Direct Numerical Simulation of Droplet Evaporation in OpenFOAM

Lennart Moltrecht
Abstract

Keywords: Droplet Evaporation, Direct Numerical Simulation, OpenFOAM, Fuel Injection Sprays, Alternative Fuels

A model for the direct numerical simulation of evaporating droplets, based on a paper by Schlottke and Weigand (2008), is modified to work without explicit interface reconstruction (Piecewise Linear Interface Construction, PLIC). The evaporation rate is calculated using a hydrodynamic approach with prescribed saturated vapor at the interface. This model is then implemented into the open-source CFD tool OpenFOAM and validated against literature results and a simplified one-dimensional ETC (Effective Thermal Conductivity) model.

The direct numerical simulation resolves important effects such as internal recirculation, non-uniform surface temperature, temperature gradients within the droplet and the vapor layer around the droplet in high detail. Moreover, deformations and topology changes such as breakup can be captured using the volume-of-fluid (VOF) method for two-phase flows.
Contents

1 Introduction 1

2 Fundamentals 3
   2.1 Evaporation 3
      2.1.1 Heat Transfer 3
      2.1.2 Mass Transfer 4
   2.2 Phenomena inside the Droplet 5
      2.2.1 Heat Conduction 5
      2.2.2 Recirculation 6
   2.3 OpenFOAM 6

3 Modeling and Simulation 8
   3.1 Fluid Flow 8
   3.2 Multiphase-Modeling with Volume-of-Fluid (VOF) 9
   3.3 Evaporation Rate 10
      3.3.1 Interface Area Estimation 12
      3.3.2 Vapor Concentration Gradient Calculation 14
   3.4 Energy Equation 15
   3.5 Modification of Governing Equations 16
   3.6 Numerical Details 17

4 Validation and Results 19
   4.1 Validation 19
      4.1.1 Comparison to 1D MATLAB Model 19
      4.1.2 Comparison to Literature Data 21
   4.2 Results 23

5 Conclusion and Outlook 25
   5.1 Further Development and Improvement 25
1 Introduction

The cluster of excellence "Tailor-Made Fuels from Biomass" (TMFB) at RWTH Aachen University, founded in 2007, is working on the development of alternative fuels made from regrowing resources for the use in combustion engines. The goal is to create tailor-made biofuels that achieve high efficiency and low pollutant emissions in the combustion process (RWTH, 2016).

In internal combustion engines (ICEs) with internal mixture formation, such as Diesel engines, fuel is injected directly into the combustion chamber at the end of the compression stroke. The fuel jet breaks into ligaments and primary droplets (primary breakup), thereafter into smaller droplets (secondary breakup), forming a spray, as shown in Figure 1.1.

![Figure 1.1: Phenomena during fuel injection (Baumgarten, 2006)](image)

The spray consists of hundreds of thousands of small droplets that evaporate, turning the liquid spray into fuel vapor. For an efficient combustion with low pollutant emissions, it is important that air and fuel vapor form an homogeneous mixture filling the entire chamber.
The Institute of Heat and Mass Transfer (WSA) analyses fuel spray break-up and mixture formation experimentally and through numerical simulations. The ambient conditions of fuel injection (high pressure, high velocities, tiny droplets and short time frames) make it a challenging task to conduct experiments that can accurately measure droplet heating and evaporation. Simulations resolving all relevant phenomena can give scientists a detailed insight into droplet evaporation, which in turn helps to design fuels that fulfill the requirements of clean and efficient combustion.

In this project, a fully resolved direct numerical simulation (DNS) of droplet evaporation, based on a modeling approach published in a paper by Schlottke and Weigand (2008), is integrated into OpenFOAM and evaluated.
2 Fundamentals

Evaporation is the process of a substance changing its phase from liquid to gaseous state at temperatures below the boiling point. In contrast, boiling refers to a phase change from liquid to gas at the boiling point, and vaporization includes both evaporation and boiling.

A droplet is defined as a small drop of liquid surrounded by gas that may have arbitrary shape, but is often assumed to be spherical.

2.1 Evaporation

Evaporation is governed by heat and mass transfer across the interface between liquid and gaseous phases. Heat can be transferred into a droplet through conduction, convection and thermal radiation, and is transferred out through the vapor leaving the droplet (Figure 2.1). Thus, evaporation occurs only at the liquid-gas interface.

2.1.1 Heat Transfer

An energy balance for the change of internal energy $U$ of the droplet over time $t$ yields

$$\frac{dU}{dt} = Q_{\text{droplet}} = Q_{\text{heating}} + Q_{\text{evap}}$$

(2.1)
with energy transfer $\dot{Q}$ due to heating

$$\dot{Q}_{\text{heating}} = \dot{Q}_{\text{conv}} + \dot{Q}_{\text{cond}} + \dot{Q}_{\text{rad}}$$

(2.2)

and evaporative cooling

$$\dot{Q}_{\text{evap}} = -\dot{m}\Delta h_v$$

(2.3)

Evaporative cooling can be understood as the reduction in temperature due to the removal of latent heat $\Delta H_v = m\Delta h_v$ from the surface, with specific latent heat $\Delta h_v$.

The mass flow rate $\dot{m}$ due to phase change is described in the next section.

### 2.1.2 Mass Transfer

There are various models for evaporative mass transfer in literature, the most commonly used one for fuel droplets being a hydrodynamic model (Sazhin et al., 2014):

The mass transfer across the interface due to evaporation is governed by the actual phase change of liquid into vapor as well as the transport of vapor away from the droplet interface by diffusion. Assuming that evaporation occurs much faster than transport away from the interface, vapor at the droplet surface will accumulate and reach a saturated state. Thus, the rate of evaporation can be described by the rate of diffusion of vapor into the surrounding gas.
2.2 Phenomena inside the Droplet

In addition to the heat and mass transfer at the interface, as discussed in section 2.1, effects inside the droplet strongly influence temperature and evaporation rate and thus need to be considered.

2.2.1 Heat Conduction

Since heat is transferred from the hot surrounding gas into the liquid, a temperature gradient forms within the droplet from the hotter surface to the colder center (see Figure 2.2). Sazhin et al. (2006) pointed out that the consideration of a temperature gradient during fuel spray simulations has a significant impact on total ignition delay and leads to a decrease of evaporation time. Consequently, it is necessary to consider conductive heat transfer in the model.

Figure 2.2: Exemplary temperature distribution within water droplet (initial temperature 293 K) in hot air (inflow temperature 363 K).
2.2.2 Recirculation

A droplet moving through a gas is subject to viscous forces resulting in surface friction. Friction, a force applied to the liquid at its surface, causes fluid flow inside the droplet in the form of two vortex pairs, as shown in Figure 2.3. The induced velocity leads to an additional convective heat transfer and a faster distribution of heat from the surface to the center that needs to be considered for realistic results (Abramzon and Sirignano, 1989).

![Figure 2.3: Recirculation: Velocity field with vortices inside droplet (Schlottke and Weigand, 2008)](image)

2.3 OpenFOAM

OpenFOAM is an open source simulation platform for the solution of physical problems, including fluid flow, using one of the included computational fluid dynamics (CFD) solvers. Due to its open source structure, the existing solvers can be modified and extended by new models.
The high-level programming approach of OpenFOAM makes the addition of new equations into an existing solver straightforward. Solving the energy equation with constant heat capacity $c_p$:

$$\frac{\partial}{\partial t} (\rho c_p T) + \nabla \cdot (\rho c_p u T) = \nabla \cdot (\lambda \nabla T)$$

(2.4)

can be accomplished with the following code:

```cpp
solve
(
    fvm::ddt(rho, T)
    + fvm::div(rhoPhi, T)
    ==
    fvm::laplacian(lambda/Cp, T)
);
```

The `fvm` class contains differential operators for the Finite-Volume Method (FVM) and converts the equation into a linear system of equations of the form $Ax = b$. This system of equations is then solved using a user-defined method (e.g. Gauss-Seidel or Preconditioned Conjugate Gradients) with the command `solve`.

In this project, the solver `interMixingFoam` in OpenFOAM version 2.3 is used because of its ability to incorporate liquid and gaseous phases and the mixing of vapor and air:

"Solver for 3 incompressible fluids, two of which are miscible, using a VOF method to capture the interface" (CFD Direct Ltd, 2016)

Required modifications include the implementation of an energy equation, interface detection and interface area estimation as well as the computation of the evaporation rate and consequent addition of source terms to the existing equations.
3 Modeling and Simulation

3.1 Fluid Flow

Fluid flow is governed by the incompressible Navier-Stokes equations, namely the continuity and momentum equations. For a single-phase flow with constant density $\rho$ and viscosity $\nu$, they read

$$\nabla \cdot (\rho \mathbf{u}) = 0$$  \hspace{1cm} (3.1)

$$\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot (\mathbf{u} \times \mathbf{u}) - \nabla \cdot (\nu \nabla \mathbf{u}) = -\nabla p$$  \hspace{1cm} (3.2)

with velocity $\mathbf{u}$ and pressure $p$. The solution of these coupled equations received a lot of attention from researchers and is described in detail in literature. In OpenFOAM’s interMixingFoam, they are solved using the PIMPLE algorithm, a hybrid of SIMPLE (Semi-Implicit Method for Pressure-Linked Equations) and PISO (Pressure Implicit with Splitting of Operators).
3.2 Multiphase-Modeling with Volume-of-Fluid (VOF)

The Volume-of-Fluid method tracks the phase interface on a stationary mesh using a tracing variable \( \alpha \), defined as the fraction of traced volume \( V_i \) to cell volume \( V \):

\[
\alpha_i = \frac{V_i}{V} \tag{3.3}
\]

Cells without traced fluid have the value \( \alpha_i = 0 \), cells completely filled by the traced volume have \( \alpha_i = 1 \), and cells that contain the phase interface have \( 0 < \alpha_i < 1 \) (see Figure 3.1). A transport equation needs to be solved for each traced fluid based on the velocity field \( \mathbf{u} \) from the Navier-Stokes equations

\[
\frac{\partial \alpha_i}{\partial t} + \nabla \cdot (\mathbf{u} \alpha_i) = 0 \tag{3.4}
\]

with the constraint that the volume of all fluids in each cell is constant:

\[
\sum_i \alpha_i = 1 \tag{3.5}
\]

The diffuse representation of the interface in the \( \alpha \)-field, combined with numerical diffusion of the advection algorithm, could possibly lead to a smudged interface. To avoid this, compression schemes are implemented in OpenFOAM that minimize interface diffusion to a minimum.

The interMixingFoam solver traces three fluids 1, 2 and 3, whereof the second and third are miscible. For droplet evaporation, liquid and gas phases need to be traced, the gas phase consisting of both bulk gas (air) and vapor. Thus, phase 1 is considered to be liquid, phase 2 vapor and phase 3 bulk gas.

The Volume-of-Fluid representation of the phases does not directly contain the exact location of the interface. Because knowledge of the interface position and area are important to calculate the evaporation rate (see section 3.3), many authors including...
Schlottke and Weigand (2008) use an explicit interface reconstruction algorithm such as PLIC (Piecewise Linear Interface Reconstruction). Since explicit reconstruction is not available in OpenFOAM and the implementation of such an algorithm would go beyond the scope of this work, methods to calculate the required quantities directly from the \( \alpha \)-field have been developed and are described below.

### 3.3 Evaporation Rate

In the following, physical properties of the different fluids are denoted with subscripts \( l \) for liquid, \( g \) for gas and \( v \) for vapor, respectively.

The evaporation rate in hydrodynamic models (see subsection 2.1.2) is equal to the rate of diffusion of vapor from the surface into the surrounding gas and is thus...
governed by Fick’s law of diffusion. In Schlottke and Weigand (2008), the mass flow rate $\dot{m}$ due to evaporation is given as

$$\dot{m} = A_\Gamma \frac{D_{vg}\rho_{gp}}{1 - X_v} \langle \nabla X_v, \hat{n}_\Gamma \rangle$$

(3.6)

with interface area $A_\Gamma$, diffusion coefficient of vapor in bulk gas $D_{vg}$ and gas phase density $\rho_{gp}$. The evaporation rate is calculated for each cell using the local values of all variables, and is never negative ($\dot{m} \geq 0$, no condensation).

$X_v$ is the vapor mass fraction and can be calculated as (Schlottke and Weigand, 2008)

$$X_v = \alpha_2 \frac{\rho_v}{1 - \alpha_1 + \epsilon \rho_{gp}}$$

(3.7)

with gas phase density

$$\rho_{gp} = \rho_g (1 - \alpha_1 - \alpha_2) + \rho_v \alpha_2 + 0.5 \epsilon (\rho_v + \rho_g)$$

(3.8)

The equations above have been augmented with the $\epsilon$ variables, which are set to a very small value $VSMALL$ in OpenFOAM, to prevent division by zero in cells completely filled with liquid.

The local interface normal $\hat{n}_\Gamma$ can be obtained by normalizing the gradient of $\alpha_l$:

$$\hat{n}_\Gamma = \frac{\nabla \alpha_l}{|\nabla \alpha_l| + \epsilon_\Gamma}$$

(3.9)

$\epsilon_\Gamma$ is a mesh-dependent stabilization constant (interface.deltaN() in code, calculated using the average cell length) provided by OpenFOAM to prevent division by zero in cells where the magnitude of the gradient is zero.

The calculation of $A_\Gamma$ and the vapor concentration gradient $\nabla X_v$ is discussed in detail in the next sections.
3. Modeling and Simulation

Since the Volume-of-Fluid method is tracking the volume fraction $\alpha$, it is convenient to express the evaporation rate as a volumetric mass source

$$ \dot{m}_{\text{evap}} = \frac{\dot{m}}{V} \quad (3.10) $$

### 3.3.1 Interface Area Estimation

The local interface area $A_\Gamma$ in each cell needs to be known to compute Equation 3.6. Since an explicit interface reconstruction is not available, it needs to be estimated. Moreover, the estimation error should be low because errors in $A_\Gamma$ directly influence $\dot{m}$, and it should not be prone to numerical diffusion through a smudged $\alpha$-field to constrain evaporation to cells that actually contain the interface.

![Figure 3.2: Definition of face $f$, owner and neighbor cell, interface and cell face normal vectors](image)

The following algorithm turned out to deliver accurate and sharp results. Variables denoted with subscript $f$ are calculated on cell faces, others in the cell center:
3. Modeling and Simulation

The interface area on cell faces $A_{\Gamma,f}$ is computed as

$$A_{\Gamma,f} = \Delta\alpha_f(S_f, \hat{n}_{\Gamma,f})$$  \hspace{1cm} (3.11)

with $\alpha$ difference $\Delta\alpha_f = \alpha_{1,\text{nei}} - \alpha_{1,\text{own}}$ between neighbor (nei) and owner (own) cell of the face, face area and normal $S_f$ (defined as face normal times its area), and interface normal $\hat{n}_{\Gamma,f}$ as in Equation 3.9, but on faces.

This interface area then needs to be mapped back to the cell centers of both cells adjacent to the face, using the following interpolation:

$$A_{\Gamma,\text{nei}} = (1 - \omega)|A_{\Gamma,f}|$$  \hspace{1cm} (3.12)

$$A_{\Gamma,\text{own}} = \omega|A_{\Gamma,f}|$$  \hspace{1cm} (3.13)

with weight $\omega$

$$\omega = 0.5(1 - \text{erf}(\log(r)))$$  \hspace{1cm} (3.14)

and cell fill ratio $r$

$$r = \frac{|\alpha_{1,\text{nei}} - 0.5|}{|\alpha_{1,\text{own}} - 0.5| + \epsilon}$$  \hspace{1cm} (3.15)

Figure 3.3: Visualization of $\alpha_1$-field (left) and resulting sharp interface area mapped to cells (right).

The area estimation with weighted face-to-cell mapping results in a sharp interface detection (see Figure 3.3) with accurate area estimation even for coarse meshes. Tests in 2D and 3D showed that the estimated area of a sphere was close to the
analytical solution with errors around 1% for meshes that resolved the sphere with 20 × 20 × 20 cells (20 × 20 in the 2D case), and decreasing error for finer meshes.

3. Modeling and Simulation

3.3.2 Vapor Concentration Gradient Calculation

The assumption of hydrodynamic evaporation models is that vapor is always saturated at the interface (see subsection 2.1.2) and needs to be incorporated into the calculation of the vapor concentration gradient $\nabla X_v$.

As derived by Schlottke and Weigand (2008), the mass fraction of saturated vapor $X_{v,\text{sat}}$ can be calculated by

$$X_{v,\text{sat}} = x_{v,\text{sat}} \frac{M_v}{x_{v,\text{sat}} M_v + (1 - x_{v,\text{sat}}) M_g}$$

(3.16)

with molecular weights $M$ of vapor and bulk gas. The mole fraction of saturated vapor $x_{v,\text{sat}}$ is given by the relation

$$x_{v,\text{sat}} = \frac{p_{v,\text{sat}}(T)}{p_{\text{stat}}}$$

(3.17)

with saturation pressure $p_{v,\text{sat}}$ of vapor at each cell’s temperature $T$ and static pressure $p_{\text{stat}}$ of the fluid.

With the knowledge of $X_v$ (Equation 3.7) and $X_{v,\text{sat}}$, the gradient can be computed. Schlottke and Weigand (2008) provide an approach that is only feasible with an explicit interface reconstruction and thus not applicable here. Instead, the following relation is used:

$$\nabla X_v := \nabla (\alpha_1 X_{v,\text{sat}} + (1 - \alpha_1) X_v)$$

(3.18)

Cells completely filled with liquid are set to the mass fraction of saturated vapor, cells with no liquid are set to the actual mass fraction of vapor, and cells containing the interface ($0 < \alpha_1 < 1$) receive an interpolated value. The validation in subsection 4.1.1 shows that the gradient computation is accurate and leads to a
correct evaporation rate for a sharp interface representation in the $\alpha$-field. Should the interface be smudged, larger errors are to be expected.

### 3.4 Energy Equation

To model all heat-related effects described in chapter 2 such as heat transfer to the droplet, temperature gradients and convective heat transfer due to recirculation within the droplet, an energy equation needs to be added to the `interMixingFoam` solver.

A convective as well as conductive term is required. Moreover, a negative source term due to the enthalpy of vaporization needs to be added:

$$\frac{\partial}{\partial t} \left( \rho c_p T \right) + \nabla \cdot \left( \rho c_p u T \right) = \nabla \cdot (\lambda \nabla T) - \Delta h_v \dot{m}''. \quad (3.19)$$

Local fluid properties such as density $\rho$, viscosity $\nu$ and heat conductivity $\lambda$ are calculated using the one-field approach (only one property for all phases) and volume averaged, e.g.:

$$\rho = \sum_i \rho_i \alpha_i \quad (3.20)$$

Only the specific heat capacity $c_p$ is mass averaged:

$$c_p = \frac{\sum_i \rho_i \alpha_i c_{p,i}}{\sum_i \rho_i \alpha_i} \quad (3.21)$$

So far, all material properties are assumed to be constant. Temperature-dependent material properties could be added in the future for improved results.
3. Modeling and Simulation

3.5 Modification of Governing Equations

The standard equations of the `interMixingFoam` solver need to be augmented for the effects of evaporation on the flow field. Due to the changes made to the original solver and the incorporation of evaporation effects, hereafter the solver is called `mixingEvapFoam`.

**Continuity Equation** Due to phase change, the flow is no longer divergence-free. Evaporation causes a volume source at the interface due to the difference in density between liquid and vapor. A simple implementation is to add a source term to the continuity equation accounting for that volume change:

\[
\nabla \mathbf{u} = -\dot{m}'\left(\frac{1}{\rho_v} - \frac{1}{\rho_l}\right)
\]

As Schlottke and Weigand (2008) point out, this formulation is not accurate for a velocity field \( \mathbf{u} \) of the VOF method because \( \mathbf{u} \) is mass-averaged and not volume conservative. A more advanced computation of the divergence including a virtual velocity field and error redistribution is detailed in their paper. In this project, however, the simple implementation in Equation 3.22 is used, with the possibility of replacing it with the formulation from Schlottke and Weigand (2008).

**Volume-of-Fluid Equations** The phase change leads to a sink in the liquid advection equation and a source in the vapor advection equation:

\[
\frac{\partial \alpha_l}{\partial t} + \nabla \cdot (\mathbf{u} \alpha_l) = -\frac{\dot{m}''}{\rho_l}
\]

\[
\frac{\partial \alpha_v}{\partial t} + \nabla \cdot (\mathbf{u} \alpha_v) = +\frac{\dot{m}''}{\rho_v}
\]

Tests have shown that the MULES limiter, a flux limiter in `interFoam` solvers that prevents over- and undershoots of the \( \alpha \)-variables by limiting fluxes, can cause problems near the interface and lead to a smudged \( \alpha \)-field. Disabling the limiter and using
an upwind scheme for the $\alpha$-equations (instead of the vanLeer scheme suggested by OpenFOAM) led to a sharp interface and no over- or undershoots.

**Energy Equation**  A heat sink has already been added to the energy equation in Equation 3.19.

### 3.6 Numerical Details

The solver has been developed for a Cartesian grid and been tested on ordered equidistant grids in 2D and 3D. Potentially, the generality of OpenFOAM’s discretization operators should ensure functionality on arbitrary grids with polyhedral cells and local mesh refinement, although this has not been tested.

Scalar values ($\alpha, \rho, T, \ldots$) are stored at cell centers, and the velocity $u$ is represented by the flow through cell faces $\phi$ ($\phi = \langle u, S_f \rangle$) for incompressible flows, with face area $S_f$. After solving for $\phi$, the velocity field $u$ is reconstructed and stored in cell centers for post-processing.

Numerical schemes are mostly left unchanged from the OpenFOAM examples for *interFoam* solvers: *Implicit Euler* time stepping scheme, *Gauss linear* gradient scheme, *Gauss linear corrected* laplacian scheme, as well as the following divergence schemes:

```
divSchemes
{
    div(rhoPhi ,U) Gauss limitedLinearV 1;
    div(phi ,alpha) Gauss vanLeer;
    div(phirb ,alpha) Gauss upwind;
    div((muEff*dev(T(grad(U)))) div(rhoPhi ,T) Gauss linear;
    Gauss upwind;
}
```
The time step is automatically adapted for a target Courant number of 0.5.
4 Validation and Results

For validation, a test case was set up as shown in Figure 4.1. Dry air ($\alpha_{v,in} = 0$) flows into the domain from the left side with constant velocity $u_{in}$ and temperature $T_{in} > T_0$. An initially spherical droplet with radius $R_0$ and initial temperature $T_0$ is placed stationary ($u = 0$) near the inlet. The distance to the outlet is large enough to allow for the formation of a homogeneous flow field.

Figure 4.1: Domain setup for test case, similar to Schlottke and Weigand (2008)

4.1 Validation

4.1.1 Comparison to 1D MATLAB Model

In this institute, an in-house MATLAB implementation for the evaporation of droplets in sprays, based on a paper by Elwardany et al. (2011), exists. The model assumes
spherical symmetry of the droplet and solves a one-dimensional heat conduction equation (see Moltrecht (2015) for details). Internal recirculation is not resolved but accounted for using an effective thermal conductivity (ETC) model. The ambient conditions of the flow (velocity, temperature, vapor concentration) are assumed to be constant, and correlations are used to calculate heat and mass transfer.

The domain for OpenFOAM DNS (Figure 4.1) is set to $150 \mu m \times 70 \mu m \times 70 \mu m$ and discretized with $215 \times 100 \times 100$ cells. The droplet consists of water ($R_0 = 5 \mu m, T_0 = 293.15 K$) and is surrounded by dry air ($T_{in} = 363.15 K, u_{x,in} = 10 m/s, p = 100 kPa$). The Courant number limitation ($Co < 0.5$) resulted in a $\Delta t$ of approximately $2.5 \times 10^{-8} \text{s}$.

Evaporation rate and surface temperature results of the OpenFOAM simulation using mixingEvapFoam are compared to MATLAB results with the same physical setup in Figure 4.2.

Figure 4.2: Evaporation rate and surface temperature results over time from OpenFOAM and MATLAB

The evaporation rate predicted by OpenFOAM in Figure 4.2a exhibits what seems like a singularity at $t = 0$, but quickly normalizes and converges against the MATLAB solution. The singularity is caused by the initial condition of dry air, leading to a very large concentration gradient $\nabla X_v$ and thus causing a very high evaporation rate. After a few time steps, a vapor layer forms around the droplet, leading to realistic concentration gradients and a reasonable evaporation rate. Both plot lines should not be expected to overlap perfectly due to the very different nature of the
models (one-dimensional versus 3D fully resolved). Nevertheless, both models predict evaporation rates in the same order of magnitude, showing that the approach for the calculation of $A_T$ and $\nabla X_v$ without explicit interface reconstruction yields feasible results.

The surface temperatures in Figure 4.2b demonstrate very different behavior. While having a similar shape (faster heating in the first $5\mu s$, thereafter flatter slope), the OpenFOAM model strongly overestimates the increase in temperature. The reasons for this have not been found until the time of writing.

### 4.1.2 Comparison to Literature Data

For a comparison to the results produced by Schlottke and Weigand (2008), a test case similar to the one in the paper has been set up: The domain (Figure 4.1) is set to $35\,\text{mm} \times 15\,\text{mm} \times 15\,\text{mm}$ and discretized with $230 \times 100 \times 100$ cells. The droplet consists of water ($R_0 = 1.15\,\text{mm}, T_0 = 343.15\,\text{K}$) and is surrounded by dry air ($T_{in} = 363.15\,\text{K}, u_{x,in} = 15\,\text{m/s}, p = 100\,\text{kPa}$). The Courant number limitation ($\text{Co} < 0.5$) resulted in a $\Delta t$ of approximately $2 \times 10^{-6}\,\text{s}$, leading to a total computation time of approximately 60 hours on 4 cores of the WSA in-house cluster.

![Figure 4.3: Vapor distribution comparison between Schlottke and Weigand (2008) and evapMixingFoam](image)
4. Validation and Results

Qualitative results are shown in Figure 4.3 and Figure 4.4. The droplet deforms from its initially spherical shape due to the sudden flow of air, forming turbulent vortices in the wake of the droplet. An exact reproduction of the turbulent flow field is not achieved since no turbulence model is used in \texttt{evapMixingFoam}, and the grid is not fine enough to fully resolve turbulent effects. Enabling a turbulence model is straight-forward in OpenFOAM, in case this is desired for the engineering application.

![Figure 4.3](image1.png) ![Figure 4.4](image2.png)

(a) Schlottke and Weigand (2008)  (b) \texttt{evapMixingFoam}

**Figure 4.4:** Temperature distribution comparison between Schlottke and Weigand (2008) and \texttt{evapMixingFoam}

The formation of a vapor layer, with the highest concentration near the surface (as expected), can be seen in Figure 4.3a. Moreover, the vapor concentration in the wake is higher due to the relatively low velocity and recirculation of gas (indicated by arrows).

The temperature distribution also shows similarities, with a strong temperature gradient (from 363 K to 343 K) near the droplet surface, and colder gas in the wake.
4. Validation and Results

4.2 Results

Since physical data is available at all mesh points, inside gas, liquid as well as at the interface, detailed plots can be created. As an example, turbulence in the wake of the droplet is showcased in Figure 4.5, and surface temperature distribution is showcased in Figure 4.6.

![Figure 4.5: Turbulent flow field around droplet in the second validation case.](image)

The flow velocity around the droplet is not uniform and drops almost to zero in the wake of the droplet. Moreover, the velocity near the perimeter is nearly 20% higher than in the freestream.

The surface temperature also shows high variations around the droplet, the hottest spots being near the wake where velocities are low and vapor concentration is high, leading to a lower evaporation rate and thus less cooling through evaporation.
4. Validation and Results

Figure 4.6: Surface temperature distribution in the second validation case.
5 Conclusion and Outlook

In contrast to simplified models, the direct numerical simulation of droplet evaporation offers a high level of detail for all relevant engineering quantities. The solver has been tested and is able to resolve all desired physical effects such as deformation, recirculation, temperature distribution and local evaporation rates. Quantitative correctness has yet to be shown through comparisons to simple cases where analytical solutions exist. Moreover, a validation using measurements from experiments is desirable.

5.1 Further Development and Improvement

Certain improvements should be made to the solver before declaring it ready-for-use, in order to assert the inclusion of all physical effects as well as numerical stability, reliability and execution speed.

As mentioned in subsection 4.1.1 and Figure 4.2b, the correctness of the energy equation should be assessed and modified, if necessary, to minimize the over-prediction of the temperature increase compared to MATLAB.

Other improvements include the addition of temperature-dependent material properties, the possibility to use a dynamic mesh (DyM in OpenFOAM) to refine cells near the interface. A refined interface would allow a sharper representation of the droplet surface, reducing errors due to interface smudging, and possibly a lower total cell count (coarser mesh far from the droplet).
5. Conclusion and Outlook

From a numerical point of view, the calculation of the interface area and evaporation rate should be restricted only to cells that are close to the interface in order to improve the speed of execution. Adapting convergence thresholds of the solvers for each equation to be solved as well as the PIMPLE loop controls could lead to a further speed-up.
Bibliography


http://cfd.direct/openfoam/user-guide/standard-solvers/


http://dx.doi.org/10.1016/j.fuel.2013.07.031
