Evaluation of CFD tools for crude oil fire simulations (modeling steps in openFOAM and FDS)

Gomaa, Islam; Elsagan, Nour; Lam, Cecilia; Ko, Yoon

https://doi.org/10.4224/23002022

NRC·CNRC CONSTRUCTION

Evaluation of CFD Tools for Crude Oil Fire Simulations
(Modeling Steps in openFOAM and FDS)

Author(s): Islam Gomaa, Nour Elsagan, Cecilia Lam and Yoon Ko

Report No.: A1-010647-02
Report Date: 14-08-2019
Contract No.: A1-010647
Agreement Date: 3 September 2015 (Original agreement)
8 January 2016 (Amendment No. 1)
17 February 2017 (Amendment No. 2)
14 September 2017 (Amendment No. 3)
16 January 2018 (Amendment No. 4)
28 March 2018 (Amendment No. 5)
19 July 2018 (Amendment No. 6)
20 December 2018 (Amendment No. 7)

National Research Council Canada | Conseil national de recherches Canada

Canada

Cat. No. **NR24-37/2019E-PDF**
ISBN **978-0-660-30283-6**

Evaluation of CFD Tools for Crude Oil Fire Simulations
(Modeling Steps in openFOAM and FDS)

Author

Islam Gomaa, Research Officer, PhD

Approved

Ahmed Kashef
2020-10-29 17:27:
19

Ahmed Kashef, FSU Director, PhD
Program Leader
Fire Laboratory Transition Program
NRC Construction Research Centre

Report No: A1-010647.2
Report Date: 14 08 2019
Contract No: A1-010647
Agreement date: 3 September 2015 (Original agreement)
8 January 2016 (Amendment No. 1)
17 February 2017 (Amendment No. 2)
14 September 2017 (Amendment No. 3)
16 January 2018 (Amendment No. 4)
28 March 2018 (Amendment No. 5)
19 July 2018 (Amendment No. 6)
20 December 2018 (Amendment No. 7)
Program: Fire Laboratory Transition

24 pages

Copy no. 1 of 5

(This page is intentionally left blank)

Table of Contents

List of Figures

Executive Summary

Numerical modelling technique has been proposed to simulate fire incidents involving crude oil tank cars. This technique could complement scaled experiments to gain insight into the physics of large fire incidents and therefore better manage involved risk. Benefits of the numerical modeling can be summarized as follows:

- Ability to extrapolate results of the scaled down experiments to a full scale incident;
- Explore several fire scenarios that might be challenging to conduct experimentally such as effect of wind, tank location, tank orientation and spill size; and
- Obtain higher spatial resolution of data such as temperature, incident heat flux and radiative and convective heat transfer.

To this end, two computational fluid dynamics (CFD) tools have been explored to be used in the simulations of crude oil pool fire. The first tool was the “open Field Operation and Manipulation” (openFOAM). The other tool was the “Fire Dynamics Simulator” (FDS). Both tools have been explored in terms of governing equations solved, sub models used, discretization methodology and structure of input file(s). Also, modelling steps in each tool were outlined and explained in this report. A preliminary numerical model of a heptane pool fire, tested in Sandia’s fire facility, was used for demonstration purposes. Advantages and disadvantages of each tool were listed in the “Summary and Conclusions” section of the report.

Both tools could be used to simulate crude oil fires involving engulfed objects. After carefully examining each tool, it would be beneficial to use both tools for the next steps of numerical modeling work. The goal then will be to recommend a robust tool to investigate fire incidents related to the transport of crude oil.

1 Background

Fire incidents involving crude oil tank cars have been the subject of investigation by Transport Canada and the NRC for the last four years. Intermediate-scale experiments of a tank car engulfed in pool fires fueled by crude oil were conducted by the National Research Council of Canada (NRC) at Sandia National Laboratories to characterize the thermal conditions external to the tank car. In these experiments, the tank car was simulated by a $1/10^{th}$ scale cylindrical calorimeter. Numerical modelling of crude oil pool fires is being proposed to enhance analysis and understanding of the Sandia experimental data. Simulations can predict experimentally-measured parameters (like temperature and heat flux) at higher resolution, in addition to other parameters including velocity, soot mass fraction, turbulent kinetic energy, radiative vs. convective heat transfer.

Numerical modelling will help improve understanding of the physics of the fire/calorimeter interaction. For example, in calculating the incident heat flux on the calorimeter from the experimental data, it was assumed that the convective flux from the flame to the calorimeter is negligible since the calorimeter is emerged in the flame (i.e., it was assumed that the heat flux on the calorimeter is only radiative). In the simulations, both the convective and radiative heat flux will be considered. Also the temperature of the calorimeter affects the temperature of the fire itself, where it might have a cooling effect at an early stage of fire. The presence of the engulfed object (calorimeter) in the flame induces turbulence which changes and often enhances the flow, mixing, and consequently the combustion processes within the fire. These interactions can be better understood through comparing the results of modelling the same fire scenario with and without the engulfed object.

Moreover, well-validated models could be used for investigating other variables that were not included in the large-scale fire experiments (e.g., wind speed, location/orientation of tank car and fire size). Open Field Operation and Manipulation (openFOAM) and Fire Dynamics Simulator (FDS) have been widely used in the numerical simulation of pool fires. However, efforts are ongoing to validate these models for fire development including pyrolysis and combustion modelling beyond the transport of heat and combustion products. Using FDS, William [1] simulated a cylindrical calorimeter engulfed in a large pool fire fuelled by JP-4, and Maragkos et al.[2] used FDS and OpenFOAM to simulate helium pool fires tested in an indoor test facility at Sandia. Given reasonable simulation results reported by the previous works, OpenFOAM and FDS were considered to simulate the experiments of a tank car engulfed in pool fires fueled by crude oil conducted by NRC at Sandia National Laboratories.

Generally, fire modelling using computational fluid dynamics (CFD) involves numerical solving of the full partial differential equation set describing the principles of local conservation of mass, momentum, energy and species, subject to the particular boundary conditions of the problem. The numerical solution of the partial differential equations requires the discretization of the domain into small cells (meshing). In addition, the solution of the conservation equations requires the incorporation of different sub-models. In case of modelling of hydrocarbon fires, a *pyrolysis* sub-model is included to model the thermal decomposition of the large hydrocarbon

molecules into smaller ones. The small molecules are volatile and react with atmospheric oxygen based on the *combustion* sub-model incorporated. The heat released from the combustion is then transferred based on the *radiative and convective heat transfer* sub-models applied. Finally, the movement of the smoke and air velocity are modelled using *hydrodynamic* and *turbulence* sub-models.

This report evaluates the capabilities of two computational fluid dynamics (CFD) softwares; OpenFOAM and FDS; in modelling crude oil fires and provides detailed discussion about the different sub-models involved in their simulation. As a demonstration, a preliminary simulation of the crude oil fire experiment conducted by NRC at Sandia National Laboratories using a $1/10^{th}$ scale engulfed tank car is also presented.

Both softwares are well developed and widely used in the fire community. Several works in literature have been conducted using FDS [3], [4] and OpenFOAM [5], [6] for simulating hydrocarbon pool fires. However, each software has its limitations, which are listed in section 4.

2 Open Field Operation and Manipulation (OpenFOAM)

This section is about using a new modelling tool for pool fires. The new tool is an open source software called OpenFOAM [7]. FM Global created fireFOAM as a software package based on OpenFOAM, which includes physical and combustion models related to fire dynamics [14]. Subsections were included about the software, the methodology used for solving the governing equations, modelling steps for this software and post-processing simulation results. The objective was to explore the capabilities of this tool for pool fire modelling. Sandia's Fire Laboratory for Accreditation of Modeling by Experiment (FLAME) facility [8] was used as an example to demonstrate the relevant steps required to run the simulation. Some models and respective parameters are still "arbitrary" at this point since the purpose of this report was to demonstrate how to use this software and explore its capabilities.

2.1 About OpenFOAM

OpenFOAM code is a general CFD software package for simulating thermo- and fluid-dynamics, chemical reactions, solid dynamics and electromagnetics, and it solves various partial differential equations using finite volume method on structured and unstructured mesh [9].

OpenFOAM is an object-oriented C++ library for computational continuum mechanics. For example, the velocity field can be represented in programming code by the symbol U that belongs to a "vectorField" class. The velocity field U would then be an instance, or an object, of the "vectorField" class; hence the term object-oriented [10].

OpenFOAM include executables, known as applications. The applications fall into two categories: solvers that are each designed to solve a specific problem in continuum mechanics; and utilities, which are designed to perform simple pre-and post-processing tasks, mainly involving data manipulation and algebraic calculations [10].

OpenFOAM also includes a set of precompiled libraries that are dynamically linked during compilation of the solvers and utilities. Libraries are source codes containing various models such as turbulent models, thermo-physical models, radiation models, etc. The overall structure of OpenFOAM is shown in Figure 1 [9]

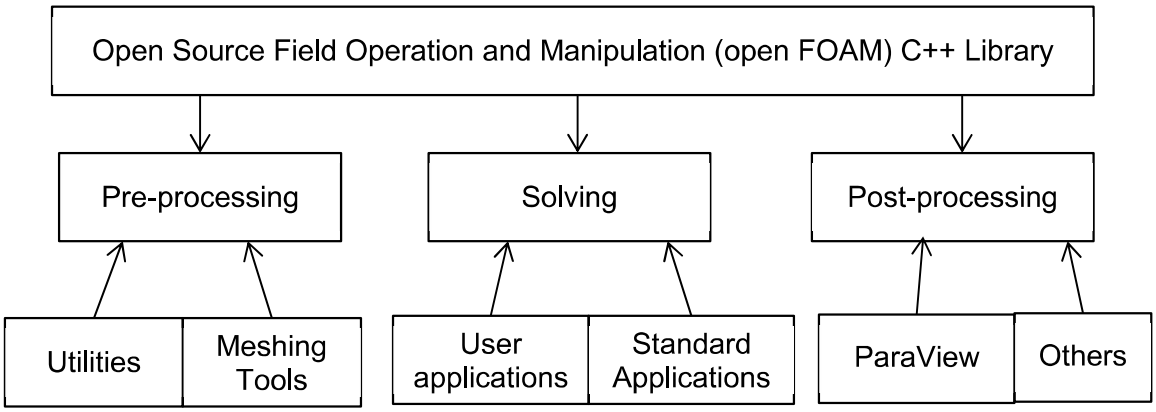

Figure 1. Overview of OpenFOAM structure

OpenFOAM solver applications are written in a syntax that is very similar to the partial differential equations being solved [9]. For example the equation

$$\frac{\partial \rho U}{\partial t} + \nabla . \emptyset U - \nabla . \mu \nabla U = -\nabla p$$

Is represented by the code

```
solve
(
fvm::ddt(rho, U)
+ fvm::div(phi, U)
- fvm::laplacian(mu, U)
==
- fvc::grad(p)
);
```

2.2 Governing Equations

The solver used in the simulation is fireFoam. FireFoam is a transient compressible flow solver for fire applications. Governing equations solved by fireFoam are [11]:

- Continuity
- Momentum
- Energy
- Species transport

- Ideal gas law

2.3 Models

2.3.1 Turbulence Model

The turbulent flow model used is the comprisable Large Eddy Simulations (LES) model. The sub grid scale model used is the one equation eddy viscosity model [12]. In this model eddy viscosity (υ_{sgs}) is computed using the equation:

$$\upsilon_{sgs} = C_k \sqrt{k_{sgs}\Delta}$$

Where C_k is a constant coefficient, k_{sgs} is the sub grid scale kinetic energy calculated using a partial differential transport equation. Δ is the LES filter width.

2.3.2 Combustion Model

The combustion model used in the summations is the Eddy Dissipation Model (EDM). This model is based on the fast-chemistry assumption, meaning that once the fuel and air are mixed, they are burned immediately. The reaction rate is expressed as [13]:

$$\omega = \frac{\rho \min\left(Y_f, \frac{Y_{O_2}}{S}\right)}{\Delta \mathrm{tC}_{stiff}} (1 - \exp(-C_{stiff}\Delta t r_t))$$

Where ρ, Y_f, Y_{O2} and S are the density, fuel mass fraction, oxygen mass fraction and oxygen stoichiometric coefficient. C_{stiff} is a constant coefficient. r_t is the reciprocal time scale defined as [13]:

$$r_t = \max(r_{tTurb}, r_{tdiff}),\ r_{tTurb} = C_{EDC}\left(\frac{\varepsilon}{k}\right),\ r_{tDiff} = C_{diff}\left(\frac{\alpha}{\rho\Delta^2}\right)$$

Where r_{tTurb} and r_{tdiff} are reciprocals of the turbulent time scale and diffusion time scale, respectively. k is the turbulent kinetic energy and ε is the energy dissipation rate. α is the thermal diffusivity and Δ is the LES filter width. C_{EDC} and C_{diff} are constant coefficients.

The Eddy Dissipation Model is not included in the OpenFOAM release. The Eddy Dissipation model has been developed by FM Global.. For the current simulation, only the combustion library has been recompiled to include the EDM.

2.3.3 Radiation Model

Radiation heat transfer, in numerical fire modeling, is accounted for by numerically solving what is called the radiative transfer equation (RTE). The equation of radiative transfer is an energy balance equation specific to radiation in which, radiation intensity of a traveling beam loses energy by absorption, gains energy by emission, and redistributes energy by scattering. Figure 2 illustrates

the process of radiative heat transfer as described by the RTE. More details on the terms of the RTE equation are in section 3.2.4.

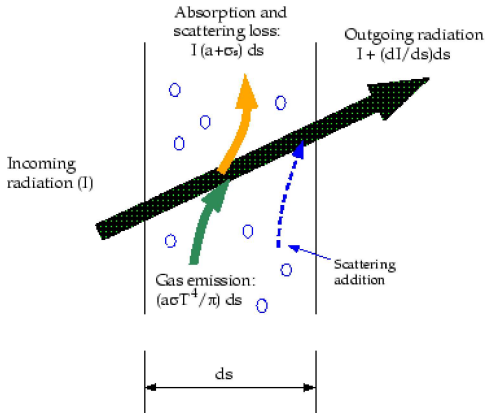

Figure 2. Radiative heat transfer [15]

Two radiation models are offered in OpenFOAM:

The finite volume discrete ordinates model: The discrete ordinates (DO) radiation model solves the radiative transfer equation (RTE) for a finite number of discrete solid angles

The P1 model: The P-1 radiation model is based on the expansion of the radiation intensity into an orthogonal series of spherical harmonics (i.e. special functions defined on a surface of a sphere)

More details on the radiation models can be found here [16], [17] and [18].

2.3.4 Pyrolysis Model

No pyrolysis model has been used in current simulation but OpenFOAM has a solid phase pyrolysis application.

2.4 Numerical Solution Methodology

For discretization, the finite volume method is used in OpenFOAM. The problem is all discretized as follows [10]:

Spatial discretization: The solution domain is defined by a set of points that fill and bound a region of space when connected;

Temporal discretization: The problem is divided in the time domain into a finite number of time intervals.

Equation discretization: The partial differential equations are transformed into linear algebraic equations containing variables defined at specific locations equivalent to the spatial discretization.

Space discretization is done by subdividing the domain into control volumes (cells). A typical cell is shown in Figure 3. Dependent variables are usually stored at the cell centroid "*P*". Otherwise they are stored on the cell faces or vertices. The finite volume is bounded by its faces "*f*". The mesh in openFOAM is usually referred to as "arbitrary unstructured", meaning that the finite volume faces are not limited in number or orientation. This offers more flexibility in terms of the shape of the domain to be modeled and local refinement of the mesh size. In openFOAM, faces are defined by a list of points. Cells are defines by a list of faces.

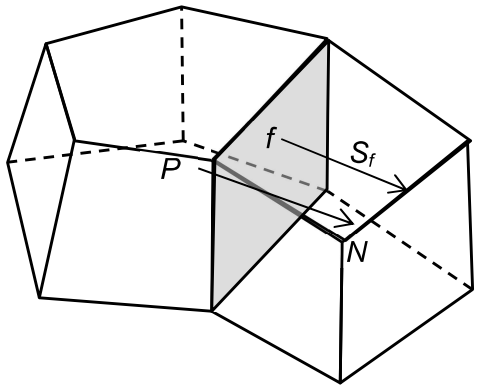

Figure 3. Typical cell in finite volume discretization [10]

Partial differential equations contain terms like the Laplacian, the divergence, the Gradient…etc. Finite volume discretization of these terms is done by integrating the term over the cell volume. However, in order to reach a set of algebraic equations, the volume integral is converted to surface integral using Gauss's theorem

$$\int_V \nabla \star \emptyset dV = \int_s dS \star \emptyset$$

Where *S* is the surface area vector, ϕ can represent any tensor field and the star notation "$\star$" represents tensor operators.

Surface integrals are then linearized using appropriate schemes such as the standard Gaussian finite volume integration. Gaussian integration is done by interpolating cell face values from the cell centers then the summation of the face values. Shown below are some examples of differential terms discretization and integration [10].

The Laplacian term

$\int_V \nabla.(\Gamma\nabla\emptyset)dV = \int_S dS.(\Gamma\nabla\emptyset) = \sum_f \Gamma_f S_f.(\nabla\emptyset)_f$, Where Γ is a diffusivity coefficient.

The convective term

$$\int_V \nabla.(\rho U \emptyset) dV = \int_S dS.(\rho U \emptyset) = \sum_f S_f.(\rho U)_f \emptyset_f$$

Where the subscript "f" indicates a face value. There are several schemes offered in OpenFOAM to evaluate face values and gradients between cells.

The divergence term

$$\int_V \nabla.\emptyset dV = \int_S dS.\emptyset = \sum_f S_f.\emptyset_f$$

The gradient term

$$\int_V \nabla \emptyset dV = \int_S dS\, \emptyset = \sum_f S_f\, \emptyset_f$$

2.5 Case Structure and Input Files

Input files are distributed in three main directories:

1. A "constant" directory

The constant directory contains the mesh, thermodynamic properties, radiation model and properties, turbulent model and the combustion model. List of the constant directory files is given below.

- **polyMesh:** domain mesh and its boundaries
- **combustionProperties:** combustion model and its parameters
- **g:** gravity
- **hRef:** reference value for enthalpy (usually 0)
- **pRef:** reference value for pressure (101325 pa in our case)
- **radiationProperties:** radiation model and its parameters. The soot model and scattering models are defined here as a well
- **reactions:** reactions
- **thermo.compressibleGas:** thermodynamic properties: polynomial coefficients for specific heat and viscosity
- **thermophysicalProperties:** thermodynamic model is defined here as well as EoS
- **turbulenceProperties:** turbulence model and its parameters

2. A "system" directory

The system directory contains the settings for solution procedures, control parameters (the controlDict file), discretization scheme (fvSchemes file) and the settings for the linear equations solver (fvSolution). List of the system directory files is given below:

- **blockMeshDict:** initial/preliminary mesh generator
- **controlDict:** main simulation control parameters
- **decomposeParDict:** mesh decomposition setting
- **fvSchemes:** discretization schemes
- **fvSolution:** settings for the linear equations solver
- **meshQualityDict:** mesh quality control parameters
- **snapyHexMeshDict**
- **surfaceFeaturextractDict**

3. A “time” directory

The time directory includes the solution at each time step. The “0” time directory contains the initial boundary conditions (BCs). List of the “0” directory files is given below:

- **alphat:** thermal diffusivity
- **C7H16:** heptane initial mass fraction in the domain and at the boundaries
- **G:** initial radiation intensity within the fluid domain
- **IDefault:** initial radiative heat flux at the boundaries
- **K:** initial turbulent kinetic energy and at the boundaries
- **N_2:** nitrogen initial mass fraction and at the boundaries
- **nut:** initial turbulent viscosity in the domain and at the boundaries
- **O_2:** oxygen initial concentration and at the boundaries
- **P:** initial static pressure
- **Ph_rgh:** initial hydrostatic pressure
- **Soot:** initial soot concentration in the domain
- **T:** thermal BCs
- **U:** flow BCs
- **Ydefault:** initial mass fraction of other species that are not explicitly specified in a separate file

2.6 Modeling Steps

Main steps for the modeling process are:

- Obtaining a geometrical model
- Meshing the domain
- Imposing boundary conditions, initial conditions and selecting the models needed (case setup)
- Running the simulation
- Post processing

2.6.1 Geometrical model

The geometry has been modeled using an open source software that provides pre- and post-processing platform for numerical simulations. The software is called Salome [19]. Boundaries of the domain were exported in "***.stl**" format to be used by SnappyHex; the OpenFOAM meshing tool. The geometry obtained using Salome is shown in Figure 4.

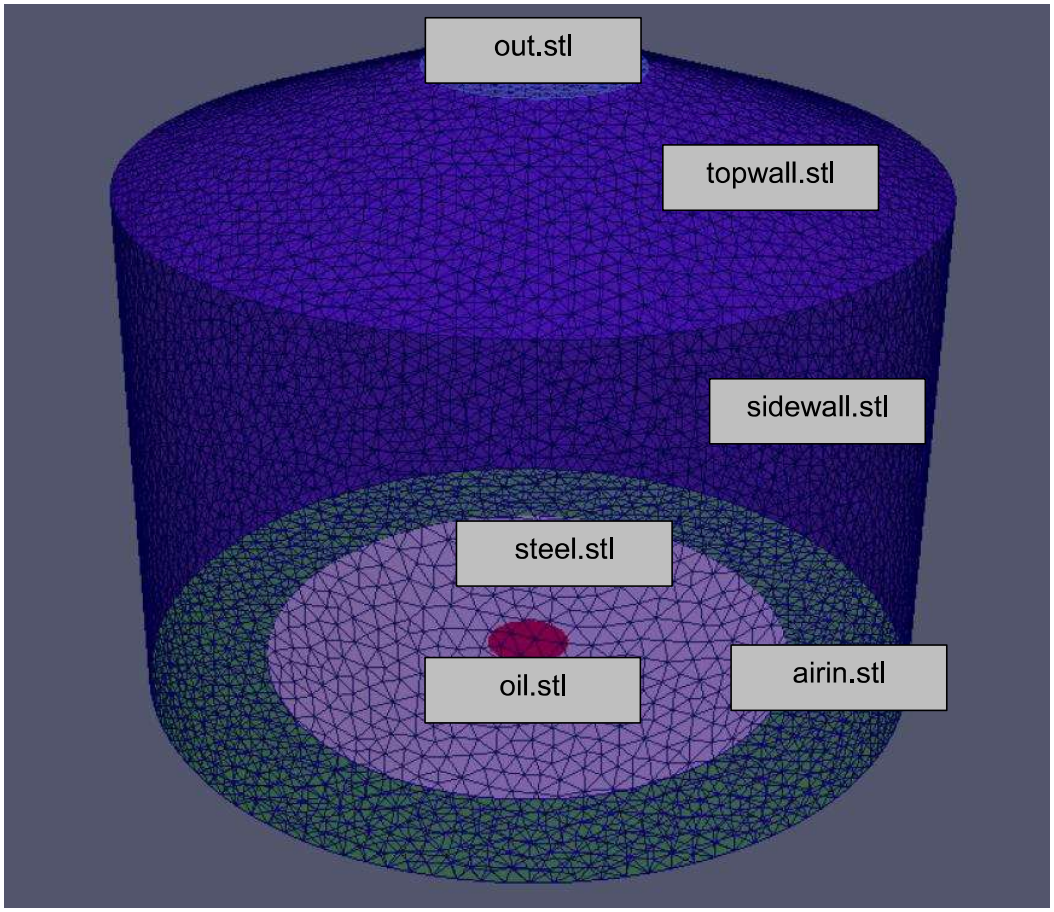

Figure 4. "stl" geometry obtained using Salome

The triangular mesh appearing on the "stl" surfaces are not the mesh used in the simulation. They are the triangulated surfaces by which the "Standard Triangle Language" (stl) describes curved surfaces.

2.6.2 Meshing

Meshing was conducted using the standard meshing tools of OpenFOAM: BlockMesh and snappyHexMesh.

blockMesh decomposes the domain into a set of 3-dimensional, hexahedral blocks. Edges of the blocks can be straight lines, arcs or splines [20]. The domain is meshed by specifying the number of cells in each direction of the block. More than one block can be used to cover the simulation domain. Each block of the geometry is defined by 8 vertices, one at each corner of a

hexahedron. The mesh generated by blockMesh has to be bigger than the geometry specified by the “stl” files

The snappyHexMesh utility generates 3-dimensional meshes containing hexahedra (hex) and split-hexahedra (split-hex) automatically from triangulated surface geometries, or tri-surfaces, in “stl” or Wavefront Object “obj” format [21]. The mesh approximately conforms to the surface by iteratively refining a starting mesh and morphing the resulting split-hex mesh to the surface.

The mesh is generated using snappyHexMesh through the following steps [22]:

- Create a background mesh using the blockMesh utility. The background mesh has to be bigger than the geometry defined by the “stl” surfaces
- Cell splitting: next step is refinement of featured edges, surfaces and geometries. Refinement setting is adjusted under the castellation settings subdirectory. In this case, the air inlet, steel plate, side walls, top wall has been assigned refinement level 1. This means that the mesh cells at these boundaries were split in the x, y and z direction once. The oil surface was assigned two levels of refinement. A cylindrical domain (2.4 m diameter X 12 m height) above the oil was refined twice as well. That means that the number of cells, in this cylinder, has been multiplied 64 times.
- Cell removal: unwanted cells are removed from the domain based on specifying one point within the fluid domain. Any volume/domain that cannot be reached or isolated by a “stl” surface is removed from the domain.
- Snapping to surface: cell vertex points near a “stl” surface/boundary are moved /shifted to the “stl” surface to remove the jagged castellated surface created in the cell splitting step
- Last step is adding thin layers to resolve the boundary layer flow. This step was not conducted for this simulation.

The final mesh is shown in Figure 5.

2.6.3 Case Setup

Case setup is done through the following steps

- Changing the initial conditions in the “0” directory
- Selecting suitable boundary conditions for the case (also in the “0” directory)
- Choosing the models and their respective parameters in the constant directory
- Selecting the discretization scheme in the “fvScheme” subdirectory
- Selecting the solution scheme for the set of linearized equation in the “fvSolution” subdirectory

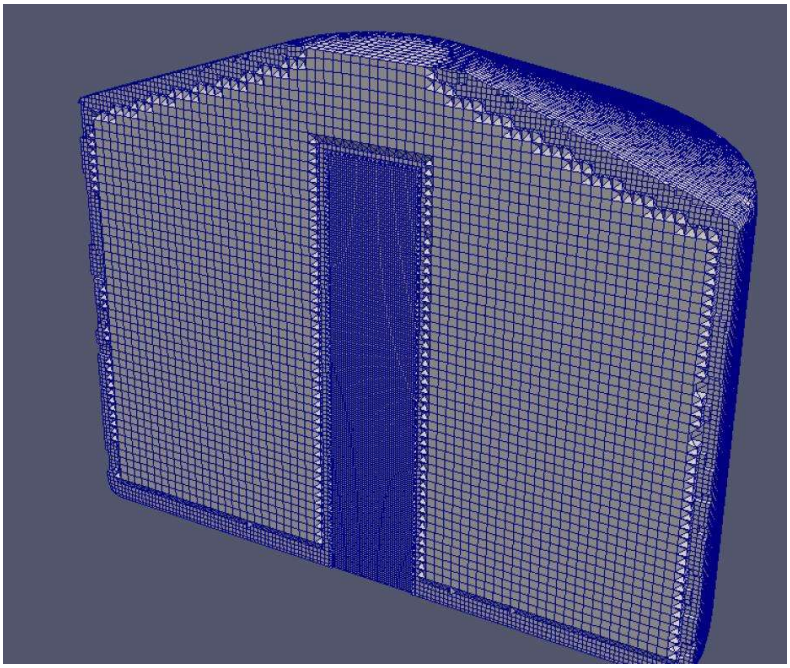

Figure 5. Final mesh using the snappyHexMesh utility

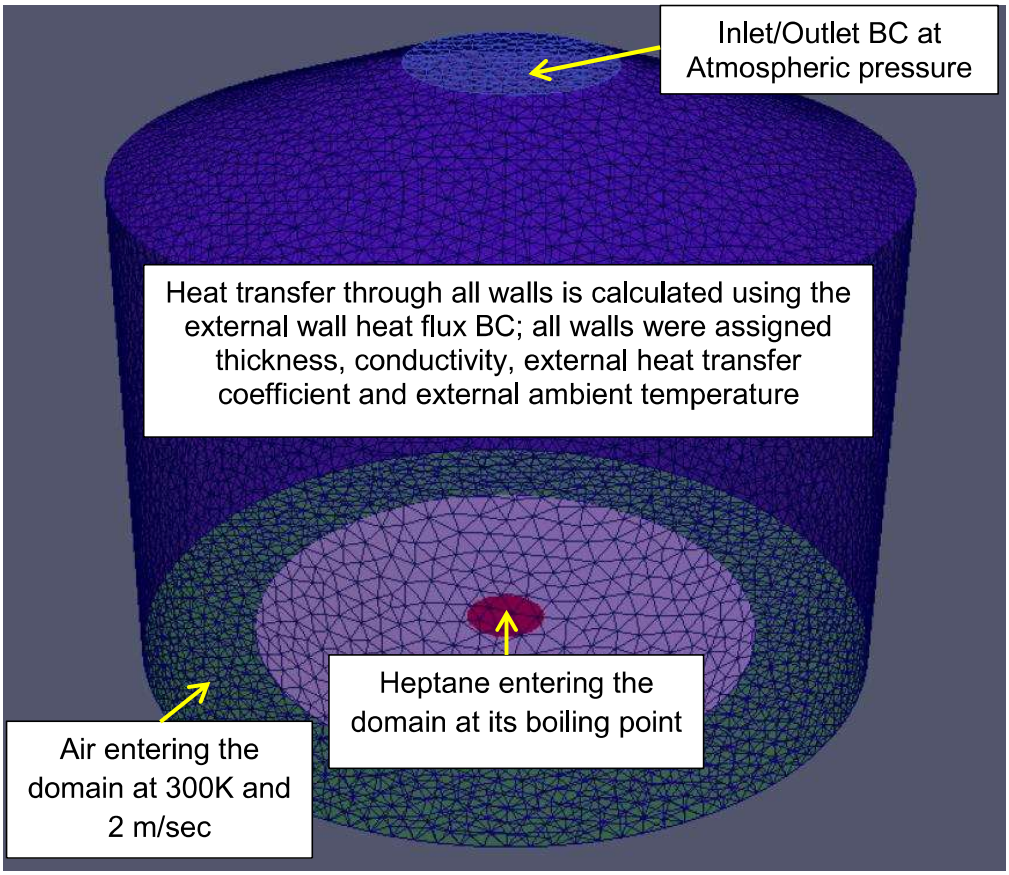

Figure 6. Boundary conditions

2.6.4 Running the Simulation

The simulation is run from a terminal using the following commands [22]:

- blockMesh
- surfaceFeaturesExtract
- decomposePar
- mpirun -np 27 snappyHexMesh -overwrite –parallel
- reconstructParMesh –constant
- decomposePar
- mpirun -n 27 renumberMesh -overwrite –parallel
- mpirun -np 27 fireFoam -parallel

2.6.5 Post Processing

For post processing the case was reconstructed and post processing software was used. The post processing software is called ParaView [23]. ParaView is an open-source, multi-platform data analysis and visualization application [23] .

Shown in figures 7 to 14 are examples of post processing results from ParaView. Please note that these are not final results at this point. This is just a demonstration exercise. Most models and parameters have to be refined or validated.

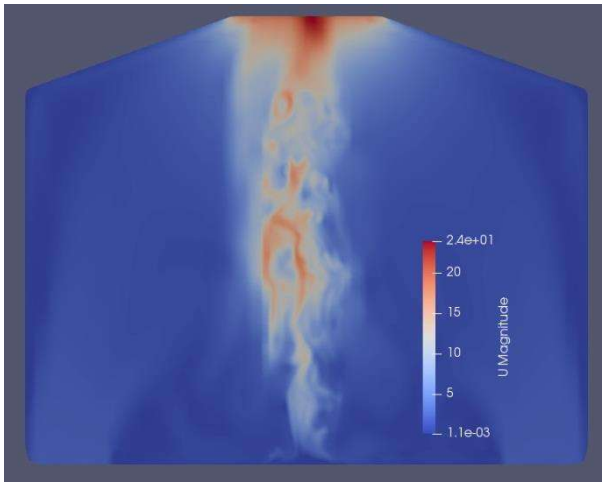

Figure 7. Velocity magnitude slice

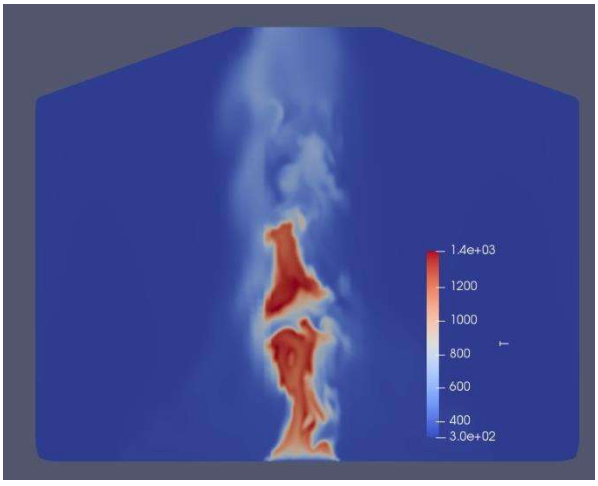

Figure 8. Temperature slice

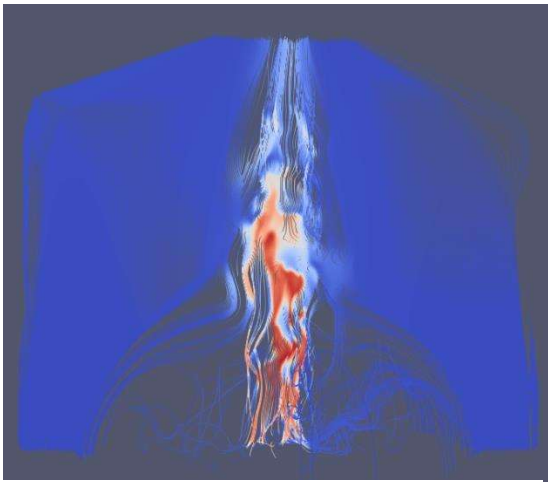

Figure 9. Stream lines colored by temperature

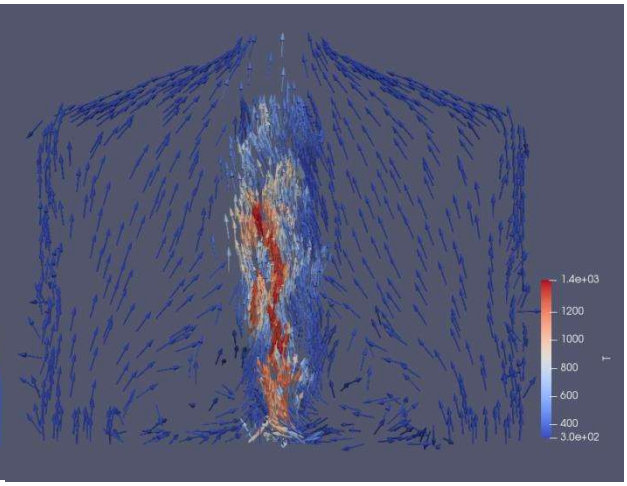

Figure 10. Velocity vectors colored by temperature

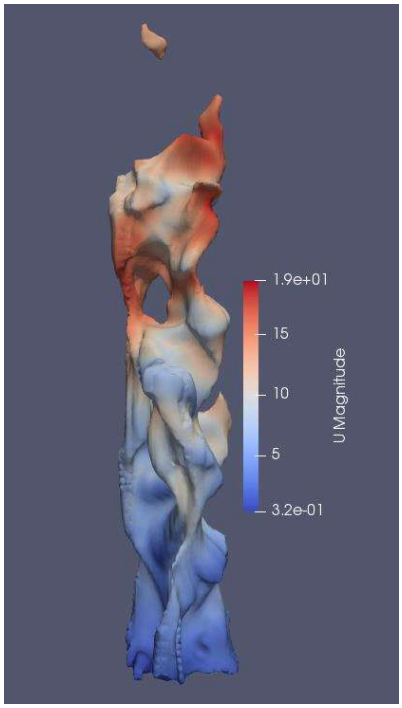

Figure 11. Temperature iso-surface of 1000k colored by velocity magnitude

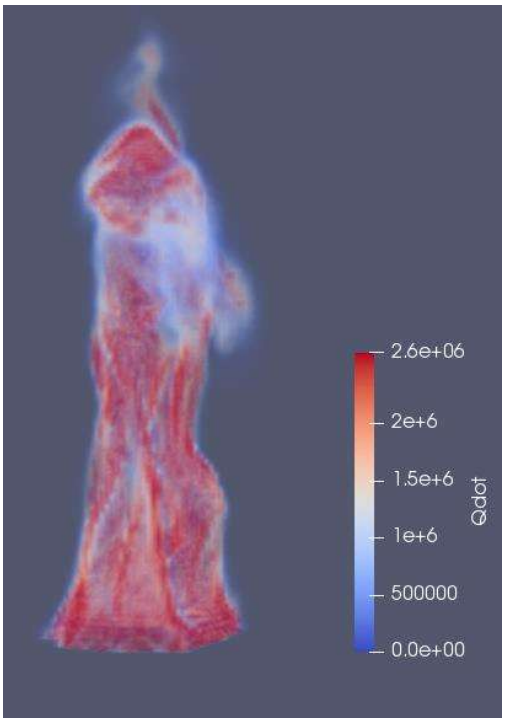

Figure 12. Volume rendering of heat release rate

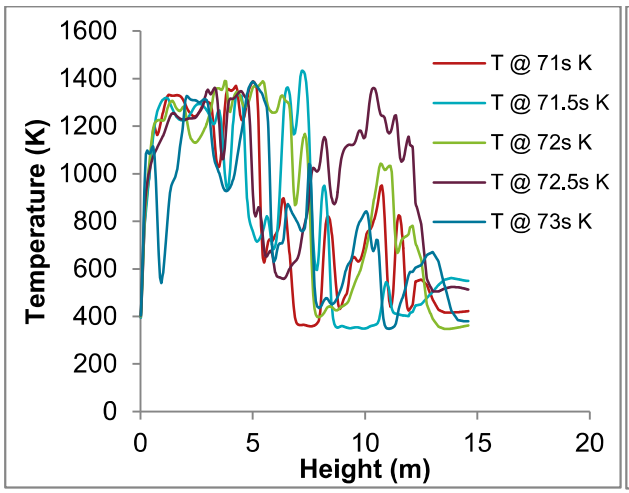

Figure 13. Centerline temperature at different time intervals

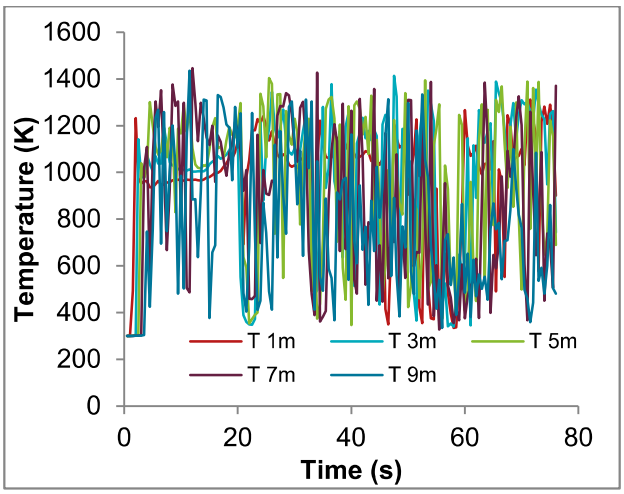

Figure 14. Temperature versus time at different heights

3 Fire Dynamics Simulator (FDS)

This Section provides detailed discussion about the capabilities of FDS in modelling crude oil fires and the different sub-models involved in their simulation.

3.1 About FDS

Fire Dynamic Simulator (FDS) is a 3D computational fluid dynamic (CFD) software used for simulating fire evolution and propagation as a thermally driven flow. It is an open-source code developed by National Institute of Standards and Technology (NIST) [24]. FDS aims at solving practical fire problems in fire protection engineering, while at the same time providing a tool to study fundamental fire dynamics and combustion. The code is very specialized in fire simulations, hence it has many sub-models focusing on smoke movement and control, fire suppression, and heating, ventilation and air conditioning (HVAC) systems. The core algorithm is an explicit predictor-corrector scheme, second order accurate in space and time. FDS uses a structured and uniform grid, and a simple immersed boundary method for treatment of flow obstructions. The chemical reaction between fuel and oxidizer is treated as a fast single step using a reaction progress variable. The radiation transport equation is solved based on gray gas radiation with finite volume solution. More details about the different sub-models implemented in FDS are discussed in the following sections.

3.2 Models

FDS treats turbulence by means of Large Eddy Simulation (LES). Gas phase combustion is solved using a single step, mixing controlled chemical reaction which uses 3 lumped species (fuel, oxidizer and products). FDS also accounts for radiative heat transfer using a technique similar to finite volume methods for convective transport (FVM). More details about these models are provided in the following sub-sections.

3.2.1 Hydrodynamic model

The core of any CFD model is its Navier-Stokes solver, which is used to solve the fluid dynamics and turbulence. Three well-known solvers can be used for solving the fluid dynamics (i.e., Navier-Stokes equation): Reynold's Average Navier-Stokes (RANS), Large Eddy Simulation (LES) and Direct Numerical Simulation (DNS). RANS provides a time-averaged solution to the Navier-Stokes equation and finds a steady-state or quasi steady-state solution. Time dependent flows can be solved as long as the time scale of the mean flow is large compared to the time scale of the turbulent fluctuations. On the other hand, DNS provides detailed solution of the Navier-Stokes equation. However, it requires very small grids (cells) (order of mm), which are computationally very expensive, especially in simulating the large domains required for accidental fire scenarios. LES is in-between both solvers; it solves the detailed fluid dynamics, similar to DNS, within the bulk of the domain, while using simplified equations, similar to RANS, at regions of high shear (e.g., the solid wall boundaries and the interface of the hot and cold flows in doors and windows).

Turbulence modelling and time accuracy are closely related. The present simulation uses LES, which is recommended for fire-driven flows. LES is considered to be accurate in time, meaning that the variations in the solution correspond to the motions resolvable by the numerical grid. This is important for accurate modelling of fire because the dynamic motions, or eddies, are responsible for most of the air entrainment into the fire plume, as found by Rehm and Baum [25].

The marching in time takes place using a short time step Δt, which is usually defined by the following stability criteria;

$$\Delta t_{ijk} < \min\left(\frac{\Delta x_{ijk}}{u_{ijk}}\right)$$

Where Δx and u are the grid cell size and velocity, respectively.

3.2.2 Pyrolysis model

During fire, the large complex molecules forming the fuel break down into smaller ones, which then go to the gas phase (volatiles). This process is called "pyrolysis". The rate of pyrolysis is a function of temperature, rate parameters and the quantity of fuel. The rate of pyrolysis in FDS is defined in the form of an Arrhenius equation;

$$r_{ij} = A_{ij} Y_{s,i}^{n_{s,ij}} \exp\left(\frac{-E_{ij}}{RT_s}\right), Y_{s,i} = \left(\frac{\rho_{s,i}}{\rho_s(0)}\right)$$

where r_{ij} defines the rate of reaction at the temperature, T_s, of the i^{th} material undergoing its j^{th} reaction; $Y_{s,i}$ is the ratio of the instantaneous and initial densities of the i^{th} material; $n_{s,ij}$ is the order of the reaction with respect to the material and oxygen; A_{ij} and E_{ij} are the rate parameters and defined as pre-exponential factor and reaction activation energy, respectively; and R is the universal gas constant. Both A_{ij} and E_{ij} are input to the model.

Generally, different crude oils have different composition which consequently affect their pyrolysis rates and burning behaviour during fire. Pyrolysis rate parameters are usually developed using experimental data from thermo-gravimetric analysis (TGA) and differential scanning calorimetry (DSC). In the present simulations, the crude oil pool was assumed to emit a certain amount of volatiles at a constant rate due to the lack of available pyrolysis rates for the experimentally investigated crude oils.

FDS numerically solves the one-dimensional heat transfer equation to compute the temperature and reactions inside the solids (i.e. fuel). The size of the cells on the surface of the solid is automatically chosen to be smaller than the square root of the material diffusivity, which is prescribed in the input file. By default, the solid mesh cells increase in size towards the middle of the material layer and are smallest on the layer boundaries.

3.2.3 Combustion model

The volatiles produced from the pyrolysis of the fuel then react with the atmospheric oxygen to release heat, part of which goes back to the fuel to maintain burning. This reaction between the volatiles and oxygen is called the gas phase combustion.

Combustion is modeled in FDS as a single step and mixing-controlled, where the rate of combustion reaction is assumed to be infinitely fast. Only 3 lumped species are considered; fuel, air and products. The products include: carbon dioxide, carbon monoxide, soot, nitrogen and water vapor. In addition, the heat of combustion of the reaction, which is the heat released in kJ from the combustion of 1 kg of the fuel, is defined.

The turbulence-chemistry interaction is treated based on the eddy dissipation model (EDC) [26], which is based on the fact that only a fraction of the air/fuel mixture within a computational cell can react. Accordingly, the mean chemical source term for the fuel ($\dot{m}_F''$) is calculated in FDS from:

$$\dot{m}_F'' = -\rho \frac{\min\left(\frac{Y_F, Y_A}{s}\right)}{\tau_{mix}}$$

Where Y_F and Y_A are the cell mean mass fractions of fuel and air, respectively; s is the mass stoichiometric coefficient for air; and τ_{mix} is the mixing time scale.

3.2.4 Radiation model

Radiation is the main mode of heat transfer in large hydrocarbon fires. It allows the gaseous combustion products to cool due to the emitted radiation and preheats combustible materials ahead of the flame front. This preheating increases the rate of flame spread, often causing ignition of surfaces without direct flame impingement. The most important radiation parameter is the fraction of energy released from the fire as thermal radiation (radiative fraction). It is a function of both the flame temperature and chemical composition. Due to the relatively coarse mesh used in FDS (order of centimeters), the flame sheet is not very well captured. Hence,

radiative fraction must be input to the model. In the preliminary crude oil simulations presented herein, a radiative fraction of 30% was assumed.

Radiative heat transfer is included in the model via the solution of the radiation transport equation for a gray gas. The equation is solved using the Finite Volume Method (FVM). The radiation transport equation (RTE) is:

$$s.\nabla I_\lambda\,(x,s) = -[\kappa_\lambda(x)+\sigma_\lambda(x)]I_\lambda(x,s)+\kappa_\lambda(x)I_b(x,\lambda)+\frac{\sigma_\lambda(x)}{4\pi}\int_{4\pi}\phi(s,s')I_\lambda(x,s')d\Omega'$$

where s is the unit direction vector; I_λ is the intensity at wavelength λ; $\kappa_\lambda(x)$ and $\sigma_\lambda(x)$ are the local absorption and scattering coefficients at λ, respectively; I_b is the emissive power of the medium; and Φ(s,s') is the scattering phase function giving the scattered intensity from direction s to s'. The terms of the RTE have the following interpretations: The left-hand side is the rate of change of the intensity in direction s; the first right-hand side term describes the attenuation by absorption and scattering to other directions; the second right-hand side term is the emission source term; the last right-hand side term is the in-scattering integral, describing how much intensity is gained by scattering from all the other directions to the present direction. The intensity depends on place, direction and wavelength.

3.3 Meshing and parallel computing

All FDS calculations must be performed within a domain that is made up of rectilinear volumes called meshes. A mesh is a single box that is divided into uniform cells, the number of which depends on the desired resolution of the flow dynamics. The smaller the size of the cell, the more accurate the result. However, a larger number of cells within a mesh slow down computational time.

To decrease the computational "clock" time while keeping the same resolution (i.e. same number of cells), the domain can be divided into multiple connected meshes. The meshes can be distributed over an equal (or less) number of processors using parallel computing. There should be a comparable number of cells within each mesh.

In the present simulation, the domain was divided into cells of size 10 cm × 10 cm × 10 cm. The cells were distributed into 500 meshes and the simulation was conducted over 500 processors for 12.8 hours of computational time on the Federal Government cluster. Figure 15 shows the cells in the domain.

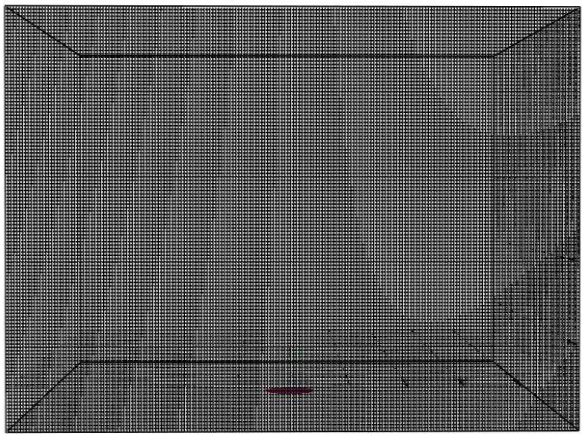

Figure 15. The mesh of the domain

3.4 Input file and Domain

The source code of FDS is written in FORTRAN. The input file to FDS is a single ASCII text file which provides all the necessary information to describe the scenario. The file includes:

- Global parameters such as simulation time, domain, mesh, hydrodynamic solver (discussed in section 3.2.1),
- The geometry of the model and boundary conditions,
- Different sub-models involved in the simulation and their associated parameters.

The domain and geometry in FDS are based on rectilinear mesh, where all obstructions are assumed to conform to orthogonal shapes, and all bounding surfaces are assumed to be flat planes.

Figure 16 shows the domain used in simulating the 1/10th scale engulfed tank car experiment. The domain size is 25 m × 25 m × 12.5 m and the exterior boundaries are treated as passive openings to the outside (open boundary). The crude oil pool is simulated as a circular vent at the center of the domain. The engulfed calorimeter is placed as a cuboid 1 m above the pool fire. It should be noted that, the experimental object was cylindrical, however it is cuboid in the simulation due to the limitation of FDS to rectilinear mesh. The simulation time was 1000 s.

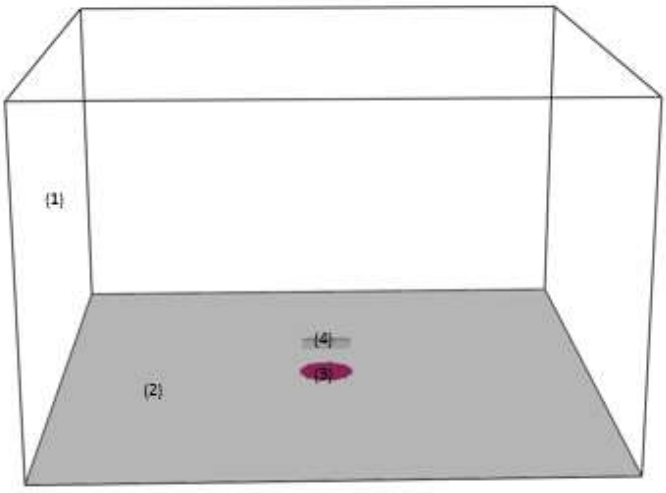

Figure 16. FDS domain for the crude oil experiment. (1) Open boundaries, (2) concrete floor, (3) pool fire and (4) calorimeter

3.5 Output and Post-processing

FDS provides different types of outputs: line plots, iso-surfaces, contours and 3D videos. The binary files output from FDS can be visualized using Smokeview [27] or Pyrosim software.

Smokeview uses quantitative physics-based methods to visualize smoke color and opacity. Flame color and heat release are visualized using an arbitrary user-specified color palette where color is mapped to gas temperature. Figure 17 shows a screenshot of the smoke and heat release from the preliminary crude oil simulation. Also heat release rate (HRR) profiles against time can be provided as seen in Figure 17. Note that a constant HRR of 6 MW was input to the model in this preliminary simulation. Figure 18 also shows the rate of heat transfer by radiation which is 30% (2 MW) of the total heat release. This is based on the radiative fraction specified in the input file to FDS (refer to section 3.2.4).

Figure 17. Screenshot of HRR and smoke in the preliminary simulation

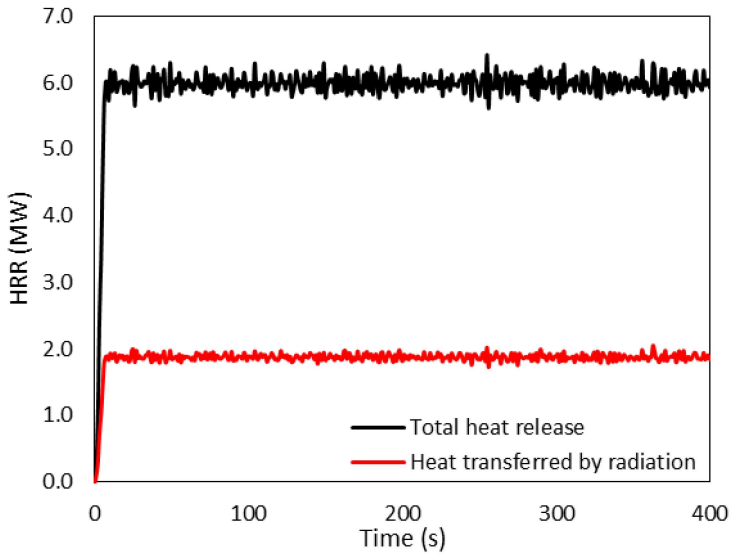

Figure 18. Total HRR and rate of heat transferred by radiation profiles from the crude oil preliminary simulation

Temperature contours (slices) obtained from FDS can also be compared to the experimental figures from the IR camera. Figure 19 shows a temperature slice (contour) at the centerline of the domain from the current simulation. Also contours for mass fractions of carbon monoxide, carbon dioxide and soot can be provided. Moreover, FDS has the capability to calculate the temperatures, net heat flux, gauge heat flux and radiative heat flux at different locations in the domain. These calculations will be compared against the experimentally measured values.

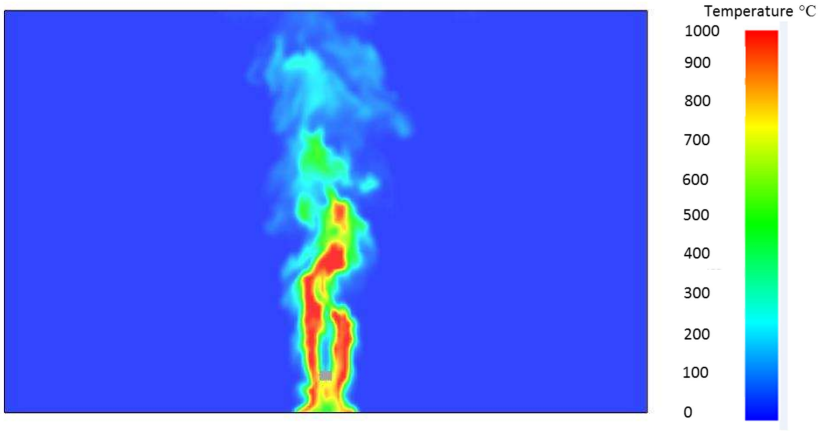

Figure 19. Temperature slice (contour) at the centerline of the domain.

4 Summary and Conclusions

The advantages for using OpenFOAM are:

- It's an open source software that is accessible by any user without the need for a license
- OpenFOAM contains a lot of precompiled models for reacting flow

- Adding new models is possible following the programming guide of OpenFOAM
- Curved and complicated geometries are easy to model in OpenFOAM, thus the cylindrical calorimeter can be easily modelled.
- Local and dynamic mesh refinement is possible in OpenFOAM
- Parallelization and mesh decomposition is achieved automatically
- OpenFOAM supports massive parallel computing
- Multiphysics simulations are possible in OpenFOAM. Oil behaviour inside the tank can be modeled in the same simulation. Structural behavior of the tank can be modeled in the same simulation

The disadvantages of OpenFOAM are:

- Steep learning curve
- Requires powerful computation resources: cluster or a powerful workstation

The advantages of using FDS are:

- FDS is accessible by any user without the need for a license
- FDS is designed for engineering application so it is easier to learn and use
- FDS includes numerous utilities tailored for fire characterization such as experimental measurement devices [thermocouples, heat flux meters …etc.]
- FDS cases requires less memory for storage
- FDS simulations are relatively faster but still require powerful computational resources
- FDS can be installed in Windows operating systems

The disadvantages of FDS are:

- Meshing is limited to structured rectilinear meshes. Curved geometries cannot be modeled in FDS
- Mesh local refinement is not usually possible. Transition from fine to coarse mesh cannot be achieved gradually
- Multiphysics simulations are not possible. FDS is designed specifically for fire simulations. However, several work in literature coupled FDS with Multiphysics softwares like; COMSOL [28], [29] , ABAQUS [30], [31] and OOFEM [32].

Due to the different advantages and disadvantages of the two CFD tools listed above, it is difficult, at this point, to select one over the other for future work. Therefore, it is recommended to continue refining the models for Sandia's experiments and conduct next year numerical modeling work using both tools.

Similar studies were conducted without recommending one tool over the other due to the balance between both tools in terms of advantages and disadvantages [13]. Consequently, it is beneficial to conduct next steps using both tools.

References

[1] W. R. Williams, “A Validation Simulation of a Large Pool Fire,” University of Tennessee, Knoxville, 2005.

[2] G. Maragkos, P. Rauwoens, and B. Merci, “Application of FDS and FireFOAM in Large Eddy Simulations of a Turbulent Buoyant Helium Plume,” *Combust. Sci. Technol.*, vol. 184, no. 7–8, pp. 1108–1120, 2012.

[3] A. Jenft, A. Collin, P. Boulet, G. Pianet, A. Breton, and A. Muller, “Experimental and numerical study of pool fire suppression using water mist,” *Fire Saf. J.*, vol. 67, pp. 1–12, Jul. 2014.

[4] J. Wahlqvist and P. van Hees, “Implementation and validation of an environmental feedback pool fire model based on oxygen depletion and radiative feedback in FDS,” *Fire Saf. J.*, vol. 85, pp. 35–49, Oct. 2016.

[5] C. J. Wang, J. X. Wen, and Z. B. Chen, “Simulation of Large-Scale LNG Pool Fires Using FireFoam,” *Combust. Sci. Technol.*, vol. 186, no. 10–11, pp. 1632–1649, Nov. 2014.

[6] P. Chatterjee, Y. Wang, K. V. Meredith, and S. B. Dorofeev, “Application of a subgrid soot-radiation model in the numerical simulation of a heptane pool fire,” *Proc. Combust. Inst.*, vol. 35, no. 3, pp. 2573–2580, Jan. 2015.

[7] “OpenFOAM | The OpenFOAM Foundation,” *OpenFOAM*. [Online]. Available: https://openfoam.org/. [Accessed: 07-Mar-2019].

[8] “Sandia National Laboratories: Burn Site.” [Online]. Available: https://www.sandia.gov/vqsec/facilities/burn-site.html. [Accessed: 07-Mar-2019].

[9] C. Greenshields, “OpenFOAM User Guide: CFD Direct, Architects of OpenFOAM,” *CFD Direct*, 02-Mar-2017. [Online]. Available: https://cfd.direct/openfoam/user-guide/. [Accessed: 26-Feb-2019].

[10] C. Greenshields, “Programmer’s Guide,” OpenFOAM Foundation Ltd., 3.0.1, Dec. 2015.

[11] T. Echekki and E. Mastorakos, “Turbulent Combustion: Concepts, Governing Equations and Modeling Strategies,” in *Turbulent Combustion Modeling: Advances, New Trends and Perspectives*, T. Echekki and E. Mastorakos, Eds. Dordrecht: Springer Netherlands, 2011, pp. 19–39.

[12] “One equation eddy-viscosity SGS model | Fumiya Nozaki’s CFD Blog.” .

[13] Y. Z. Li *et al.*, “Verification, validation and evaluation of FireFOAM as a tool for performance design,” 2017.

[14] “Open Source Fire Modeling,” *FM Global*. [Online]. Available: https://www.fmglobal.com/research-and-resources/research-and-testing/theoretical-computational-and-experimental-research/open-source-fire-modeling. [Accessed: 07-Mar-2019].

[15] “ANSYS FLUENT 12.0 Theory Guide - 5.3.2 Radiative Transfer Equation.” [Online]. Available: http://www.afs.enea.it/project/neptunius/docs/fluent/html/th/node111.htm. [Accessed: 22-Jul-2019].

[16] A. Vdovin and A. O. González, “Radiation heat transfer in OpenFOAM,” p. 17.

[17] “ANSYS FLUENT 12.0 Theory Guide - 5.3.6 Discrete Ordinates (DO) Radiation Model Theory.” [Online]. Available: http://www.afs.enea.it/project/neptunius/docs/fluent/html/th/node115.htm. [Accessed: 06-Mar-2019].

[18] “ANSYS FLUENT 12.0 Theory Guide - 5.3.3 P-1 Radiation Model Theory.” [Online]. Available: http://www.afs.enea.it/project/neptunius/docs/fluent/html/th/node112.htm. [Accessed: 06-Mar-2019].

[19] “Welcome to the www.salome-platform.org — SALOME Platform.” [Online]. Available: https://www.salome-platform.org/. [Accessed: 04-Mar-2019].
[20] C. Greenshields, “OpenFOAM v6 User Guide: 5.3 Mesh generation - blockMesh,” *CFD Direct*, 10-Jul-2018. [Online]. Available: https://cfd.direct/openfoam/user-guide/v6-blockmesh/. [Accessed: 07-Mar-2019].
[21] C. Greenshields, “OpenFOAM v6 User Guide: 5.4 Meshing with snappyHexMesh,” *CFD Direct*, 10-Jul-2018. [Online]. Available: https://cfd.direct/openfoam/user-guide/v6-snappyhexmesh/. [Accessed: 07-Mar-2019].
[22] “OpenFOAM Tutorial snappyHexMesh,” *Calum Douglas*. [Online]. Available: https://www.calum-douglas.com/openfoam-tutorial-snappyhexmesh/. [Accessed: 07-Mar-2019].
[23] “ParaView.” [Online]. Available: https://www.paraview.org/. [Accessed: 07-Mar-2019].
[24] K. B. McGrattan, R. J. McDermott, C. G. Weinschenk, and G. P. Forney, “Fire dynamics simulator, technical reference guide,” 2013.
[25] H. R. Baum and R. G. Rehm, “Calculations of three dimensional buoyant plumes in enclosures,” *Combust. Sci. Technol.*, vol. 40, no. 1–4, pp. 55–77, 1984.
[26] B. F. Magnussen, “The Eddy Dissipation Concept—A Bridge Between Science and Technology,” in *ECCOMAS thematic conference on computational combustion*, 2005, vol. 21, p. 24.
[27] G. P. Forney and K. B. McGrattan, *User’s Guide for Smokeview Version4: A Tool for Visualizing Fire Dynamics Simulation Data*. US Department of Commerce, National Institute of Standards and Technology, 2004.
[28] F. Larsson, J. Anderson, P. Andersson, and B.-E. Mellander, “Thermal Modelling of Cell-to-Cell Fire Propagation and Cascading Thermal Runaway Failure Effects for Lithium-Ion Battery Cells and Modules Using Fire Walls,” *J. Electrochem. Soc.*, vol. 163, no. 14, pp. A2854–A2865, Jan. 2016.
[29] A. Davidy, “CFD Simulation and Mitigation with Boiling Liquid Expanding Vapor Explosion (BLEVE) Caused by Jet Fire,” *ChemEngineering*, vol. 3, no. 1, p. 1, Mar. 2019.
[30] Yu Xiaojun and Jeffers Ann E., “Bridging Disparities in Scale across the Fire and Structural Domains,” *Struct. Congr. 2013*, pp. 2534–2544.
[31] S. Dutta, R. Das, and D. Bhattacharyya, “A multi-physics framework model towards coupled fire-structure interaction for Flax/PP composite beams,” *Compos. Part B Eng.*, vol. 157, pp. 207–218, Jan. 2019.
[32] S. Šulc, V. Šmilauer, and F. Wald, “VALIDATION OF LINKED SIMULATION FOR FIRE-EXPOSED CONCRETE STRUCTURES,” *Acta Polytech. CTU Proc.*, vol. 15, no. 0, pp. 120–125, Dec. 2018.