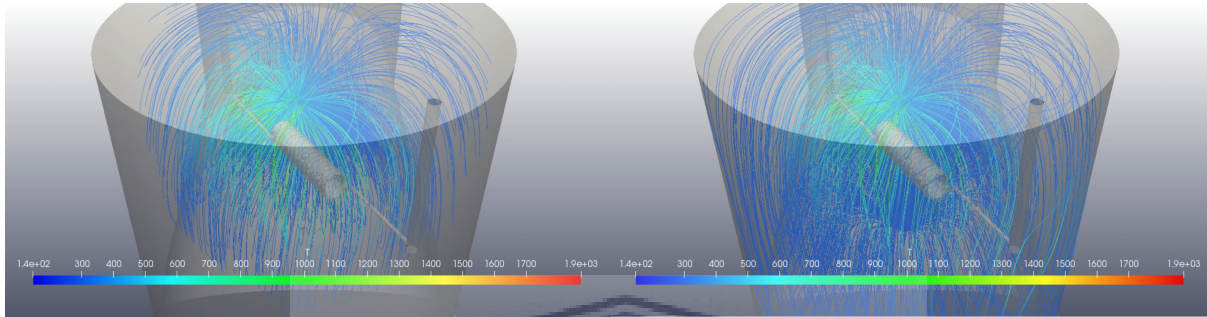


FIGURE 5.15: Top-view of temperature streamline plots for coiled wire configuration (K)



FIGURE 5.16: Side-view of temperature streamline plots for coiled wire configuration (K)

The streamline plots show the extent of the heat transfer from the wires to the gas. The results from the streamline plots support the idea that heat transfer occurs without direct contact between the gas and the wire. The distance from the coil at which the non-contact heat transfer occurs also increased for this wire configuration which leads to the increase in the overall temperature of the chamber, in the region adjacent to the coil, seen in Figure 5.14. The temperature dissipates over relatively short distances which leads to the rest of the vessel remaining at the initial temperature of 293 K (20°C).



## 5.5 Temperature Plots

Most depositions require substrate heating during the process of deposition. The temperature of the substrate ( $T_{sub}$ ) has an effect on the quality of the final film [58]. As such, controlling and being able to maintain  $T_{sub}$  during the deposition process is of notable importance. The substrate-to-filament distance ( $d_{f-s}$ ) thus plays an important role in controlling  $T_{sub}$ . The placement of the substrate inside the reactor chamber is key to maintaining the desired temperature as this allows you to minimize the filaments contribution to the overall temperature of the substrate. This can be seen in a study conducted by Lee et al.[5] in which the effects of filament temperature on the crystallographic properties of poly-Si films are considered. In their study, they varied the  $d_{f-s}$  distance, using a  $T_f = 1800^\circ\text{C}$  and  $2000^\circ\text{C}$ , to determine which value for  $d_{f-s}$  would maintain a substrate temperature of  $400^\circ\text{C}$ . A plot of their results is given in the figure below.

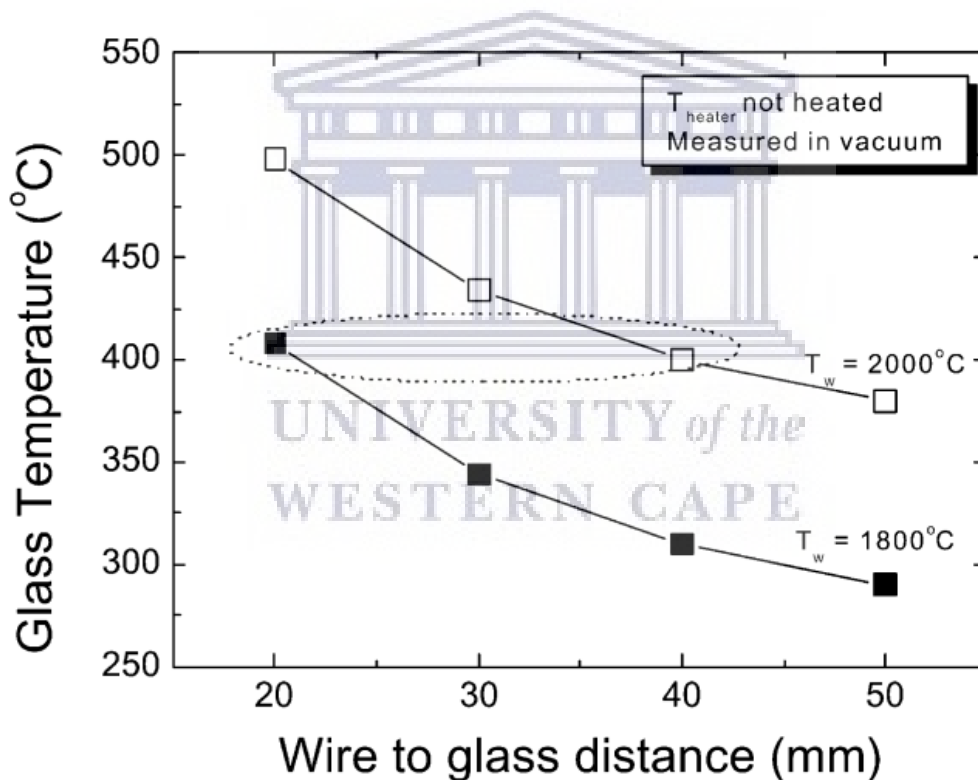


FIGURE 5.17: The variation in glass temperature as a function of  $d_{f-s}$  distance and filament temperature[5].

As seen in Figure 5.17, Lee et al. determined the optimum  $d_{f-s}$  distance to be 20 mm and 40 mm for  $T_f = 1800^\circ\text{C}$  and  $2000^\circ\text{C}$  respectively. Since limiting the amount of heat transfer from the filament to the substrates is key in maintaining the desired substrate temperature, it is important to know the exact value of the temperature inside the reactor chamber at various  $d_{f-s}$  distances in what is considered the deposition region i.e. the area between the filament and

the substrate holder. To investigate the changes in temperature in the model, the temperature, in the deposition region, is plotted as a function of  $d_{f-s}$  distance for a fixed wire temperature of  $T_f = 1600^\circ\text{C}$ . These plots are shown in Figure 5.18 where the temperature plots are presented for the simulation results obtained from the radiation (Vacuum) and heat transfer (with gas) cases.

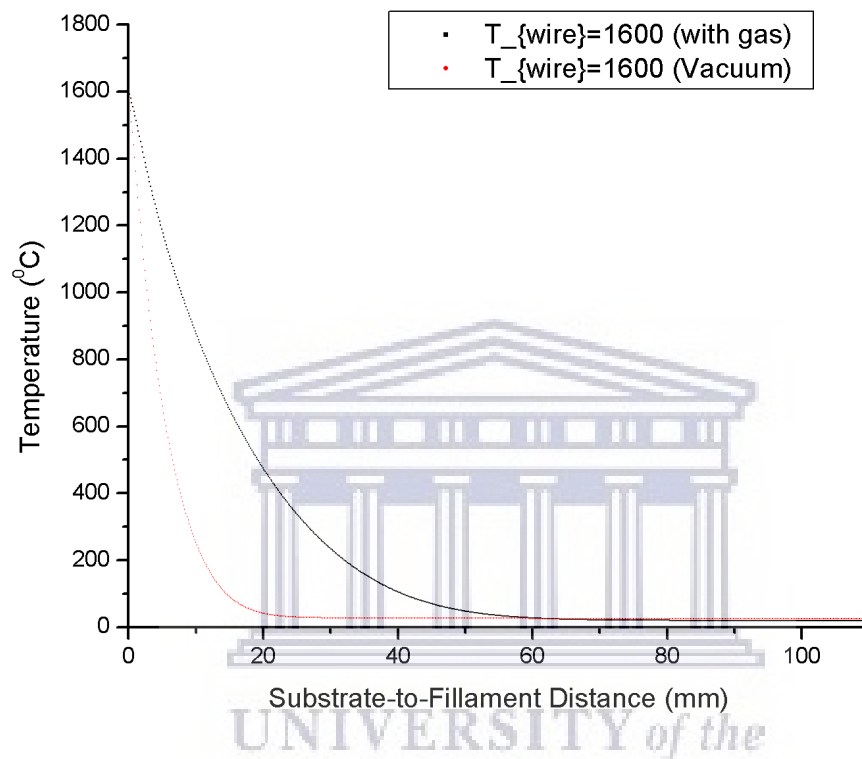


FIGURE 5.18: Substrate-to-Filament Temperature plots for  $T_f = 1600^\circ\text{C}$

Figure 5.18 shows the large difference in temperature between the cases with and without gas flow. This difference stems from the addition of conduction and convection to the modes heat transfer that comes with the flow of gas in the reactor. The temperature dissipation in the region appears to be proportional to  $\frac{1}{r^2}$ , where  $r$  is the distance from the filament. The results for the temperature distribution and dissipation can now be used, in conjunction with the mean free path and recombination length information for the precursor gas, to establish a safe deposition distance value for  $d_{f-s}$  as well as assist with maintaining control over the substrate temperature.

For the purpose of comparison, Figure 5.18 is replotted for various wire temperatures, namely:  $T_f = 1600^\circ\text{C}$ ,  $1800^\circ\text{C}$  and  $2000^\circ\text{C}$ , and compared to the temperature plots for  $T_f = 1600^\circ\text{C}$  in vacuum. This was done to try and ascertain whether there is any relation between the temperature dissipation in the chamber and the filament temperature. The results of this investigation are shown in Figure 5.19.

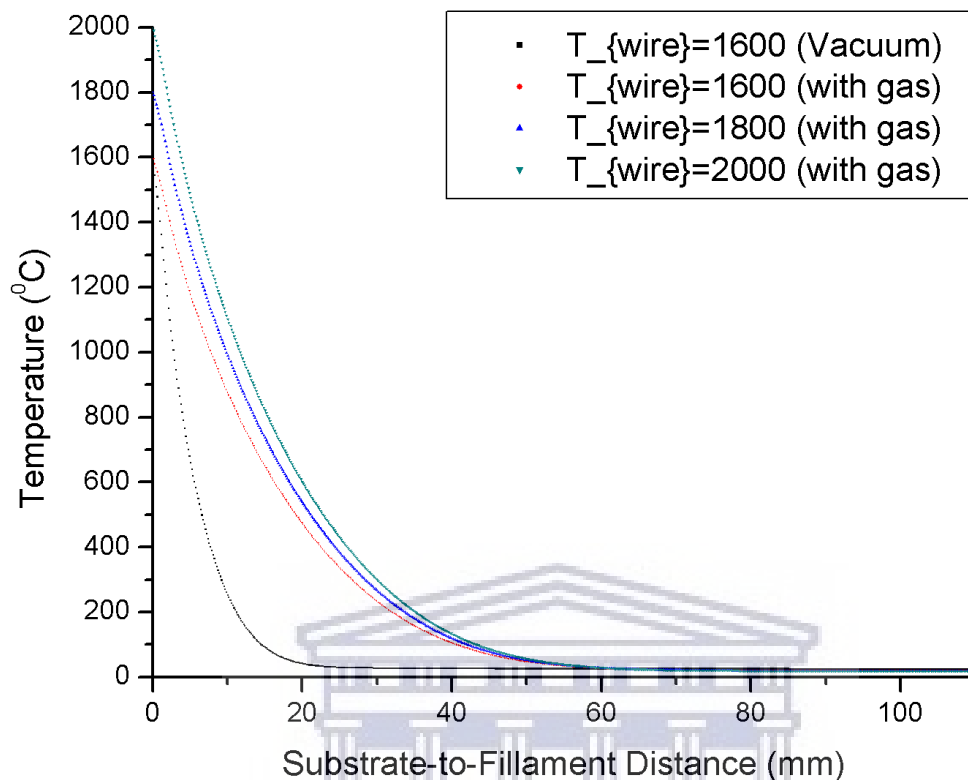


FIGURE 5.19: Substrate-to-Filament Temperature plots for various wire temperatures

Figure 5.19 shows that the temperature dissipation in the model, for different filament temperatures, follows a similar trend inside the reactor. The temperature plots, for the three cases with gas, shows that the temperature in the deposition region stabilizes approximately 40 mm from the filament regardless of the filament temperature. At this point, the three plots converge at approximately 200°C after-which they tend towards room temperature, 20°C, at 60 mm and beyond. In order to compare the results for the vacuum plots to that of Lee et al., shown in Figure 5.16, one would have to consider points much closer to the filaments in order to obtain similar substrate temperatures. Where HWCVD reactor volume used by Lee et al., is one quarter that of the CADAR HWCVD reactor chamber. This leads to the radiation from the filaments in their reactor being more concentrated resulting in higher localized temperatures. This once again speaks to the reactor geometry's effect on the deposition parameters as well as why parameter information is not transferable from one chamber to the other.

## Chapter 6

# Conclusion

In this investigation, the HWCVD reactor of the CADAR deposition system was studied in the framework of the open-source CFD package, OpenFOAM. The objective of this study was to ascertain the deposition conditions inside the reactor during a standard deposition process. Coupled with this was the goal of creating a working model that can be used, by experimentalists wanting to use the reactor to determine the optimal deposition parameters for their film requirements. To test the functionality of the model, standard deposition parameters for the production of atomic hydrogen from molecular hydrogen were used in this investigation.

The complete set conservation equations for mass, momentum and energy were taken into account and solved numerically using solvers deemed to appropriate for the purposes of the study. In modelling the HWCVD reactor, the internal gas remaining in the chamber at the base vacuum pressure was modelled as air which includes all the gas phase properties that come with this selection. In the transport phase, the hydrogen gas is modelled using Sutherland's Law, in conjunction with the gas phase properties of hydrogen, to calculate the values of the transport properties during the simulation process.

The material properties of each layer of the reactor for both the boundary layers and gas transport layers was selected from values presented in literature. These can be found in chapter 4 under the section on the boundary conditions.

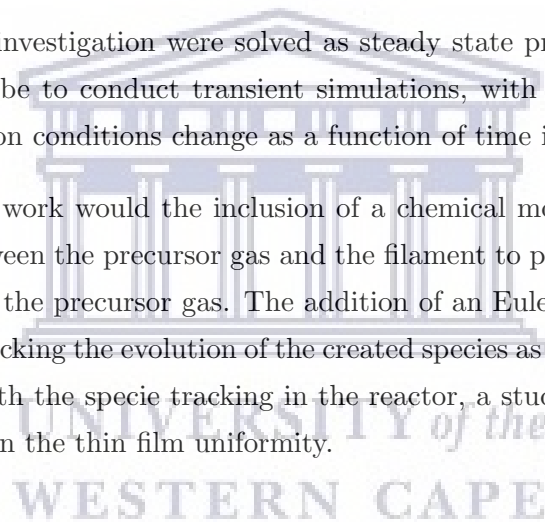
In chapter 5, the model's results for the two sets of wire configurations are presented and discussed. The results for the two wire configuration, coiled and straight, showed the characteristics which make each a favoured choice depending on the type of deposition in question as outlined in the work by Sali et al.[9]. In their work, Sali et al. concluded that the straight wire configuration would be favourable for growth depositions while the coiled configuration would be better suited for the post deposition treatment of films. The former stemming from the even distribution of heat around the wires and the latter providing a more concentrated heat profile around

the coil. This conclusion is evident in the model's temperature profile results, Figures 5.11 and 5.14 for the straight and coiled wire configuration respectively, in which the distribution of heat around the wires is clearly shown. In this investigation, a means of determining an optimal distance,  $d_{f-s}$ , by means of the gas temperature at different points in the deposition region was also explored. The results of this investigation are shown in Figures 5.18 and 5.19 respectively. From these plots, we were able to establish a trend for the gas temperature dissipation which can be used along with the recombination length information for the gas to determine the  $d_{f-s}$  distance. Similar trends were reported by Lee et al. [5], however, a direct comparison of their results to the results obtained in this work is not possible due to the differences in reactor geometry and size.

## 6.1 Future Work

Since all the cases in this investigation were solved as steady state problems, the natural evolution of this work would be to conduct transient simulations, with the same parameters, to ascertain how the deposition conditions change as a function of time inside the reactor.

A notable addition to the work would be the inclusion of a chemical model that will resolve the reactions taking place between the precursor gas and the filament to predict the species created from the decomposition of the precursor gas. The addition of an Eulerian model to this would then provide a means of tracking the evolution of the created species as they migrate through the reactor vessel. Coupled with the specie tracking in the reactor, a study can also be conducted on the effects of pressure on the thin film uniformity.





Create polyMesh for time = 0

Time = 0

Mesh stats

```

points:          16886388
faces:           50328392
internal faces:  49856973
cells:           16721137
faces per cell:  5.99154
boundary patches: 5
point zones:    0
face zones:     0
cell zones:     0

```

Overall number of cells of each type:

```

hexahedra:      16579729
prisms:         140057
wedges:         0
pyramids:       0
tet wedges:     0
tetrahedra:    0
polyhedra:      1351

```

Breakdown of polyhedra by number of faces:

faces	number of cells
4	49
5	1302

Checking topology...

```

Boundary definition OK.
Cell to face addressing OK.
Point usage OK.
Upper triangular ordering OK.
Face vertices OK.
Number of regions: 1 (OK).

```

Checking patch topology for multiply connected surfaces...

Patch	Faces	Points	Surface topology
sides	460081	460523	multiply connected (shared edge)

```

wires      1448      1472 ok (non-closed singly connected)
inlet      12        16  ok (non-closed singly connected)
stage     9868     10094 ok (non-closed singly connected)
outlet     10        16  ok (non-closed singly connected)

```

```
<<Writing 4 conflicting points to set nonManifoldPoints
```

```
Checking geometry...
```

```
Overall domain bounding box (-0.398288 -0.0899753 0) (0.1417 0.0899753 0.303)
```

```
Mesh has 3 geometric (non-empty/wedge) directions (1 1 1)
```

```
Mesh has 3 solution (non-empty) directions (1 1 1)
```

```
Boundary openness (1.85899e-15 1.13688e-15 4.19692e-16) OK.
```

```
Max cell openness = 4.30572e-16 OK.
```

```
Max aspect ratio = 3.80743 OK.
```

```
Minimum face area = 1.9063e-07. Maximum face area = 2.40921e-06. Face area magnitudes OK
```

```
Min volume = 2.23495e-10. Max volume = 2.71017e-09. Total volume = 0.0167006. Cell volume OK
```

```
Mesh non-orthogonality Max: 46.9139 average: 1.13111
```

```
Non-orthogonality check OK.
```

```
Face pyramids OK.
```

```
Max skewness = 2.13338 OK.
```

```
Coupled point location match (average 0) OK.
```

```
Mesh OK.
```

```
End
```



UNIVERSITY of the  
WESTERN CAPE

## A.2 checkMesh output for Coiled wire mesh

```

/*-----*\
| ===== |
| \\ / F i e l d | OpenFOAM: The Open Source CFD Toolbox |
| \\ / O p e r a t i o n | Version: 5.x |
| \\ / A n d | Web: www.OpenFOAM.org |
| \\ / M a n i p u l a t i o n | |
\*-----*/

Build : 5.x-197d9d3bf20a
Exec : checkMesh
Date : Jul 25 2019

```



```

Time    : 10:29:28
Host    : "lionel"
PID     : 4391
I/O     : uncollated
Case    : /home/lionel/OpenFOAM/lionel-5.0/run/Flow2
nProcs  : 1
sigFpe  : Enabling floating point exception trapping (FOAM_SIGFPE).
fileModificationChecking : Monitoring run-time modified files using timeStampMaster (fileMod
allowSystemOperations : Allowing user-supplied system call operations
    
```

// \* //

Create time

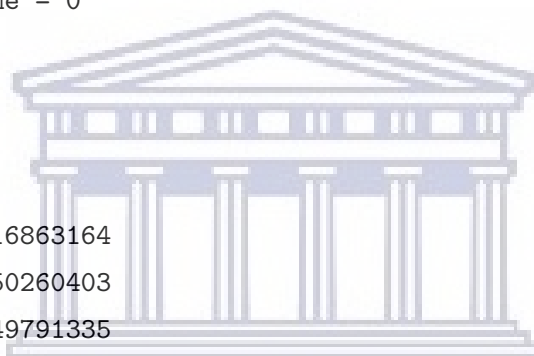
Create polyMesh for time = 0

Time = 0

Mesh stats

```

points:          16863164
faces:           50260403
internal faces:  49791335
cells:           16699201
faces per cell:  5.99141
boundary patches: 5
point zones:    0
face zones:     0
cell zones:     0
    
```



UNIVERSITY of the  
WESTERN CAPE

Overall number of cells of each type:

```

hexahedra:      16555936
prisms:         141395
wedges:         0
pyramids:      0
tet wedges:    3
tetrahedra:    1
polyhedra:     1866
    
```

Breakdown of polyhedra by number of faces:

faces	number of cells
4	199

5 1667

Checking topology...

Boundary definition OK.  
 Cell to face addressing OK.  
 Point usage OK.  
 Upper triangular ordering OK.  
 Face vertices OK.  
 Number of regions: 1 (OK).

Checking patch topology for multiply connected surfaces...

Patch	Faces	Points	Surface topology
heater	9850	10075	ok (non-closed singly connected)
sides	455110	455544	ok (non-closed singly connected)
wires	4082	4485	ok (non-closed singly connected)
outlet	14	17	ok (non-closed singly connected)
inlet	12	13	ok (non-closed singly connected)

Checking geometry...

Overall domain bounding box (-0.3983 -0.0899753 0) (0.1417 0.0899753 0.303)  
 Mesh has 3 geometric (non-empty/wedge) directions (1 1 1)  
 Mesh has 3 solution (non-empty) directions (1 1 1)  
 Boundary openness (4.87703e-15 -1.88983e-15 7.65912e-16) OK.  
 Max cell openness = 4.23484e-16 OK.  
 Max aspect ratio = 4.72467 OK.  
 Minimum face area = 2.64727e-08. Maximum face area = 3.46982e-06. Face area magnitudes  
 Min volume = 1.47405e-10. Max volume = 2.22271e-09. Total volume = 0.0167012. Cell vol  
 Mesh non-orthogonality Max: 61.5042 average: 1.12385  
 Non-orthogonality check OK.  
 Face pyramids OK.  
 Max skewness = 3.58223 OK.  
 Coupled point location match (average 0) OK.

Mesh OK.

End


## Appendix B

# buoyantSimpleFoam dictionaries

This appendix contains a selection of the buoyantSimpleFoam dictionaries. They are listed according to their parent folder.

### B.1 0 folder

#### B.1.1 U



```
/*-----*- C++ -*-----*\
| =====|
| \\ / F i e l d | OpenFOAM: The Open Source CFD Toolbox |
| \\ / O p e r a t i o n | Version: 5 |
| \\ / A n d | Web: www.OpenFOAM.org |
| \\ / M a n i p u l a t i o n | |
\*-----*-
FoamFile
{
    version      2.0;
    format       ascii;
    class        volVectorField;
    object       U;
}
// * * * * *

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

```
internalField    uniform (0 0 0);
```

```
boundaryField
```

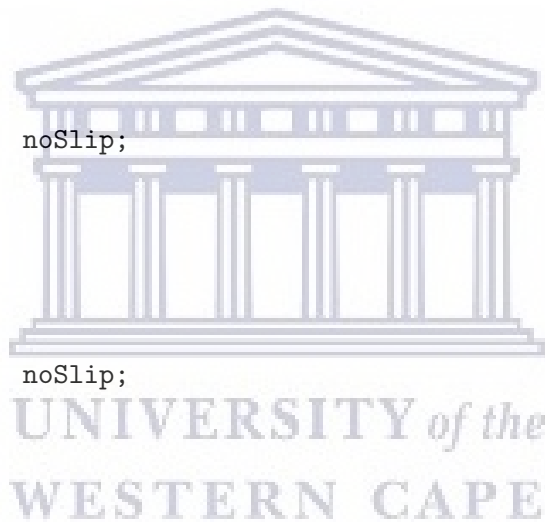
```
{
  inlet
  {
    type          noSlip;
  }

  heater
  {
    type          noSlip;
  }

  sides
  {
    type          noSlip;
  }

  wires
  {
    type          noSlip;
  }

  outlet
  {
    type          noSlip;
  }
}
```



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

### B.1.2 p

```
/*-----* C++ *-----*\
| ===== |
| \\ / F i e l d | OpenFOAM: The Open Source CFD Toolbox |
| \\ / O p e r a t i o n | Version: 5 |
```

```

|  \ \ /   A nd           | Web:      www.OpenFOAM.org           |
|  \ \ /   M anipulation |                                     |
\*-----*
FoamFile
{
    version      2.0;
    format       ascii;
    class        volScalarField;
    object       p;
}
// * * * * *

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

internalField   uniform 6e-4;

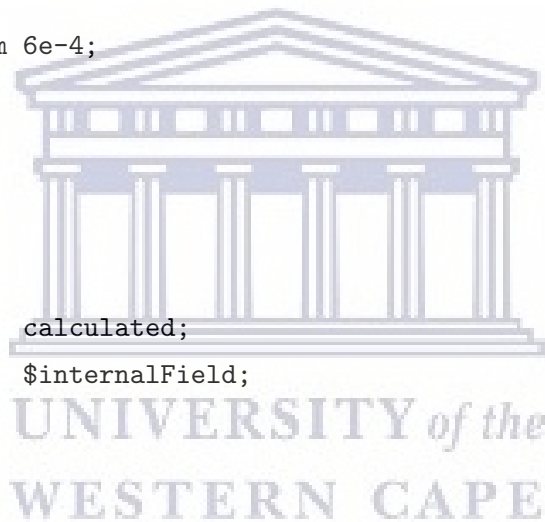
boundaryField
{
    heater
    {
        type      calculated;
        value     $internalField;
    }

    sides
    {
        type      calculated;
        value     $internalField;
    }

    inlet
    {
        type      calculated;
        value     $internalField;
    }

    wires
    {
        type      calculated;

```



```

        value          $internalField;
    }
    outlet
    {
        type           calculated;
        value          $internalField;
    }
}

// ***** //

```

### B.1.3 T

```

/*-----*- C++ -*-----*\
|=====|
| \ \ / F i e l d | OpenFOAM: The Open Source CFD Toolbox |
| \ \ / O p e r a t i o n | Version: 5 |
| \ \ / A n d | Web: www.OpenFOAM.org |
| \ \ / M a n i p u l a t i o n | |
\*-----*/
FoamFile
{
    version      2.0;
    format       ascii;
    class        volScalarField;
    object       T;
}
// ***** //

```

```
dimensions      [0 0 0 1 0 0 0];
```

```
internalField   uniform 293;
```

```
boundaryField
{
    heater
    {

```

```

        type          fixedValue;
        value         uniform 293;
    }

    inlet
    {
        type          zeroGradient;
    }

    sides
    {
        type          zeroGradient;
    }

    wires
    {
        type          fixedValue;
        value         uniform 1873;
    }

    outlet
    {
        type          zeroGradient;
    }
}

// ***** //

```

#### B.1.4 G

```

/*-----*- C++ -*-----*\
| ===== |
| \\ / F i e l d | OpenFOAM: The Open Source CFD Toolbox |
| \\ / O p e r a t i o n | Version: 5 |
| \\ / A n d | Web: www.OpenFOAM.org |
| \\ / M a n i p u l a t i o n |
\*-----*/

```

```
FoamFile
{
    version      2.0;
    format       ascii;
    class        volScalarField;
    object       G;
}
// * * * * *


dimensions      [1 0 -3 0 0 0 0];

internalField   uniform 0;

boundaryField
{
    inlet
    {
        type          MarshakRadiation;
        emissivityMode lookup;
        emissivity     uniform 0.85;
        value          uniform 0;
    }

    heater
    {
        type          MarshakRadiation;
        emissivityMode lookup;
        emissivity     uniform 0.85;
        value          uniform 0;
    }

    sides
    {
        type          MarshakRadiation;
        emissivityMode lookup;
        emissivity     uniform 0.85;
        value          uniform 0;
    }
}
```

The logo of the University of the Western Cape, featuring a classical building with columns and a pediment, with the text "UNIVERSITY of the WESTERN CAPE" below it.



```

wires
{
    type            MarshakRadiation;
    emissivityMode  lookup;
    emissivity      uniform 0.35;
    value           uniform 0;
}
outlet
{
    type            MarshakRadiation;
    emissivityMode  lookup;
    emissivity      uniform 0.85;
    value           uniform 0;
}
}

// ***** //

```

## B.2 constant folder

### B.2.1 g

```

/*-----* C++ *-----*\
| ===== | |
| \\ / F i e l d | OpenFOAM: The Open Source CFD Toolbox |
| \\ / O p e r a t i o n | Version: 5 |
| \\ / A n d | Web: www.OpenFOAM.org |
| \\ / M a n i p u l a t i o n | |
\*-----*/

FoamFile
{
    version      2.0;
    format       ascii;
    class        uniformDimensionedVectorField;
    location     "constant";
    object       g;
}

```



```

    absorptivity    absorptivity    [0 -1 0 0 0 0 0] 0.5;
    emissivity      emissivity      [0 -1 0 0 0 0 0] 0.5;
    E               E               [1 -1 -3 0 0 0 0] 0;
}

```

```
scatterModel    none;
```

```
sootModel      none;
```

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

### B.2.3 thermophysicalProperties

```

/*-----*- C++ -*-----*\
|=====|
| \ \ / Field | OpenFOAM: The Open Source CFD Toolbox |
| \ \ / Operation | Version: 5 |
| \ \ / And | Web: www.OpenFOAM.org |
| \ \ / Manipulation |
\*-----*\

```

```

FoamFile
{
    version    2.0;
    format     ascii;
    class      dictionary;
    location   "constant";
    object     thermophysicalProperties;
}

```

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

```

thermoType
{
    type          heRhoThermo;
    mixture       pureMixture;
    transport     const;
    thermo        hConst;
    equationOfState perfectGas;
}

```

```

    specie      specie;
    energy      sensibleEnthalpy;
}

```

```
pRef          10000;
```

```
mixture
```

```

{
    specie
    {
        molWeight      28.97;
    }
    thermodynamics
    {
        Cp              1000;
        Hf              0;
    }
    transport
    {
        mu              1.8e-05;
        Pr              0.7;
    }
}

```



UNIVERSITY of the  
WESTERN CAPE

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

## B.2.4 turbulenceProperties

```

/*-----* C++ -*-----*\
| ===== |
| \\ / Field | OpenFOAM: The Open Source CFD Toolbox |
| \\ / Operation | Version: 5 |
| \\ / And | Web: www.OpenFOAM.org |
| \\ / Manipulation |
\*-----*/
FoamFile
{

```

```

    version      2.0;
    format       ascii;
    class        dictionary;
    location     "constant";
    object       RASProperties;
}
// * * * * * //

```

```

simulationType laminar;

// ***** //

```

### B.3 system folder

#### B.3.1 controlDict

```

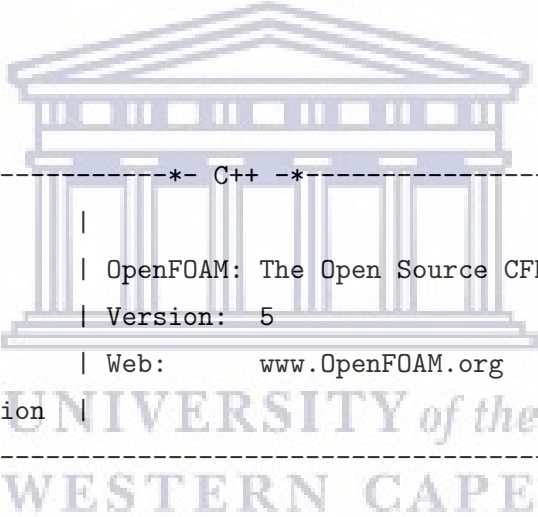
/*----- C++ -----*\
| ===== |
|  \  /   F ield | OpenFOAM: The Open Source CFD Toolbox |
|  \  /   O peration | Version: 5 |
|  \  /   A nd | Web: www.OpenFOAM.org |
|  \ \ /   M anipulation |
\*-----*/
FoamFile
{
    version      2.0;
    format       ascii;
    class        dictionary;
    location     "system";
    object       controlDict;
}
// * * * * * //

application      buoyantSimpleFoam;

startFrom        latestTime;

startTime        0;

```



```

stopAt      endTime;

endTime     1000;

deltaT      1;

writeControl  timeStep;

writeInterval  100;

purgeWrite  0;

writeFormat  ascii;

writePrecision  6;

writeCompression off;

timeFormat   general;

timePrecision  6;

runTimeModifiable true;

functions
{
#includeFunc residuals
}

```

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

### B.3.2 fvSchemes

```

/*-----* C++ *-----\
| ===== | |

```

```

| \ \      / F i e l d          | OpenFOAM: The Open Source CFD Toolbox          |
| \ \      / O p e r a t i o n    | Version: 5                                |
| \ \      / A n d                 | Web:      www.OpenFOAM.org                |
| \ \ \ /   M a n i p u l a t i o n |                                         |
\*-----*//
FoamFile
{
    version      2.0;
    format        ascii;
    class         dictionary;
    location      "system";
    object        fvSchemes;
}
// * * * * *

ddtSchemes
{
    default      steadyState;
}

gradSchemes
{
    default      Gauss linear;
}

divSchemes
{
    default      none;
    div(phi,U)   bounded Gauss upwind;
    div(phi,K)   bounded Gauss upwind;
    div(phi,h)   bounded Gauss upwind;
    div(phi,k)   bounded Gauss upwind;
    div(phi,epsilon) bounded Gauss upwind;
    div(((rho*nuEff)*dev2(T(grad(U)))) Gauss linear;
}

laplacianSchemes
{
    default      Gauss linear corrected;
}

```



```

}

interpolationSchemes
{
    default          linear;
}

snGradSchemes
{
    default          corrected;
}

// ***** //

```

### B.3.3 fvSolutions

```

/*-----*- C++ -*-----*\
| ===== |
| \\      / F i e l d | OpenFOAM: The Open Source CFD Toolbox |
| \\      / O p e r a t i o n | Version: 5 |
| \\      / A n d | Web: www.OpenFOAM.org |
|  \\    / M a n i p u l a t i o n |
\*-----*-----*/
FoamFile
{
    version      2.0;
    format       ascii;
    class        dictionary;
    location     "system";
    object       fvSolution;
}

// ***** //

solvers
{
    p_rgh
    {

```



```

        solver          PCG;
        preconditioner  DIC;
        tolerance       1e-06;
        relTol          0.01;
    }

    "(U|h|k|epsilon)"
    {
        solver          PBiCGStab;
        preconditioner  DILU;
        tolerance       1e-05;
        relTol          0.1;
    }

    G
    {
        $p_rgh;
        tolerance       1e-05;
        relTol          0.1;
    }
}

SIMPLE
{
    nNonOrthogonalCorrectors 0;
    pRefCell          0;
    pRefValue         0;

    residualControl
    {
        p_rgh          1e-2;
        U              1e-3;
        h              1e-3;
        G              1e-3;

        // possibly check turbulence fields
        "(k|epsilon|omega)" 1e-3;
    }
}

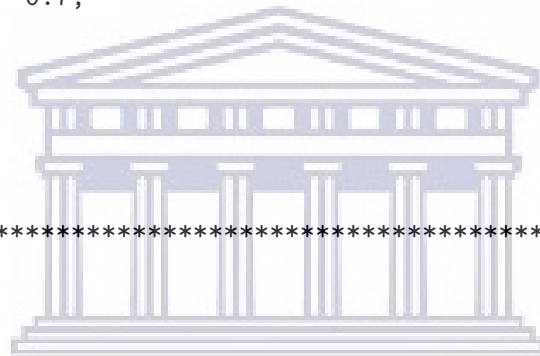
```



UNIVERSITY of the  
WESTERN CAPE

```
relaxationFactors
{
    fields
    {
        rho          1.0;
        p_rgh        0.7;
    }
    equations
    {
        U            0.2;
        h            0.2;
        "(k|epsilon|R)" 0.5;
        G            0.7;
    }
}

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



UNIVERSITY *of the*  
WESTERN CAPE


## Appendix C

# rhoSimpleFoam dictionaries

This appendix contains a selection of the rhoSimpleFoam dictionaries. They are listed according to their parent folder.

### C.1 0 folder

#### C.1.1 U



```
/*-----*- C++ -*-----*\
| =====|
| \\      / F i e l d      | OpenFOAM: The Open Source CFD Toolbox |
| \\      / O p e r a t i o n | Version: 5 |
| \\      / A n d          | Web:      www.OpenFOAM.org |
|  \\    / M a n i p u l a t i o n |
\*-----*- C++ -*-----*/

FoamFile
{
    version      2.0;
    format       ascii;
    class        volVectorField;
    object       U;
}

// *****

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

```

internalField    uniform (0 0 0);

boundaryField
{
    stage
    {
        type      noSlip;
    }

    inlet
    {
type      flowRateInletVelocity;
        massFlowRate    constant 8.988e-8;
rhoInlet 0.09;
        //value          uniform (1 0 0);
    }

    sides
    {
        type      noSlip;
    }

    wires
    {
        type      noSlip;
    }

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

```



UNIVERSITY of the  
WESTERN CAPE

## C.1.2 p

```

/*-----*- C++ -*-----*\
| ===== |
| \\ / F i e l d | OpenFOAM: The Open Source CFD Toolbox |
| \\ / O p e r a t i o n | Version: 5 |
| \\ / A n d | Web: www.OpenFOAM.org |
| \\ / M a n i p u l a t i o n | |
\*-----*-*/

FoamFile
{
    version      2.0;
    format       ascii;
    class        volScalarField;
    object       p;
}
// *****

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

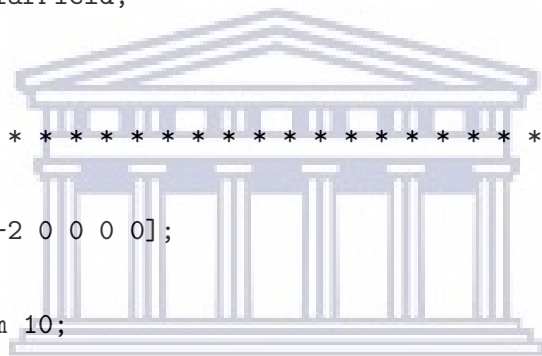
internalField   uniform 10;

boundaryField
{
    stage
    {
        type          zeroGradient;
    }

    inlet
    {
        type          zeroGradient;
        //type        mixed;
        refValue       uniform 10;
        refGradient    uniform 0;
        valueFraction  uniform 0.3;
    }

    outlet
    {

```



UNIVERSITY of the  
WESTERN CAPE

```

        type          fixedValue;
        value         uniform 10;
    }

    sides
    {
        type          zeroGradient;
    }

    wires
    {
        type          zeroGradient;
    }
}

// ***** //

C.1.3 T

/*-----*- C++ -*-----*\
| ===== |
| \\      / F i e l d | OpenFOAM: The Open Source CFD Toolbox |
| \\      / O p e r a t i o n | Version: 5 |
|  \\    / A n d | Web: www.OpenFOAM.org |
|   \\  / M a n i p u l a t i o n | |
\*-----*-/

FoamFile
{
    version      2.0;
    format       ascii;
    class        volScalarField;
    object       T;
}

// ***** //

dimensions      [0 0 0 1 0 0 0];

internalField   uniform 293;

```

```
boundaryField
{
    stage
    {
        type            zeroGradient;
    }

    inlet
    {
        type            fixedValue;
value    uniform 293;

    }
    outlet
    {
        type            inletOutlet;
        //type          zeroGradient;
        value           uniform 293;
        inletValue      uniform 293;
    }

    sides
    {
        type            zeroGradient;
    }

    wires
    {
        type            fixedValue;
        value           uniform 1873;
    }
}

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



## C.1.4 k

```

/*-----*- C++ -*-----*\
| ===== |
| \\ / F i e l d | OpenFOAM: The Open Source CFD Toolbox |
| \\ / O p e r a t i o n | Version: 5 |
| \\ / A n d | Web: www.OpenFOAM.org |
| \\ / M a n i p u l a t i o n | |
\*-----*/

FoamFile
{
    version      2.0;
    format       ascii;
    class        volScalarField;
    location     "0";
    object       k;
}
// *****

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

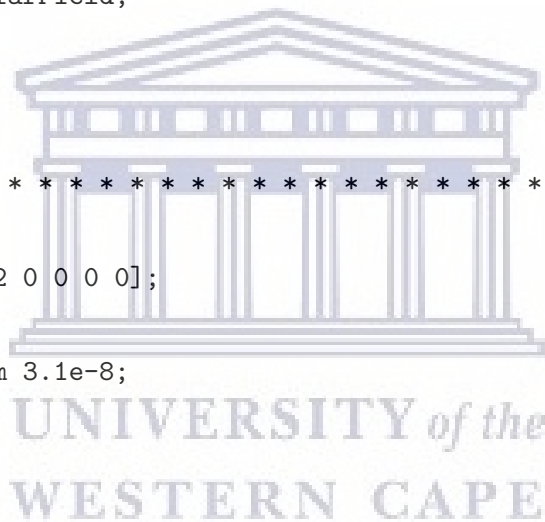
internalField   uniform 3.1e-8;

boundaryField
{
    stage
    {
        type      kqRWallFunction;
        value     $internalField;
    }

    inlet
    {
        type      inletOutlet;
        inletValue $internalField;
        value     $internalField;
    }

    outlet

```





```

    {
        type            inletOutlet;
        inletValue      $internalField;
        value            $internalField;
    }

    sides
    {
        type            kqRWallFunction;
        value            $internalField;
    }

    wires
    {
        type            kqRWallFunction;
        value            $internalField;
    }
}

// ***** //

```

### C.1.5 nut

```

/*-----* C++ *-----*\
|=====|
| \ \ / F i e l d | OpenFOAM: The Open Source CFD Toolbox |
| \ \ / O p e r a t i o n | Version: 5 |
| \ \ / A n d | Web: www.OpenFOAM.org |
| \ \ / M a n i p u l a t i o n | |
\*-----*/

FoamFile
{
    version      2.0;
    format       ascii;
    class        volScalarField;
    location     "0";
    object       nut;
}

```



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

### C.1.6 epsilon

```
/*-----* C++ -*-----*\
| ===== |
| \\ / Field | OpenFOAM: The Open Source CFD Toolbox |
| \\ / Operation | Version: 5 |
| \\ / And | Web: www.OpenFOAM.org |
| \\ / Manipulation |
\*-----*/
```

```
FoamFile
```

```
{
```

```
    version    2.0;
```

```
    format     ascii;
```

```
    class      volScalarField;
```

```
    location   "0";
```

```
    object     epsilon;
```

```
}
```

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

```
dimensions    [0 2 -3 0 0 0 0];
```

```
internalField uniform 5.9e-9;
```

```
boundaryField
```

```
{
```

```
    stage
```

```
    {
```

```
        type          epsilonWallFunction;
```

```
        value         $internalField;
```

```
    }
```

```
    inlet
```

```
    {
```

```
        type          inletOutlet;
```



UNIVERSITY of the  
WESTERN CAPE

```

    inletValue $internalField;
    value      $internalField;

}
outlet
{
    type      inletOutlet;
    inletValue $internalField;
    value      $internalField;
}

sides
{
    type      epsilonWallFunction;
    value      $internalField;
}

wires
{
    type      epsilonWallFunction;
    value      $internalField;
}
}

// ***** //

```

## C.2 constant folder

### C.2.1 thermophysicalProperties

```

/*-----*- C++ -*-----*\
| ===== |
| \\ / F i e l d | OpenFOAM: The Open Source CFD Toolbox |
| \\ / O p e r a t i o n | Version: 5 |
| \\ / A n d | Web: www.OpenFOAM.org |
| \\ / M a n i p u l a t i o n | |
\*-----*/

```

```

FoamFile
{
    version      2.0;
    format       ascii;
    class        dictionary;
    location     "constant";
    object       thermophysicalProperties;
}
// * * * * *

thermoType
{
    type          hePsiThermo;
    mixture       pureMixture;
    transport     sutherland;
    thermo        hConst;
    equationOfState perfectGas;
    specie        specie;
    energy        sensibleInternalEnergy;
}

mixture
{
    specie
    {
        molWeight  2;
    }
    thermodynamics
    {
        Cp          14130;
        Hf          0.0587e+3;
    }
    transport
    {
        As          6.89e-07;
        Ts          96.67;
    }
}

```



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

## C.2.2 turbulenceProperties

```
/*-----* C++ -*-----*\
| ===== |
| \\ / Field | OpenFOAM: The Open Source CFD Toolbox |
| \\ / Operation | Version: 5 |
| \\ / And | Web: www.OpenFOAM.org |
| \\ / Manipulation |
\*-----*/
```

```
FoamFile
```

```
{
    version      2.0;
    format       ascii;
    class        dictionary;
    location     "constant";
    object       turbulenceProperties;
}
```

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

```
simulationType RAS;
```

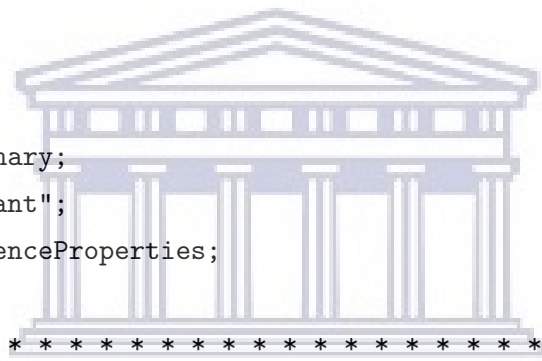
```
RAS
```

```
{
    RASModel      kEpsilon;

    turbulence     on;

    printCoeffs   on;
}
```

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



UNIVERSITY of the  
WESTERN CAPE

## C.3 system folder

### C.3.1 controlDict

```

/*-----* C++ *-----*\
| ===== | |
| \\ / Field | OpenFOAM: The Open Source CFD Toolbox |
| \\ / Operation | Version: 5 |
| \\ / And | Web: www.OpenFOAM.org |
| \\ / Manipulation | |
\*-----*/

FoamFile
{
    version 2.0;
    format ascii;
    class dictionary;
    location "system";
    object controlDict;
}
// *****

application rhoSimpleFoam;

startFrom startTime;

startTime 0;

stopAt endTime;

endTime 1000;

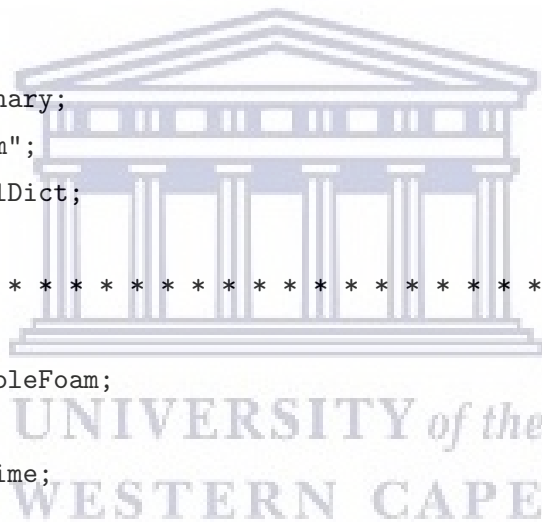
deltaT 1;

writeControl timeStep;

writeInterval 100;

purgeWrite 0;

```



```

writeFormat      ascii;

writePrecision   6;

writeCompression on;

timeFormat       general;

timePrecision    6;

graphFormat      raw;

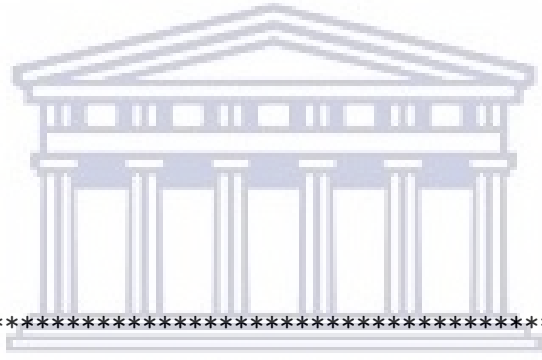
runTimeModifiable true;

functions
{
#includeFunc residuals
}

// ***** //

```

### C.3.2 fvSchemes



UNIVERSITY of the  
WESTERN CAPE

```

/*-----* C++ *-----*\
| ===== | |
| \\ / Field | OpenFOAM: The Open Source CFD Toolbox |
| \\ / Operation | Version: 5 |
| \\ / And | Web: www.OpenFOAM.org |
| \\ / Manipulation | |
\*-----*-----*/

FoamFile
{
    version      2.0;
    format       ascii;
    class        dictionary;
    location     "system";
    object       fvSchemes;
}

```





```
{
    default          corrected;
}
```

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

### C.3.3 fvSolutions

```
/*-----* C++ *-----*\
| ===== | |
| \\ / F i e l d | OpenFOAM: The Open Source CFD Toolbox |
| \\ / O p e r a t i o n | Version: 5 |
| \\ / A n d | Web: www.OpenFOAM.org |
| \\ / M a n i p u l a t i o n | |
\*-----*/
FoamFile
{
    version      2.0;
    format       ascii;
    class        dictionary;
    location     "system";
    object       fvSolution;
}
// ***** //
```

```
solvers
{
    p
    {
        solver      GAMG;
        tolerance   1e-08;
        relTol      0.1;
        smoother    GaussSeidel;
        nCellsInCoarsestLevel 20;
    }

    "(U|e|k|epsilon)"
```

```
{
    solver          GAMG;
    tolerance       1e-08;
    relTol          0.1;
    smoother        GaussSeidel;
    nCellsInCoarsestLevel 20;
}
}
```

## SIMPLE

```
{
    nNonOrthogonalCorrectors 0;
    pMinFactor          0.1;
    pMaxFactor          2;
    transonic           no;
    consistent           yes;
```

## residualControl

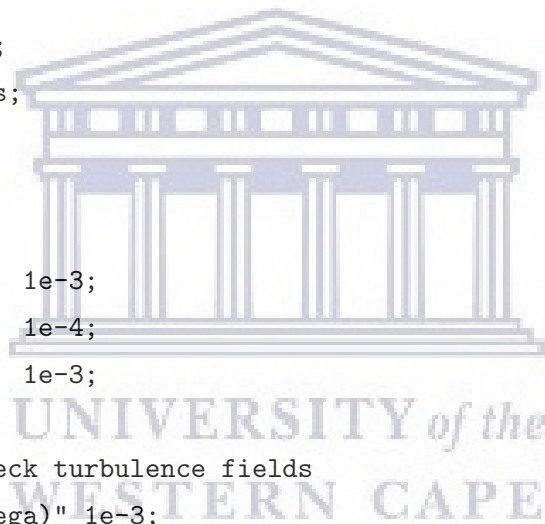
```
{
    p          1e-3;
    U          1e-4;
    e          1e-3;

    // possibly check turbulence fields
    "(k|epsilon|omega)" 1e-3;
}
```

}

## relaxationFactors

```
{
    fields
    {
        p          0.4;
    }
    equations
    {
        p          0.4;
        U          0.6;
        e          0.6;
    }
}
```



```
        k            0.3;
        epsilon      0.3;
    }
}

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



UNIVERSITY *of the*  
WESTERN CAPE

# Bibliography

- [1] User guided, openfoam.org, 2018. [Online]. Available: <http://www.OpenFOAM.org/docs/user/>. Accessed: 17-May-2018.
- [2] Cfd direct, 'openfoam user guide:5.1 openfoam case directory', 2017. [Online]. Available: <https://cfd.direct/openfoam/user-guide/v5-introduction/#x3-20001>. Accessed: 19-May-2018.
- [3] Henk Kaarle Versteeg and Weeratunge Malalasekera. *An introduction to computational fluid dynamics: the finite volume method*. Pearson education, 2007.
- [4] Vivek V Ranade. *Computational flow modeling for chemical reactor engineering*, volume 5. Elsevier, 2001.
- [5] Jeong Chul Lee, Ki Hwan Kang, Seok Ki Kim, Kyung Hoon Yoon, Jinsoo Song, and I Jun Park. The influence of filament temperature on crystallographic properties of poly-si films prepared by the hot-wire cvd method. *Thin Solid Films*, 395(1-2):188–193, 2001.
- [6] H. J. Wiesmann, December 2 1980. U.S. Patent 4 237 150.
- [7] H Wiesmann, AK Ghosh, T McMahon, and Myron Strongin. a-si: H produced by high-temperature thermal decomposition of silane. *Journal of Applied Physics*, 50(5):3752–3754, 1979.
- [8] Andrea Pflüger, Bernd Schröder, and Hans-Jörg Bart. Monte carlo simulations on large-area deposition of amorphous silicon by hot-wire cvd. *Thin solid films*, 430(1-2):73–77, 2003.
- [9] Jaydeep V Sali, SB Patil, SR Jadkar, and MG Takwale. Hot-wire cvd growth simulation for thickness uniformity. *Thin Solid Films*, 395(1-2):66–70, 2001.
- [10] Bibhu P Swain and Rajiv O Dusane. Effect of filament temperature on hwcvd deposited a-sic: H. *Materials Letters*, 60(24):2915–2919, 2006.
- [11] Yousef Sharifi and Luke EK Achenie. Effect of substrate geometry on the deposition rate in chemical vapor deposition. *Journal of crystal growth*, 304(2):520–525, 2007.

- [12] Andreas Pflug, Markus Höfer, Tino Harig, Markus Armgardt, Chris Britze, Michael Siemers, Thomas Melzig, and Lothar Schäfer. Modeling of gas flow and deposition profile in hwcvd processes. *Thin Solid Films*, 595:266–271, 2015.
- [13] Chang Song, Yong Lee, Si Heo, Nong-Moon Hwang, Sooseok Choi, and Kwang Kim. Computer simulation of temperature parameter for diamond formation by using hot-filament chemical vapor deposition. *Coatings*, 8(1):15, 2017.
- [14] M Olivas-Martinez, M Perez-Tello, R Cabanillas-Lopez, O Contreras-Lopez, G Soto-Herrera, and F Castillon-Barraza. A computational model for the hot-filament chemical vapour deposition process to produce diamond films. *Modelling and Simulation in Materials Science and Engineering*, 15(3):237, 2007.
- [15] EH Wahl, TG Owano, CH Kruger, Y Ma, P Zalicki, and RN Zare. Spatially resolved measurements of absolute ch<sub>3</sub> concentration in a hot-filament reactor. *Diamond and Related Materials*, 6(2-4):476–480, 1997.
- [16] A Dasgupta, Y Huang, L Houben, S Klein, F Finger, R Carius, and M Luysberg. Effect of filament and substrate temperatures on the structural and electrical properties of sic thin films grown by the hwcvd technique. *Thin Solid Films*, 516(5):622–625, 2008.
- [17] JJH Strengers, Francisco Alberto Rubinelli, JK Rath, and REI Schropp. A combined experimental and computer simulation study of hwcvd nip microcrystalline silicon solar cells. *Thin Solid Films*, 501(1-2):291–294, 2006.
- [18] Abdunaser Sayma. *Computational fluid dynamics*. Bookboon, 2009.
- [19] Jiri Blazek. *Computational fluid dynamics: principles and applications*. Butterworth-Heinemann, 2015.
- [20] Richard E Meyer. *Transonic, Shock, and Multidimensional Flows: Advances in Scientific Computing*. Number 47. Academic Press, 2014.
- [21] Umran Abdul Rahman and Faizal Mustapha. Validations of openfoam steady state compressible solver rhosimplefoam. In *International Conference on Mechanical and Industrial Engineering*, 2015.
- [22] JD Anderson. Basic philosophy of cfd. In *Computational Fluid Dynamics*, pages 3–14. Springer, 2009.
- [23] Wikipedia, 'equation of state', 2015. [Online]. Available: [http://en.wikipedia.org/wiki/Equation\\_of\\_state#Peng.E2.80.93Robinson\\_equation\\_of\\_state](http://en.wikipedia.org/wiki/Equation_of_state#Peng.E2.80.93Robinson_equation_of_state). Accessed: 29-Apr-2019.

- [24] Ding-Yu Peng and Donald B Robinson. A new two-constant equation of state. *Industrial & Engineering Chemistry Fundamentals*, 15(1):59–64, 1976.
- [25] En Zhou. The use of fluent for heat flow studies of the hot-wire chemical vapor deposition system to determine the temperatures reached at the growing layer surface. Master's thesis, University of the Western Cape, 2009.
- [26] David W Hahn and M Necati Özisik. *Heat conduction*. John Wiley & Sons, 2012.
- [27] Yunus A Çengel and Robert H Turner. Steady heat conduction. *Fundamentals of Thermal-Fluid Science*. New York: McGraw-Hill Publishing Company, pages 741–791, 2011.
- [28] Radiation heat transfer in openfoam, 2009. [Online]. Available: [http://www.tfd.chalmers.se/~hani/kurser/OS\\_CFD\\_2009/AlexeyVdovin/Radiation\\_in\\_OpenFoam\\_final.pdf](http://www.tfd.chalmers.se/~hani/kurser/OS_CFD_2009/AlexeyVdovin/Radiation_in_OpenFoam_final.pdf). Accessed: 22-Apr-2019.
- [29] F. Nozaki. buoyantpimplefoam and buoyantsimplefoam in openfoam, 2016. [Online]. Available: <http://caefn.com/openfoam/solvers-buoyantpimplefoam>. Accessed: 03-August-2018.
- [30] C. Greenshields. Computational fluid dynamics, 2016. [Online]. Available: <https://cfd.direct/openfoam/computational-fluid-dynamics/>. Accessed: 09-August-2018.
- [31] Jakob Trydal. Cfd analysis of temperature development due to flow restriction in pipeline. Master's thesis, University of Stavanger, Norway, 2015.
- [32] C. J. Greenshields, H. G. Weller, L. Gasparini, and J. M. Reese. Implementation of semi-discrete, non-staggered central schemes in a colocated, polyhedral, finite volume framework, for high-speed viscous flows. *International journal for numerical methods in fluids*, 63(1): 1–21, 2010.
- [33] CFD Direct. 'openfoam user guide: 5.4 numerical schemes', 2015. [Online]. Available: <http://www.OpenFOAM.org/docs/user/fvSchemes.php>, . Accessed: 07-January-2019.
- [34] Openfoam course, 1st ed. genoa: University of genoa, 2015. [Online]. Available: <http://www.dicat.unige.it/guerrero/of2015a/9tipsandtricks.pdf>. Accessed: 16-May-2019.
- [35] Thomas Engen. Cfd analysis of gas-particle flow in a scaled circulating fluidized bed. Master's thesis, University of Stavanger, Norway, 2016.
- [36] CFD Direct. 'openfoam user guide: 5.5 numerical schemes', 2015. [Online]. Available: <http://www.OpenFOAM.org/docs/user/fvSolution.php>, . Accessed: 07-January-2019.
- [37] BH Hjertager. Lecture notes in openfoam. *Stavanger: University of Stavanger*, pages 3–12, 2009.

- [38] Wikipedia. 'preconditioner', 2015. [Online]. Available: <http://en.wikipedia.org/wiki/Preconditioner>, . Accessed: 19-July-2018.
- [39] B. Hjertager. *Computational Analysis of Fluid Flow Processes*, volume 2. Stavanger, 2002.
- [40] Cfd-online.com. 'two equation turbulence models – cfd-wiki', 2015. [Online]. Available: [http://www.cfd-online.com/Wiki/Two\\_equation\\_turbulence\\_models](http://www.cfd-online.com/Wiki/Two_equation_turbulence_models). Accessed: 19-March-2019.
- [41] Jorge E Bardina, Peter G Huang, and Thomas J Coakley. Turbulence modeling validation, testing, and development. 1997.
- [42] David C Wilcox. *Turbulence modeling for CFD*, volume 2. DCW industries La Canada, CA, 1998.
- [43] Table of emissivity of various surfaces for infrared thermometry, 2017. [Online]. Available: [http://www-eng.lbl.gov/~dw/projects/DW4229\\_LHC\\_detector\\_analysis/calculations/emissivity2.pdf](http://www-eng.lbl.gov/~dw/projects/DW4229_LHC_detector_analysis/calculations/emissivity2.pdf). Accessed: 16-October-2018.
- [44] Nenad D Milošević, GS Vuković, DZ Pavičić, and KD Maglić. Thermal properties of tantalum between 300 and 2300 k. *International journal of thermophysics*, 20(4):1129–1136, 1999.
- [45] G. L. Shires. Prandtl number, 2011. [Online]. Available: <http://www.thermopedia.com/content/1053/>. Accessed: 20-March-2019.
- [46] Wikipedia. Viscosity, 2015. [Online]. Available: <http://en.wikipedia.org/wiki/Viscosity>, . Accessed: 22-March-2019.
- [47] Wikipedia. Prandtl number, 2015. [Online]. Available: [http://en.wikipedia.org/wiki/Prandtl\\_number](http://en.wikipedia.org/wiki/Prandtl_number), . Accessed: 20-March-2019.
- [48] Israel Urieli. Specific heat capacities of air, 2008. [Online]. Available: [https://www.ohio.edu/mechanical/thermo/property\\_tables/air/air\\_cp\\_cv.html](https://www.ohio.edu/mechanical/thermo/property_tables/air/air_cp_cv.html). Accessed: 29-March-2019.
- [49] F Liu. A thorough description of how wall functions are implemented in openfoam. *Proceedings of CFD with OpenSource Software*, pages 1–33, 2016.
- [50] k-epsilon model. [Online]. Available: <https://www.simscale.com/docs/content/simulation/model/turbulenceModel/kEpsilon.html>. Accessed: 30-March-2019.
- [51] Engineering ToolBox. Specific heat of hydrogen gas, 2005. [Online]. Available: [https://www.engineeringtoolbox.com/hydrogen-d\\_976.html](https://www.engineeringtoolbox.com/hydrogen-d_976.html), . Accessed: 11-July-2018.



- [52] Nuclear Power. Hydrogen specific heat, latent heat of fusion, latent heat of vaporization, 2018. [Online]. Available: <https://www.nuclear-power.net/hydrogen-specific-heat-latent-heat-vaporization-fusion/>. Accessed: 01-September-2018.
- [53] CFD Direct. 'openfoam user guide: 7.1 thermophysical models', 2015. [Online]. Available: <http://www.OpenFOAM.org/docs/user/thermophysical.php>, . Accessed: 11-June-2018.
- [54] Engineering ToolBox. Gases - dynamic viscosity, 2014. [Online]. Available: [https://www.engineeringtoolbox.com/gases-absolute-dynamic-viscosity-d\\_1888.html](https://www.engineeringtoolbox.com/gases-absolute-dynamic-viscosity-d_1888.html), . Accessed: 11-September-2018.
- [55] LMNO Engineering. Gases viscosity calculator, 2015. [Online]. Available: <https://www.lmnoeng.com/Flow/GasViscosity.php>. Accessed: 11-September-2018.
- [56] André Bakker. Lecture 10-turbulence models applied computational fluid dynamics. *Power-Point presentation*, 2002.
- [57] Pascal Boulet, Gilles Parent, Zoubir Acem, Anthony Collin, Michael Försth, N Bal, G Rein, and J Torero. Radiation emission from a heating coil or a halogen lamp on a semitransparent sample. *International journal of thermal sciences*, 77:223–232, 2014.
- [58] SR Jadkar, Jaydeep V Sali, ST Kshirsagar, and MG Takwale. The effect of substrate temperature on hw-cvd deposited a-sige: H films. *Journal of non-crystalline solids*, 299: 168–173, 2002.



UNIVERSITY of the  
WESTERN CAPE