NTNU – Trondheim
Norwegian University of
Science and Technology

Evaluation of Gas and Oil Dispersion during Subsea Blowouts

Mikkel Bakli

Master of Energy and Environmental Engineering
Submission date: June 2014
Supervisor: Iver Håkon Brevik, EPT
Co-supervisor: Jan Erik Olsen, SINTEF

Norwegian University of Science and Technology
Department of Energy and Process Engineering

NTNU

Norwegian University
of Science and Technology

Department of Energy
and Process Engineering

EPT-M-2014-6

MASTER THESIS

for

Student Mikkel Bakli

Spring 2014

English title: Evaluation of gas and oil dispersion for subsea blowouts.

Norwegian title: Evaluering av gass- og oljedispersjon for utblåsninger under vann.

Background and objective:

Several incidents of subsea gas release in the past years have underlined the need for improved knowledge about subsea release of hydrocarbons. Subsea release of oil and gas poses a threat to the safety operations, integrity of assets and safety of third parties operating offshore. As the number of subsea installations and pipelines increases, the risk of potential faults will also increase. In order to perform risk assessments it is important to understand the qualitative behavior and to make reliable quantitative estimates of how the oil and gas will surface. Since quantitative descriptions of subsea gas releases do not exists and performing realistic experiments offshore would be prohibitively expensive, quantitative models have been identified as interesting research tools.

An evaluation of the gas and oil dispersion from the blowout is therefore important for the safety of the operation. The hazardous area is dependent on fluid composition of the blowing well, blowout rates, tidal currents and weather conditions.
As buoyancy is much stronger for gas than for oil, the gas plume will initially be important also for the oil transport. With the presence of either tidal or wind induced current, the gas and oil plumes may separate.
In the present project, gas and oil dispersion is suggested to be evaluated by using the discrete particle multiphase model in the computational fluid dynamic (CFD) code ANSYS FLUENT.

The following tasks are to be considered:

1. The main task is to study a deep water oil/gas plume, where the model will be incorporated in the presence of sea currents. Also, one wishes to include the density changes in the bubbles, their size variation, and gas dissolution.

2. We will start with a gas plume for which some experimental data and validated integral models or CFD models exist that can be used for comparison. Initially the sea surface will be treated as a free slip boundary where gas bubbles are allowed to escape. A simple model for the gas bubble size that takes into account for bubble expansion should be evaluated.

3. In the case of combined gas and oil plumes, the difference in boundary conditions at the sea surface must be handled. The gas bubbles should be allowed to escape while oil bubbles must stay in the water column. Boundary conditions due to sea currents are expected to be challenging.

-- " --

Within 14 days of receiving the written text on the master thesis, the candidate shall submit a research plan for his project to the department.

When the thesis is evaluated, emphasis is put on processing of the results, and that they are presented in tabular and/or graphic form in a clear manner, and that they are analyzed carefully.

The thesis should be formulated as a research report with summary both in English and Norwegian, conclusion, literature references, table of contents etc. During the preparation of the text, the candidate should make an effort to produce a well-structured and easily readable report. In order to ease the evaluation of the thesis, it is important that the cross-references are correct. In the making of the report, strong emphasis should be placed on both a thorough discussion of the results and an orderly presentation.

The candidate is requested to initiate and keep close contact with his/her academic supervisor(s) throughout the working period. The candidate must follow the rules and regulations of NTNU as well as passive directions given by the Department of Energy and Process Engineering.

Risk assessment of the candidate's work shall be carried out according to the department's procedures. The risk assessment must be documented and included as part of the final report. Events related to the candidate's work adversely affecting the health, safety or security, must be documented and included as part of the final report. If the documentation on risk assessment represents a large number of pages, the full version is to be submitted electronically to the supervisor and an excerpt is included in the report.

Pursuant to "Regulations concerning the supplementary provisions to the technology study program/Master of Science" at NTNU §20, the Department reserves the permission to utilize all the results and data for teaching and research purposes as well as in future publications.

The final report is to be submitted digitally in DAIM. An executive summary of the thesis including title, student's name, supervisor's name, year, department name, and NTNU's logo and name, shall be submitted to the department as a separate pdf file. Based on an agreement with the supervisor, the final report and other material and documents may be given to the supervisor in digital format.

☐ Work to be done in lab (Water power lab, Fluids engineering lab, Thermal engineering lab)
☐ Field work

Department of Energy and Process Engineering, 14. January 2014

Olav Bolland
Department Head

Iver H. Brevik
Academic Supervisor

Research Advisor: Jan Erik Olsen (SINTEF Materials and Chemistry)

Abstract

The global demand for hydrocarbons is high and is also believed to be high in the future. Much of today's oil and gas exploration is carried out offshore and consequently, there is a risk if oil and gas blowouts at the seabed. Major concerns from a subsea oil and gas release are fire and toxic hazard to people working on offshore installations and loss of buoyancy of ships and floating installations. In addition, oil spills will result in both immediate and long-term environmental damage. Risk assessments are a very useful tool to pinpoint the risks of offshore oil and gas exploration and production. In terms of blowouts, these assessments require knowledge of the qualitative behavior and reliable quantitative estimates for where and when the oil and gas will surface. Since execution of underwater test releases of hydrocarbons is extremely costly, computer models are interesting research subjects.

In this thesis, a simulation concept for forecasting oil and gas blowouts is presented. ANSYS FLUENT 15.0.0, a commercial Computational Fluid Dynamics (CFD) package, is used to obtain both the qualitative behavior and the quantitative estimates. The model accounts for variation in bubble size and bubble density. In addition the model allows for the presence of ocean currents and gas dissolution. The released oil droplets and the natural gas bubbles are tracked while they rise towards the ocean surface in order to estimate the effect of ambient ocean currents. The general model set-up is first validated against experimental data, for which air-bubbles are released in a 7 m deep basin.

The primary simulations are based upon a field experiment conducted in Norwegian waters during June 2000, known as DeepSpill. Four discharges of oil and gas from a water depth of 844 m was carried out under controlled circumstances. Extensive observations and documentation were acquired during the experiments, in addition chemical and biological samples were collected along the water column. In the present work, simulation results are presented, discussed and compared with chosen field data obtained from the DeepSpill experiment.

The overall simulation results are found to correspond quite good with the results from the DeepSpill experiment. The mean path of oil corresponds favorably with the overall shape of the echo-sound images taken during the experiments. The point of complete gas dissolution is found to match the field data, as long as a mass transfer reduction factor is employed. However, the rise time of oil droplets are somewhat over-predicted, which may indicate a need for denser grid in the release zone and/or a reconsideration of the oil droplet size distribution.

Sammendrag

Etterspørselen etter hydrokarboner er høy, og den er også antatt å forbli høy i framtiden. Mye av dagens letevirksomhet etter nye utvinnbare forekomster forgår på havbunnen. Utblåsninger (enten fra havbunnen eller utstyr) vil derfor representere en risiko i lete-, produksjon- og nedstengningsfasen. Utblåsninger av olje og gass vil kunne få store konsekvenser for personsikkerhet og miljøet. De vil også gi betydelig økonomiske tap for involverte parter.

Som verktøy for å håndtere risikoen ved undervanns olje- og gassvirksomhet benyttes det risikovurderinger. Disse krever kunnskap om konsekvensene og sannsynligheten for at en hendelse vil skje. Denne avhandlingen tar for seg noe av konsekvensdelen ved en undervanns flerfase utblåsning. Ettersom realistiske testutslipp er ekstremt kostbart vil datamodeller være hensiktsmessige forskningsverktøy.

Avhandlingen presentere et konsept for å simulere oppførselen og hva som vil skje med en blanding av olje og gass som slippes ut på havbunnen på vei mot overflaten.

Programpakken ANSYS FLUENT 15.0.0, en kommersiell Computational Fluid Dynamics (CFD) pakke, blir brukt til å simulere den kvalitative oppførselen og de kvantitative beregningene. Modellen tar for seg variasjon i boblestørrelse og bobletetthet. I tillegg tar modellen høyde for tilstedeværelsen av havstrømmer og gassoppløsning. Oljedråpene og naturgassboblene spores mens de stiger mot overflaten for å se effekten av havstrømmene. Det generelle modelloppsettet er først validert mot eksperimentelle data, hvor luft blir sluppet ut på bunnen av et 7 meters dypt basseng.

Hovedmodellen er basert på et feltforsøk gjennomført i norske farvann i juni 2000, kjent som DeepSpill. Fire utslipp av olje og gass fra 844 meters dybde ble gjort under kontrollerte forhold. Omfattende observasjoner og målinger ble gjort, også av kjemiske og biologiske forhold.

I det foreliggende arbeidet blir simuleringsresultatene presentert, diskutert og sammenlignet med aktuelle feltdata innhentet fra DeepSpill eksperimentet. Simuleringsresultatene sett under ett stemmer godt overens med resultatene fra DeepSpill eksperimentet. «Stigebanen» til oljedråpene har god overensstemmelse med ekko-lyd signalene målt under eksperimentet. Punktet hvor gassplumen er fullstendig oppløst i sjøvannet passer godt med feltdataene, så lenge en reduksjonsfaktor er anvendt. Stigetiden til oljen er noe overestimert, noe som indikerer et behov for finere mesh rundt utslippssonen og/eller grundigere evaluering av oljedråpe størrelsen.

Preface

Background for Thesis and Organization

This thesis is submitted in partial fulfillment of the requirements set for obtain a Master of Science and Technology, with a specialization within fluid mechanics at the Norwegian University of Science and Technology (NTNU). It contains work carried out from January 2014 to June 2014, and is awarded 30 credits. The time is spent at literature reviews, developing skills related CFD simulations and C Programming, and formulation of this thesis.

The work is performed in collaboration with SINTEF, Trondheim and Acona Flow Technology in Skien.

Organization

The thesis is organized into nine chapters and two appendixes.

CHAPTER ONE: INTRODUCTION *deals with motivation and background for the studying of underwater hydrocarbon releases. In this chapter the overall objective and scope of this thesis is presented.*

CHAPTER TWO: BASICS OF UNDERWATER BLOWOUTS *describes the main physics of deep water oil and gas blowouts. A short overview of the history of plume modeling is presented, together with a brief introduction of integral models.*

CHAPTER THREE: BASIC CDF AND GOVERNING MODELS *describes the principle behind Computational Fluid Dynamics (CFD). The chapter includes a detailed derivation of the turbulence and multiphase models used in the thesis.*

CHAPTER FOUR: USER-DEFINED MODELING *deals with the theory behind the user-defined functions (UDFs) implemented to complement the standard CFD code. These functions are required to achieve a desired plume behavior.*

CHAPTER FIVE: NUMERICAL ACCURACY AND COMPUTATIONAL TIME *deals with the balance between computational time and numerical accuracy. Different methods of mesh generation is presented and discussed, in addition to solver settings.*

CHAPTER SIX: DISCUSSION OF ASSUMPTIONS *presents assumptions of the user-defined functions and the physical models used.*

CHAPTER SEVEN: VALIDATION MODEL: ROTVOLL EXPERIMENT *compares the simulation results with the observations and measurements from the Rotvoll experiment.*

CHAPTER EIGHT: PRIMARY MODEL: DEEPSPILL EXPERIMENT *compares the simulation results with the observations and measurements from the DeepSpill experiment, as well as modeling results provided by the DeepBlow model.*

CHAPTER NINE: CONCLUSION REMARKS *summarizes the findings in the thesis.*

APPENDIX A: *Presents the UDFs used in simulation of the validation model.*

APPENDIX B: *Presents the UDFs used in simulation of the primary model.*

Acknowledgement

I would like to thank Professor Iver H. Brevik for being my supervisor at NTNU, and for always being helpful and available for discussions.

Outside of NTNU, I would like to thank Senior Scientist Jan Erik Olsen at SINTEF for excellent advices and inspirational discussions. It has been extremely helpful with his thorough insight along the way. I would also like to thank Trygve Rinde and Tron Solberg at Acona Flow Technology AS for introducing me to CFD and for letting me write this thesis.

Finally, I would like to thank Senior Technical Consultant Love Håkansson at EDRMedeso for helping out with difficulties regarding UDFs.

Trondheim, 23/06 - 2014

Mikkel Bakli

Table of Contents

List of Figures

Chapter 2: Deep Water Blowouts

Chapter 3: Basic CFD and Governing Models

Chapter 4: User-Defined Modeling

Chapter 5: Numerical Accuracy and Computational Time

Chapter 6: Discussion of Assumptions

Chapter 7: Validation Model: Rotvoll Experiment

Chapter 8: Primary Model:DeepSpill Experiment

List of Tables

Chapter 7: Validation Model: Rotvoll Experiment

Chapter 8: Primary Model:DeepSpill Experiment

List of Symbols

σ	Surface tension (N/m)	g	Gravitational acceleration (m/s^2)
σ	Prandtl number	H_p	Depth particle (m)
ε	Turbulence kinetic energy dissipation rate (m^2/s^3)		
μ	Dynamic viscosity (Pa.s)	H_z	Water depth (m)
ρ	Density (kg/m^3)	I	Turbulent intensity
λ	Ratio	J	Mass flux
ζ	Random number	k	Turbulent kinetic energy (m^2/s^3)
$\emptyset$	Flow variable	k	Mass transfer coefficient (m/s)
ω	Spesific dissipation rate (s^{-1})	L	Length scale (m)
δ	Kronecker delta	M	Mass (kg)
φ	Shape coefficient	M	Molar mass (kg/mol)
α	Volume fraction	m	Mass flow rate (kg/s)
α	Entrainment coefficient	N	Ensemble size
τ	Time scale	N	Number of
$\bar{\bar{\tau}}$	Stress tensor	n	Spread parameter
A	Area (m^2)	n	Solubility (kg/m^3)
b	Width of buoyancy profile (m)	P	Pressure (Pa)
D	Diffusivity (m^2/s)	P	Production term
d	Diameter (m)	p	Pressure (Pa)
$\bar{d}$	Mean diameter (m)	Pe	Peclet number
d_b^{eq}	Equilibrium diameter (m)	q	q'th phase
E_0	Eotvos number	R	Individual gas constant
F	Force (N)	r	Radius (m)
G	Generation term	Re	Reynolds number

S	*Source term*	u'	*Velocity fluctuation x-direction (m/s)*
T	*Integral time (s)*	$\bar{u}$	*Mean velocity (m/s)*
T	*Temperature (K)*	V	*Volume (m^3)*
t	*Time (s)*	v	*Velocity y-direction (m/s)*
u	*Velocity x-direction (m/s)*	w	*Velocity z-direction (m/s)*

Acronyms

CFD	*Computational fluid dynamics*	*RKE*	*Realizable k-ε*
DNS	*Direct numerical simulation*	*RSM*	*Reynolds stress model*
DPM	*Discrete phase model*	*RWM*	*Random walk model*
DRW	*Discrete random walk*	*SKE*	*Standard k-ε*
LES	*Large eddy simulation*	*UDF*	*User-defined function*
RANS	*Reynolds-Averaged Navier-Stokes*	*VOF*	*Volume of fluid*

Chapter 1

Introduction

1.1 Motivation

With increasing subsea activities plumes have acquired additional relevance from a risk assessment point of view. The global demand for hydrocarbons is still significant, which has led to exploration into deeper water and more hazardous projects. Underwater releases of oil and natural gas, resulting from accidents in offshore drilling (blowouts), subsea installations or broken gas pipelines, represents a potential danger for ships, offshore structures, and may have major environmental and safety impacts. For this purpose, it is important to know as much as possible about the dynamics of underwater releases of oil and natural gas.

As the number of offshore installations is growing, the probability of potential faults is increasing. Therefore, risk assessments are necessary to ensure safe offshore operations. These assessments require knowledge of the probability and the consequence for a certain incident. In terms of blowouts, this will include knowledge of the qualitative behavior and reliable quantitative estimates for when, where and how much oil and gas that potentially can surface. Since execution of realistic blowout experiments are extremely costly, computer simulations are interesting research subjects.

Many regions around the world have active offshore production or will develop. Examples are Brazil, North Seas, West Africa, and the USA including the Gulf of Mexico. The offshore production in USA accounts for about 30 percent of the total domestic production. In addition, China and Japan are working on their own deep water exploration programs that have found promising deposits (35). As part of deep water exploration programs there should always be an assessment of the risk and consequences of a blowout. Computer simulations can provide information about impact assessment and form the basis for emergency response.

In June 2000, four discharges of oil and gas from a water depth of 844 m was carried out under controlled circumstances in Norwegian waters. The experiment was named "DeepSpill", and the main objective was to obtain data for verification and testing of numerical models for simulating/modeling accidental releases in deep water (34). Few years before the DeepSpill experiment was conducted, an integral model, named "DeepBlow", was developed with the purpose of recreating experimental data obtained from blowout scenarios (15). The model did a reasonable job of predicting the time to surface and the

location of the slick through some tuning of different parameters (37). Historically, classical integral models offer an efficient and good representation of a rising plume. However, these models provide limited information on the surface effects and rely heavily on appropriate empirical data. Computational Fluid Dynamics (CFD) is more flexible and fundamental, and makes it possible to provide information on both the multi-phase plume and the surface interactions.

No literature has been found on development of CFD model simulation of underwater releases including both oil and gas. However, in recent time there has been quite high activity in CFD simulations of underwater gas releases, such as Cloete (5), Cloete et al. (11), Skjetne & Olsen (2), and Pan (47).

1.2 Thesis Objective

The main objective of this thesis is to provide a general framework for CFD simulation of a deep water oil and gas release. The fundamental theoretical framework and model set-up is based on work developed by Cloete (5) ,Cloete et al. (11), Skjetne & Olsen (2), and generalized to allow for presence of oil droplets and plume tracking, underwater currents and gas dissolution. A detailed presentation of the general model set-up and theoretical framework is presented and discussed, in addition to some alternative models/methods. The general model set-up is first validated against experimental data of Engebretsen et al. (54), before simulation results are presented, discussed, and compared with chosen experimental data obtained from the report of Johansen et al. (34) and relevant modeling results provided by the DeepBlow model.

1.3 Scope

In deep water blowouts the volume of natural gas may be depleted through dissolution into the sea water and the driving buoyancy of a rising gas bubble plume may be completely lost. In case of complete gas dissolution, the oil droplets will eventually rise slowly alone due to buoyancy, without any assistance of a more buoyant gas (as was the case in the DeepSpill experiment). However, the mean cloud path of oil droplets is strongly affected by ocean currents during their rise. This may result in separation of oil droplets from the more buoyant gas bubbles, before the gas is completely dissolved (1). In this context, the following bullet points are to be considered:

- *Mean cloud path of oil.*
- *Point of complete gas dissolution.*
- *Rise time of oil.*

These three points will provide valuable information in any blowout scenarios. Even though the probability of such accident is low thanks to today's technology, the oil and gas industry has to be prepared.

- ***Mean cloud path of oil:*** *In order to start the clean-up work as early as possible, the oil and gas industry needs to know where the oil is expected to surface.*
- ***Point of complete gas dissolution:*** *In deep water, gas bubbles may be "trapped" below the ocean surface due to effects of dissolution. This is valuable information for oil and gas operators as it concern surface restrictions governing surface vessel activity and possible evacuation of offshore structures.*
- ***Rise time of oil:*** *Rise time is the time taken for the first oil droplets to surface. The computed rise time is an indication of the model validity, when compared with empirical data.*

ANSYS FLUENT 15.0.0, a commercial CFD package, is used to provide both the qualitative behavior and the quantitative estimates. The Discrete Phase Model (DPM) is applied for simulation of the gas bubbles and the oil droplets, which is a parcel based Lagrangian method. One parcel consists of multiple bubbles with same velocity and density. The interaction between the sea water and atmospheric air is captured by a sharp interface tracking scheme implemented in the Volume of Fluid (VOF) model, which is based on the Eulerian formulation.

Chapter 2

Deep Water Blowouts

Subsea blowouts generally involve oil and/or natural gas. The volume ratio of these two fluids is dependent of the characteristics of the fluids and the producing reservoir. The natural gas is strongly affected by buoyancy, which usually provides the driving force for an uncontrolled blowout. This buoyancy force may increase with gas flow rate and reservoir pressure, while decrease with depth due to compression. Subsea release of hydrocarbons is often referred to as either shallow or deep water blowouts (1). In deep water, the effect of gas dissolution and ocean currents is more dominant due to longer residence time (i.e. the time a bubble/droplet has spent in the surrounding water). Throughout this thesis, oil droplets and gas bubbles are often referred to as a dispersed phase, while water and atmospheric air is referred as a continuous phase.

2.1 Deep Water Blowouts – Hydrate Formation

Deep water blowouts differ mainly from shallow water blowouts by oil and natural gas exiting from the seabed release point into high water pressure and low temperature. Multiple literature sources assert that the natural gas quickly combines with cold water and form solid ice-like substance known as hydrates. This usually occurs at depths below 300-700 meters (1; 36; 38; 39), depending on the ocean properties. These assumptions of hydrate formation are based upon hydrate phase diagrams and simulated deep sea environment. However, in the DeepSpill experiment, no gas hydrates were observed even though thermodynamic equilibrium suggested they should. Moreover, the DeepBlow model predicted nearly identical plume trap height, with and without equations related hydrate formation (37). This lack of hydrate formation is briefly explained by Johansen (56): "This lack of hydrate formation is most easily explained by the absence of the dissolved gas saturation condition necessary for sustained hydrate formation." Based on these observations and modeling results, hydrate formation are neglected in the current application.

2.2 Basics of Underwater Blowouts

Without the presence of hydrate formation, ocean currents, and gas dissolution, deep water blowouts has mainly the same dynamics as shallow water blowouts, which is explained below.

Underwater oil and gas blowouts will pass through three zones of interest as oil and gas move towards the sea surface (ref. figure 1). The high velocity at the well head exit generates a jet zone, known as the zone of flow establishment. This zone is dominated by initial momentum of oil and gas, gas expansion, and breakup into gas bubbles (22). In addition, the high degree of turbulence in this region is responsible for the fragmentation of oil into droplets (41). A drag force is exerted from the ambient fluid and a net momentum transfer from the dispersed phase to the ambient water occurs. This effect creates a motion of water alongside the bubbles/droplets, a motion denoted as entrainment of liquid fragments (23). The effect of entrained water causes a rapid loss of momentum a few meters from the discharge location.

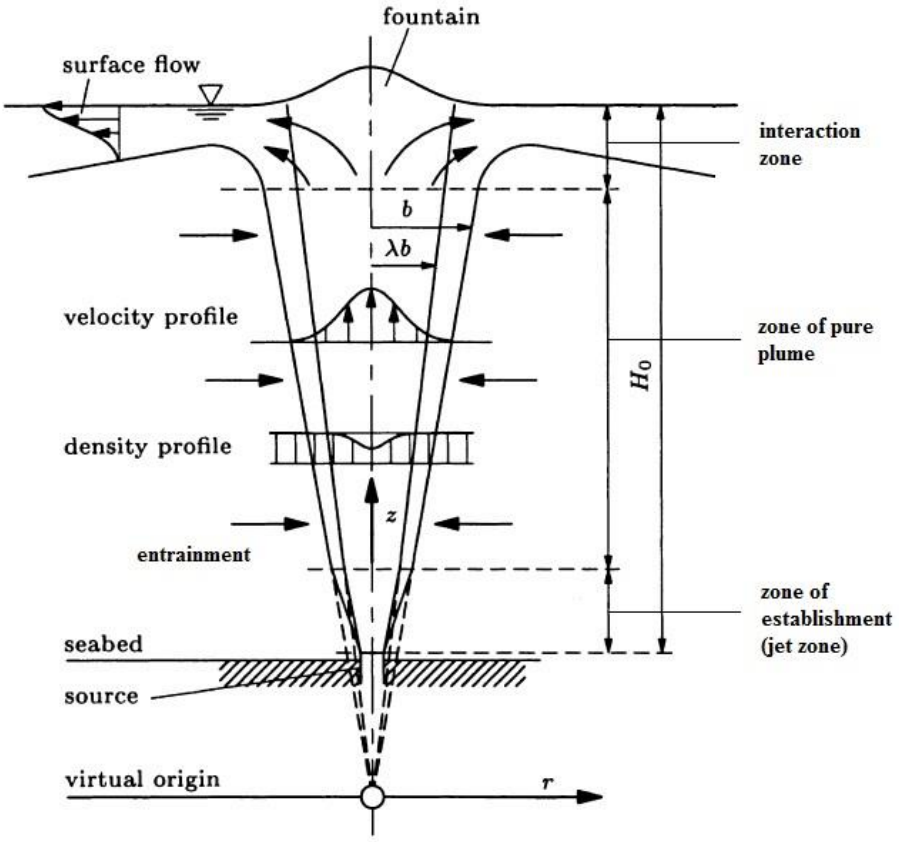

Figure 1: Definition sketch of a time-averaged bubble plume inspired by Fanneløp & Freidel (22).

In the zone of pure plume, momentum is no longer significant relative to buoyancy. Buoyancy becomes the dominant driving force for the remainder of the plume rise towards the surface.

In this region, bubble size is governed by material properties and turbulence parameters (ref. section 4.2.4). As the gas continuous to rise, oil droplets and sea water are entrained into the flow and carried to the surface (41). When the gas approaches the ocean surface, entrained water and oil droplets cannot follow the gas into the atmosphere (2). Water is diverted radially outwards the horizontal surface, dragging the oil droplets and some of the gas bubbles away from the plume axis. This surface influence spreads the oil over the surface up to the point where the flow no longer influences the surface water motion (41). This turning process, which occurs in the interaction zone, causes an elevation of the water surface due to entrained water momentum. The elevation of the water surface is referred to as a fountain (22). Most of the gas particles will continue their rise into atmosphere, creating a surface disturbance or "boil zone".

Fountain effects are only evaluated in the validation model, chapter 7. The oil slick developing at the sea surface is not evaluated in the present work.

Figure 1 shows a sketch of the ocean flow resulting from an underwater release of hydrocarbons. The oil and gas originate from a point source at water depth H_0. The three zones of interest are marked by name. Figure 1 is inspired by similar sketch presented in Friedl & Fanneløp (22). Other features of figure 1 are further explained in section 2.4.

2.2.1 Starting Plume Dynamics

In case of a underwater plume from an instantaneously started source, the gas feeds into a cap which builds up buoyancy before it rises to the surface (2). Due to drag forces exerted on the initial cap, the surrounding water is set in motion upwards making the subsequent bubbles and oil droplets to rise faster and more individually. The shape of a starting plume may appear as an ice-cream cone, with a spherical cap on top (ref. figure 2).

Fanneløp & Bettelini (29) explain that the top of the cap mainly consists of large bubbles. The large bubbles rise much faster than the mean flow inside the cap. The highly turbulent plume region consists mainly of small bubbles, while the larger bubbles exists in the extremes of the plume, where coalescence dominates due to lower turbulence level (47) (ref. section 4.2.4). Moreover, the amount of surfacing bubbles and droplets may be influenced by the cap. This variation is mainly explained by the different dynamics of the initial cap and the subsequent bubbles (29).

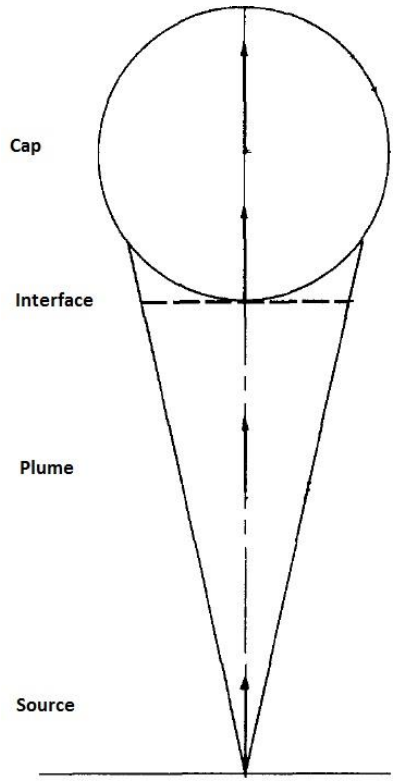

Figure 2: Sketch of a starting bubble plume inspired by Fanneløp & Bettelini (29).

2.3 Effect of Stratification, Ocean Currents and Gas Dissolution

Without the presence of ocean currents and gas dissolution, oil droplets may be transported quickly to the water surface by the more buoyant gas phase, as explained in section 2.2. However, in deep water releases, external factors are more dominant and may strongly affect the dispersed phase.

2.3.1 Stratification

Multi-phase plumes differs from single-phase plumes (e.g. wastewater plumes) by the fact that gas bubbles and oil droplets, that are source of buoyancy, may separate from the entrained sea water plume as it becomes trapped by water stratification (ref. figure 3) or deflected by ocean currents (40). By stratification the author refers to water masses with different properties (e.g. salinity, oxygenation, density, and temperature). This stratification effect may form layers that act as barriers to water mixing. These layers are normally arranged according to density, with the least dense water sitting above the more dense water.

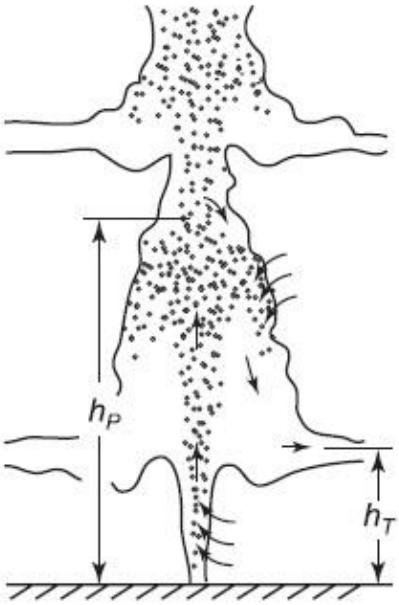

Figure 3: Sketch of pure water stratification (40).

As seen from figure 3, the plume reaches a peel height, h_p, where the buoyancy of the dispersed phase no longer manages to lift the entrained sea water. At this point, an outer downdraught plume of dissolved and finely dispersed hydrocarbons and sea water is created, forming a horizontal intrusion at a level of neutral buoyancy, h_T (50). In the current application, this stratification effect is neglected. The properties of sea water are assumed constant along the depth (i.e. density, temperature, and salinity), and only the hydrostatic pressure is changing. Figure 3 illustrates the effect of pure stratification for multi-phase plumes.

2.3.2 Effect of Ocean Currents and Gas Dissolution

When strong horizontal ocean currents dominate multi-phase plumes, a complete separation occurs between the gas bubbles and the wake of entrained water plume, due to their individual buoyancy (ref. figure 4). The entrained water wake is then left with finely dispersed and dissolved hydrocarbons (50). Due to small difference in density, the degree of ocean currents determines whether the oil droplets may separate from the plume of entrained ambient water (ref. figure 5) (40). Moreover, a separation of oil droplets may occur based on the drop size. Larger oil droplets will surface first, while the smaller drops may be carried further down-current prior to reaching the surface (41).

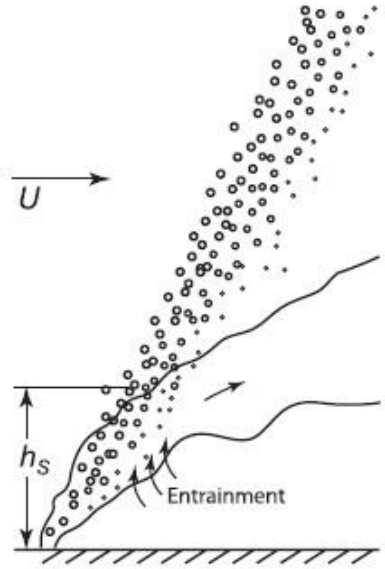

Figure 4: Sketch of the effect of pure ocean current on gas bubbles (40).

In case of deep water blowouts, the driving buoyancy force of gas bubbles can be completely lost due to dissolution, which is strongly dependent of sufficient residence time in the ambient sea water. Situations where the ocean currents are too weak to completely separate the two dispersed phases, oil droplets will start to rise slowly due to buoyancy forces alone and without any assistance from the more buoyant gas bubbles (41). Since this separation is a self-reinforcing process, the oil droplets tend to be even more affected by the ambient ocean currents after the gas volume is depleted (56).

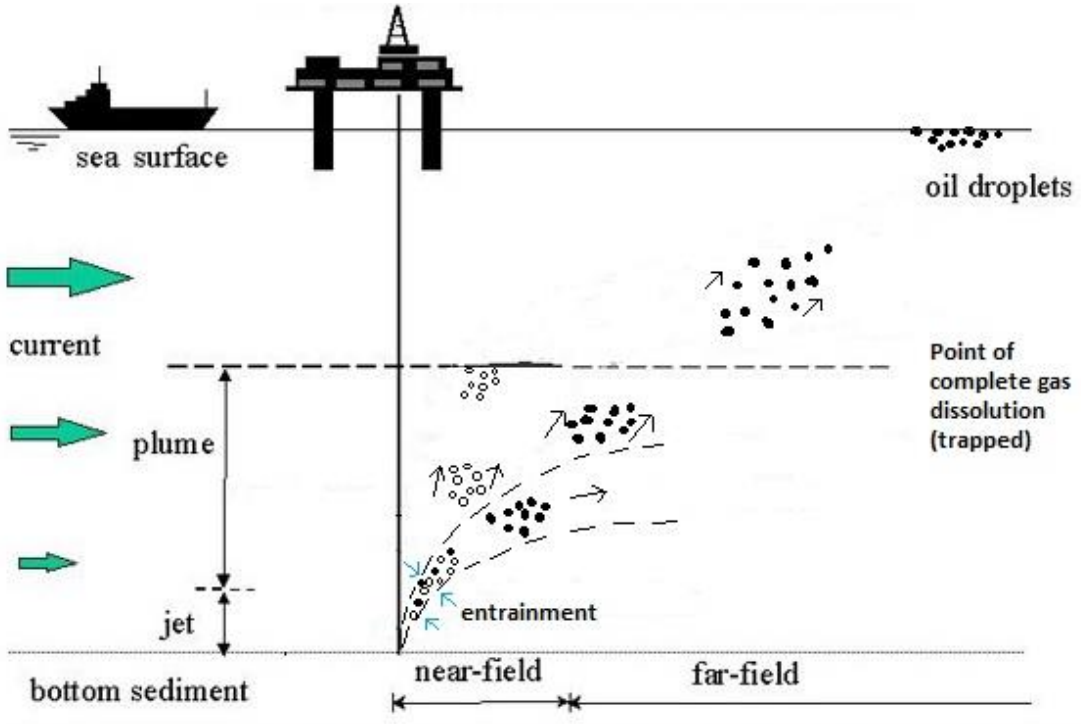

Figure 5: Deep water blowouts of oil and gas inspired by Yapa (61).

Figure 5 illustrates a possible blowout scenario, including the entrained water wake, gas bubbles, and oil droplets. As seen from figure 5, ambient ocean currents are increasing as the ocean depth decreases. The highly buoyant gas bubbles may separate from the plume of entrained ocean water, leaving the less buoyant oil droplets behind. As ocean currents continues to increase, the oil droplets starts to separate from the entrained ocean water. When the mass transfer rate of gas into the ambient ocean is depleted, the gas plume is trapped below the ocean surface. The oil drops will eventually surface, and develop an oil slick on the sea surface. However, oil slicks will not be evaluated in the present work.

Properties of gas bubbles and oil droplets are further discussed in chapter 4 and 6. Theory and implementation of gas dissolution and ocean currents are explained in Chapter 4.

2.4 History of Plume Modeling

Over the last 40 years, the research on buoyant bubble plumes has been quite active. Fanneløp & Sjøen (61) performed both theoretical and experiments studies of gas plumes on which a lot of later research is based. McDougall (63), Milgram (24), and Fanneløp et al. (65) developed all numerical models which considered gas expansion within the jet/plume zone (ref. figure 1). These models were limited to vertical buoyant plumes, and the ambient ocean currents where not taken into account. A common goal of these models was to determine and explain the relationship between local plume properties and the entrainment coefficients. In addition, Milgram (24) introduced a momentum amplification factor to account for the momentum transfer caused by turbulent fluctuations. The integral models mentioned above are based upon Eulerian integral formulations (66), where the control volumes are fixed in space.

In addition, multiple oil spills models have been developed through the years, such as Spaulding (64) and ASCE (62). The point of interest for most of these models are surface or near surface spills such as from a tanker. However, some exception exists. Yapa & Zheng (66) developed a highly complex integral model which takes into account oil, gas and entrained sea water from sub-sea blowouts, in addition to the effects of ocean currents and density stratification. Johansen (15) extended this model to be valid for deep water blowouts by including among other effects of non-ideal gas behavior, gas dissolution and the possibility of hydrate formation. The model was named "DeepBlow" and a brief introduction of this model is provided in section 2.4.2.

2.4.1 Governing Equations

Various integral models may be solved by different governing equations; however, the fundamental concept is often the same. A brief introduction to the basics behind the integral model developed by Fanneløp & Bettelini (59) is presented in this section.

For most integral plume models, some coefficients are dependent on the profile assumptions used. Radial distribution of density and velocity profiles is often assumed Gaussian distributed, as can be seen in figure 1, and may be expressed as follows (59):

$$\overline{w}(r,z) = w(z)\, e^{(-r^2/b^2)} \qquad \text{EQUATION 1}$$

where w refers to the water velocity. r and z is the horizontal distance from the plume axis and the vertical distance from the source, and b defines the width of the buoyancy profile (ref. figure 1). The overbar is used for all quantities dependent on both r and z. Equation 1 is, however, not valid for interactions in the water surface. Further, the Gaussian distributed gas density profile may take following form:

$$\rho_w - \bar{\rho}(r,z) = \ [\rho_w - \rho(z)]e^{\lfloor -r^2/(\lambda b)^2 \rfloor} \qquad \text{EQUATION 2}$$

where λ denotes the ratio between the widths of the buoyancy and momentum profiles (ref. figure 1). For steady state bubble plumes the mass balance equation and the momentum balance equation are given by (59):

$$\frac{d}{dz}(\varphi_1 w b^2) = 2\alpha w b \qquad \text{EQUATION 3}$$

and

$$\frac{d}{dz}(\varphi_2 w^2 b^2) = \varphi_3 g \frac{\rho_w - \rho}{\rho_w} \qquad \text{EQUATION 4}$$

,where φ_i is dependent on the assumed shape of the velocity and buoyancy profiles in the plume, and α is the entrainment coefficient. Modeling results by Fanneløp & Bettelini (59) revealed that the plume development is sensitive to variations in the entrainment coefficient, while variations of the remaining parameters in their expected uncertainty ranges have only a minor influence on the results. More information about the governing equations and general model set-up is found in Fanneløp & Bettelini (59).

If the coefficients are tuned properly integral models provide a good representation of a rising bubble plume, but may offer limited information on the surface interactions (2). These models represent a cheap and efficient analysis tool, but obviously rely heavily on appropriate empirical data available (e.g. entrainment coefficient).

2.4.2 The DeepBlow Model

As noted in section 2.4, the DeepBlow model accounts for the effects of deep water. The model is an integral model based on a Lagrangian concept, in contrast to the bubble plume models mentioned in section 2.4, which is based on Eulerian formulation. In a Lagrangian formulation the control volumes are moving with the plume, where the plume is represented by a series of non-interfering elements. Johansen (15) describes the concept as follows: "Each element, which can be thought of as a cylinder or section of a bent cone, is characterized by its mass, location, width (radius), length (thickness), average velocity, pollutant concentration, temperature and salinity. These parameters will change as the elements moves along the trajectory, i.e., the element may increase in mass due to shear-induced and forced entrainment, while rising by buoyancy and sheared over the cross-flow." For a detailed review of the model concept, see Johansen (15) and Yapa & Zheng (66).

A number of case studies were simulated by the DeepBlow model (15). The modeling results were found to compare favorably with field observations when dissolution of gas into sea water was accounted for. In chapter 9, modeling results provided by the DeepBlow model is discussed and compared with relevant CFD simulation results.

In recent years, CFD techniques have been employed to calculate the density and velocity distributions for the underlying conservation equations, together with suitable models representing the phase interactions (5; 11; 2; 47). Acting equations and models are discussed in detail in the following chapters, together with relevant literature.

Chapter 3

Basic CFD and Governing Models

Computational Fluid Dynamics (CFD) is a computer based mathematical simulation tool providing a cost-effective mean of simulating fluid related problems (3). Mass and heat transfer, fluid flow and chemical reactions are solved by numerical solution of the governing equations of fluid dynamics, i.e. the conservation of mass, momentum and energy. These equations combine to form the Navier-Stokes equations, which are a set of partial differential equations that cannot be solved analytically except in a limited number of cases (46). However, an approximate solution can be obtained using a discretization method that approximates the partial differential equations by a set of algebraic equations. In ANSYS FLUENT, this discretization method is the Finite Volume Method (FVM) (ref. section 3.2.1).

CFD simulation allow for testing of conditions which is difficult and costly to measure experimentally. However, it is important to point out that CFD may provide misleading results due to lack of sufficient understanding of the large number of simplifications and approximations being made. The results may look plausible and pretty, while in fact being far from correct. When presenting CFD data it is therefore exceedingly important to be thorough, precise and specific.

On todays marked there are a large number of different commercial CFD packages. In the present work the use of ANSYS FLUENT 15.0.0 is employed, which offers a multitude of numerical models, applicable to different flow situations.

3.1 Typical Stages in CFD Simulations

Solving a typical CFD problem consists of several stages (46):

1. **GEOMETRY AND PHYSICAL BOUNDS**

   The geometrical model is a representation of the shape and extent for where the computational flow domain is estimated. The geometry is approximated in the preprocessor (e.g. ANSYS Workbench).

2. NUMERICAL GRID

In order to solve the flow variables, the volume occupied by the fluid requires to be divided into numerical grid cells (discretization), also known as mesh. This is done in the preprocessor. Moreover, additional refinements can be created by adaptive mesh generation in ANSYS FLUENT (ref. chapter 5).

3. MODELS AND MODELING PARAMETERS

The geometry, which is divided into a number of non-overlapping grid cells, is loaded into the solver (e.g. ANSYS FLUENT), where the required models are selected and tuned to fit the current flow problem.

4. CALCULATION OF FLOW VARIABLES

Discretization is the process where the governing partial differential equations are converted into algebraic equations (one set for each numerical grid cell). Setting up the discrete system and solving it is a matrix inversion problem, and involves a large amount of repetitive calculations (67). Error or residual values are computed from these discretized equations using an iterative method.

5. CONVERGED SOLUTION

Convergence is detected when the numerical solution approaching a single answer. When the residual values in the system becomes sufficiently small, the solution is considered converged. For steady state simulations, the calculations are stopped when convergence is reached.

6. POST PROCESSING

The results may be presented as e.g. vector plots or contour plots. When the discrete phase model is applied, particle plots can be colored by various particle properties.

7. VERIFICATION AND VALIDATION

The results should be verified and validated, which is the process of determining if the simulation meets the specification and how accurately a simulation represents a real flow situation. In situations where validation is not possible, re-simulations with denser grid and/or improvements of models and initial values may be required.

3.2 Basic Concept of CFD

As indicated above, the main strategy of CFD is to replace the continuous problem domain with a discrete domain using numerical grid cells. In real flow situations (i.e. a continuous domain) each flow variable is defined at every point in the domain. This is illustrated in figure 6, where the pressure p for the continuous domain is given by (67):

$$p = p(x)\ ,\ 0 < x < 1 \qquad \text{EQUATION 5}$$

In the discrete domain, each flow variable is defined at the grid points or in the cell centers, depending on the discretization method. Equation 6 and right hand side of figure 6 indicates that the pressure p is only defined at N grid points in the discrete domain.

$$p_i = p(x_i)\ ,\ i = 1, 2, \dots, N \qquad \text{EQUATION 6}$$

Figure 6: Continuous domain (l.s.) and discrete domain (r.s.) (67).

The approach above shows the finite-difference method which illustrates the underlying concept of discretization. The relevant flow variables are solved only at the grid point. The values at intermediate locations are determined by interpolating (67).

In ANSYS FLUENT, discretization is carried out by the Finite Volume Method (FVM), which is briefly explained in the following section.

3.2.1 The Finite Volume Method (FVM)

In the FVM, the variables of interest are located at the centroid of the control volume (i.e. grid cells). The integral form of the governing equations (i.e. conservation of mass, momentum, and energy) is applied to the control volume to establish the discrete equations for the cell. Interpolation profiles are assumed in order to describe the variation of the relevant variables between the cell centers. As an example, the integral form of the continuity equation for steady (i.e. no variation in time) and incompressible (i.e. no variation in time) flow is (67):

$$\int_s \boldsymbol{u} \cdot \hat{\boldsymbol{n}}\, dS = 0$$ EQUATION 7

The surface S represents the control volume and $\hat{n}$ is the normalized vector perpendicular at the surface. Physically, this equation claims that the net volume flow into the control volume is zero. Figure 7 shows a typical grid cell.

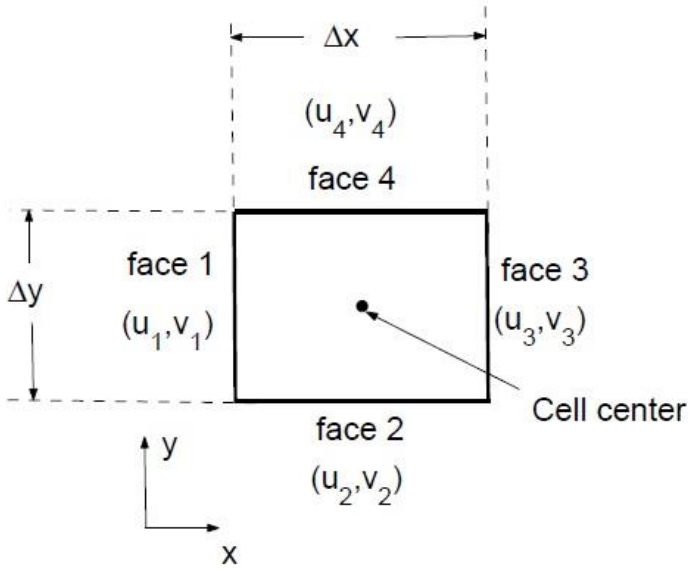

Figure 7: Typical numerical grid cell (67).

In relation to figure 7, the discrete velocity may be written as $\boldsymbol{u_i} = u_i \boldsymbol{i} + v_i \boldsymbol{j}$. Applying the integral form of the mass conservation equation (ref. Equation 7) to the control volume defined by the cell may provide the following expression:

$$-u_1 \Delta y - v_2 \Delta x + u_3 \Delta y + v_4 \Delta x = 0$$ EQUATION 8

Equation 8 illustrates the discrete form of the mass conservation equation for the control volume, showed in figure 7. The equation implies mass conservation for the grid cell. The face values u_1, v_2 are estimated by interpolating the cell center values at neighboring cells (67). This concept can be extended to any conservation equation and flow domain.

3.3 Governing Equations

In most CFD applications, the systems are based on the fundamental governing equations of fluids dynamics - the conservations of mass, momentum and energy equation (4). These conservations laws states that measurable properties of a closed system does not change as the system evolves. They are mathematical statements of three fundamental physical principles upon which all of fluid dynamics is based:

- *Mass is conserved*
- *Newton`s second law,* $F = m \cdot a$ EQUATION 9
- *Energy is conserved*

In ANSYS FLUENT, all flows solve the conservation equations for mass and momentum. The energy equation is solved when the model requires heat transfer or compressibility. In the presences of species mixing or reactions, a species conservation equation is solved. In the current application, only the conservation equations for mass and momentum is solved.

3.3.1 Conservation of Mass

The equation for conservation of mass, or continuity equation, can be written as follows (14):

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho u) = 0$$ EQUATION 10

The first term on the left side is the transient term, which indicates Equation 10 is valid for both incompressible and compressible flows. For three dimensional geometries, the operator (∇) will be on the form $\nabla = i\frac{\partial}{\partial x} + j\frac{\partial}{\partial y} + k\frac{\partial}{\partial z}$. *ρ is the fluid density and the velocity vector is defined as* $\boldsymbol{u} = u\boldsymbol{i} + v\boldsymbol{j} + w\boldsymbol{k}$.

3.3.2 Conservation of Momentum

The conservation of momentum is implemented in ANSYS FLUENT as:

$$\frac{\partial}{\partial t}(\rho \boldsymbol{u}) + \nabla \cdot (\rho \boldsymbol{u}\boldsymbol{u}) = -\nabla p + \nabla \cdot (\bar{\bar{\tau}}) + \rho \boldsymbol{g} + \boldsymbol{F} \qquad \text{EQUATION 11}$$

where p is the static pressure, and $\boldsymbol{g}$ and $\boldsymbol{F}$ are the gravitational body force and external body forces, such as interactions with the dispersed phase. The stress tensor, $\bar{\bar{\tau}}$, describes a Newtonian fluid and is given by (28):

$$\bar{\bar{\tau}} = \mu \left[(\nabla \boldsymbol{u} + \nabla \boldsymbol{u}^T) - \frac{2}{3} \nabla \cdot \boldsymbol{u} I \right] \qquad \text{EQUATION 12}$$

where μ is the molecular viscosity, I is the unit tensor, and the second term on the right hand side is the effect of volume dilation.

For incompressible flows the momentum equations may be written on the following form, for all three Cartesian directions (55):

$$\rho \left(\frac{\partial u}{\partial t} + \nabla \cdot (u \boldsymbol{u}) \right) = -\frac{\partial p}{\partial x} + \nabla \cdot (\mu \nabla u) + S_{Mu} \qquad \text{EQUATION 13}$$

$$\rho \left(\frac{\partial v}{\partial t} + \nabla \cdot (v \boldsymbol{u}) \right) = -\frac{\partial p}{\partial x} + \nabla \cdot (\mu \nabla v) + S_{Mv} \qquad \text{EQUATION 14}$$

$$\rho \left(\frac{\partial w}{\partial t} + \nabla \cdot (w \boldsymbol{u}) \right) = -\frac{\partial p}{\partial x} + \nabla \cdot (\mu \nabla w) + \rho \boldsymbol{g} + S_{Mw} \qquad \text{EQUATION 15}$$

As seen from Equation 15, gravitation works in z-direction in the current application. From left to right, the terms in the above equations may be identified as the transient term, the convection term, the pressure source term, the diffusion term and the source term. The source S_M can be any user-defined sources, e.g. a user-defined drag force (ref. chapter 4).

The set of equations above is together with the mass conservation equation (ref. Equation 10) collectively known as the Navier-Stokes equations.

3.4 Solution Algorithm

In the current application, a numerical method known as the Pressure-based solver is applied. This solution method contains multiphase models and discretization schemes in which fits the present flow situation (11). This is further discussed throughout this thesis.

3.4.1 Algorithms

The Pressure-based solver includes four segregated types of algorithms: SIMPLE, SIMPLEC, PISO, and Fractional Step (FSM). Steady-state calculations will generally use SIMPLE or SIMPLEC, while PISO is recommended for transient calculations (28). The FSM is typically used for time-dependent flows, but requires a considerable amount of computational effort due to a large number of outer iterations performed at each time-step. Thus, transient calculations through the PISO algorithm are employed in the current application (11).

The PISO algorithm is briefly explained in chapter 5.

In ANSYS FLUENT there are multiple models fitting different type of flow problems. The user's task is to select the appropriate models and tune them according to the current situation. Simulation of underwater oil and gas blowouts in ANSYS FLUENT requires at least two additional physical models: a turbulence model to account for the basic turbulent nature of the process and a multiphase model to separate the different phases.

3.5 Turbulence

Turbulence is a three-dimensional, time-dependent, nonlinear phenomenon. In the 1880`s, Osborne Reynolds carried out an historic visualization of flow in pipe studies. He observed that well-ordered laminar flow degenerated into a chaotic motion when the velocity in the pipe reached a certain value. This certain value is today known as the Reynolds number, which represent the ratio between inertia forces and viscous forces. If the Reynolds number is low, the flow is orderly with parallel streamlines. For high numbers, the flow will at some point give rise to a flow structure characterized by large-scale eddies (3).

In figure 8, the straight, parallel black lines are streamlines, which are everywhere parallel to the mean flow. In laminar flow the fluid particles follow the streamlines exactly. In turbulent flow eddies of many sizes are superimposed onto the mean flow and velocity fluctuations are

developed in all directions (6). In such flows, the number of scales (degree of freedom), which is defined as the number of parameters of the system that may vary independently, will be infinite.

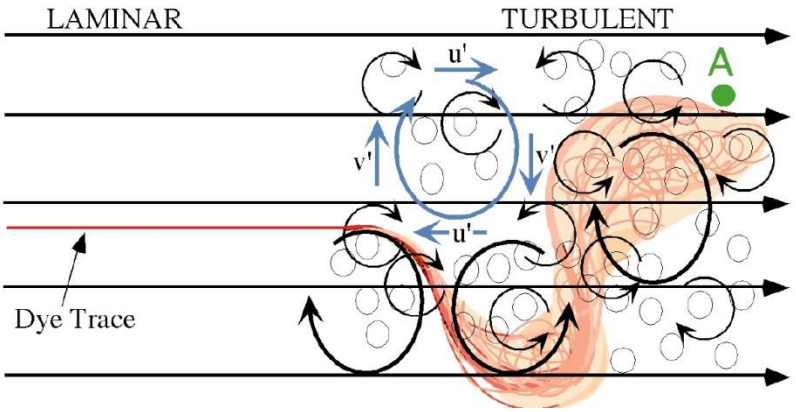

Figure 8: Dye tracer of both laminar and turbulent flow (6).

3.5.1 Modeling Turbulence

For clearness, this thesis is not a study of the most suited turbulence model in simulations of underwater oil and gas plumes. The choice of turbulence model is based upon the studies of Cloete (5) and Cloete et al. (11). However, a detailed explanation of why the chosen turbulence model may fit the current application is given in the following sections, together with some short discussions of alternative models.

In principle, turbulence is described by the Navier-Stokes equations. In most flow situations it is not achievable to resolve the wide range of scales in time and space by Direct Numerical Simulation (DNS). As the name indicates, DNS is solving the full Navier-Stokes equations and therefore resolves the whole spectrum of scales. As the CPU requirements would exceed the available computing power, DNS is not available in ANSYS FLUENT (28). For this reason, averaging procedures have to be applied to the Navier-Stokes equations to filter out at least parts of the turbulent spectrum.

The most widely used approach for calculating industrial flows is Reynolds-averaging of the equations, resulting in the Reynolds-Averaged Navier-Stokes (RANS) equations. The three most common perceptions of this term is time averaging, space averaging or ensemble averaging (31). By solving the RANS-equations, all of the unsteadiness in the flow is averaged out and regarded as part of the turbulence. By this method a smooth variation of the averaged velocities and pressure fields may be accomplished. However, the averaging process introduces additional unknown terms into the transport equations which need to be solved by suitable turbulence models.

Both computational efficiency and quality of the simulation can be affected by the choice of turbulence model. An alternative to RANS are Large Eddy Simulation (LES), which tracks the behavior of the larger eddies. However, Cloete (5) claims that LES offers very little improvement when determining bulk flow, compared with the RANS approach.

There is not a single, practical turbulence model that can reliably predict all turbulent flows with sufficient accuracy. Figure 9 illustrates the computational resources required for simulation of boundary layers (8).

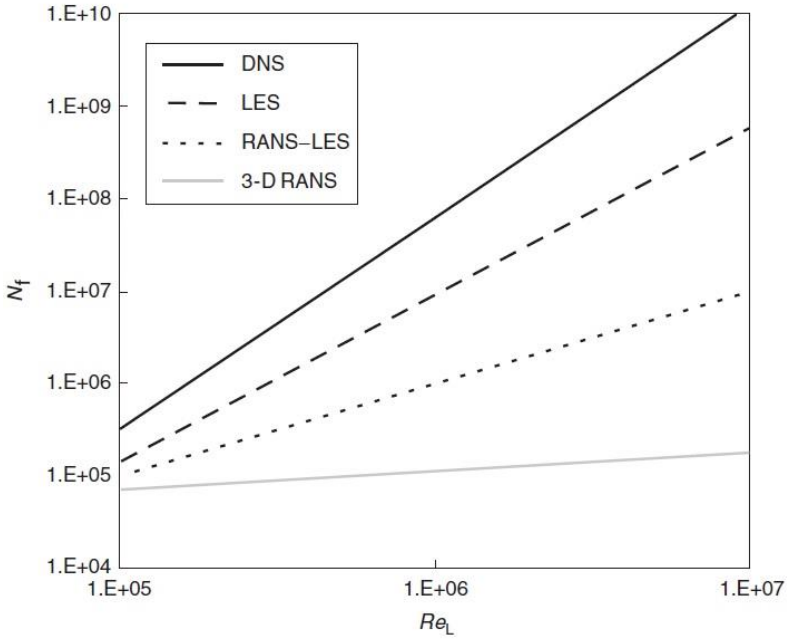

Figure 9: Approximation of computational node requirements for RANS, LES, and DNS and RANS-LES approaches (8).

It can be seen from figure 9, even at modest Reynolds numbers, that the number of grid points for DNS and LES are significant larger compared with RANS approaches. As LES is reported to offer very little improvement in calculation of the turbulent nature of plumes (5) and, moreover, requires substantial more computer resources, the RANS approach are chosen over the LES approach. The RANS approach is also applied in the work of Cloete. et al. (11), Skjetne & Olsen (2), and Pan (47).

3.5.2 Reynolds Averaging

As mentioned above, Reynolds averaging will decompose the exact Navier-Stokes equations into mean and fluctuating components. The fluctuating velocity components can be seen from figure 8. Any vector or scalar may be written as the sum of an average and a fluctuation, as seen in Equation 16. The following equation describes any flow variable (∅) such as pressure of a Reynolds averaging solution, for statistically steady process (28):

$$\emptyset(x,y,z,t) = \overline{\emptyset}(x,y,z) + \emptyset'(x,y,z,t) \qquad \text{EQUATION 16}$$

where

$$\overline{\emptyset} = lim_{T\to\infty} \frac{1}{T} \int_0^T \emptyset(x,y,z,t)\, dt \qquad \text{EQUATION 17}$$

where $\bar{u}$ and u' are the mean and fluctuating velocity components. T is the averaging interval and must be large compared to the typical time scale of the fluctuations.

The same principle yields for the standard (instantaneous) velocities. In x-direction:

$$u = \bar{u} + u' \qquad \text{EQUATION 18}$$

The new equations will provide a solution of an average flow field, and not a solution for the exact turbulent flow field (3). In the application covered by this thesis the flow is unsteady (variable statistics vary over time), which implies time averaging cannot be used and it has to be replaced with ensemble averaging. The concept of this is to imagine a set of flows in which all of the flow variables that can be controlled are identical, but the initial conditions are generated randomly. This may give flows that differ considerably from each other. An average over a large set of such flows is an ensemble average (32):

$$\overline{\emptyset}(x,y,z,t) = lim_{N\to\infty} \frac{1}{N} \int_0^T \emptyset(x,y,z,t)dt \qquad \text{EQUATION 19}$$

Here N represents the number of members of the ensemble (a set of flows in which all controllable variables are identical). N must be sufficiently large to eliminate the effects of the fluctuations (5).

As seen from Equation 18, the standard velocities are also written as the sum of an average and a fluctuation component of Reynolds-averaging solution. The ensemble-avereraged Navier-Stokes equations yield the following (55):

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \bar{\boldsymbol{u}}) = 0 \qquad \text{EQUATION 20}$$

$$\rho\left(\frac{\partial \bar{u}}{\partial t} + \nabla \cdot (\bar{\boldsymbol{u}}\bar{u})\right) = -\frac{\partial \bar{p}}{\partial x} + \nabla \cdot (\mu \nabla \bar{u}) + \overline{S_u} + \left(\frac{\partial}{\partial x}(-\rho\overline{u'u'}) + \frac{\partial}{\partial y}(-\rho\overline{u'v'}) + \frac{\partial}{\partial z}(-\rho\overline{u'w'})\right) \qquad \text{EQUATION 21}$$

$$\rho\left(\frac{\partial \bar{v}}{\partial t} + \nabla \cdot (\bar{\boldsymbol{u}}\bar{v})\right) = -\frac{\partial \bar{p}}{\partial y} + \nabla \cdot (\mu \nabla \bar{v}) + \overline{S_v} + \left(\frac{\partial}{\partial x}(-\rho\overline{v'u'}) + \frac{\partial}{\partial y}(-\rho\overline{v'v'}) + \frac{\partial}{\partial z}(-\rho\overline{v'w'})\right) \qquad \text{EQUATION 22}$$

$$\rho\left(\frac{\partial \bar{w}}{\partial t} + \nabla \cdot (\bar{\boldsymbol{u}}\bar{w})\right) = -\frac{\partial \bar{p}}{\partial z} + \nabla \cdot (\mu \nabla \bar{w}) + \overline{S_w} + \left(\frac{\partial}{\partial x}(-\rho\overline{w'u'}) + \frac{\partial}{\partial y}(-\rho\overline{w'v'}) + \frac{\partial}{\partial z}(-\rho\overline{w'w'})\right) \qquad \text{EQUATION 23}$$

The effects of turbulence appear from the additional terms such as $-\rho\overline{u'v'}$ and $-\rho\overline{w'u'}$, which represents the Reynolds stresses that must be estimated in order to close Equation 21-23. Thus, there are more unknowns than there are equations. In order to solve these equations, additional turbulence models are required to describe the Reynolds stress terms.

3.5.3 Handling Reynolds Stresses – Boussinesq Approach vs. RSM

A. Boussinesq Approach

A turbulent viscosity hypothesis was introduced by Boussinesq in 1877. With this common method the Reynolds stresses, which occurs from the Reynolds-averaged approach, are in many flow situations appropriately computed. This method is often referred to as the Eddy Viscosity Model. The Boussinesq hypothesis relates the Reynolds stresses to the mean velocity gradients (5):

$$-\rho\overline{u_i' u_j'} = \mu_t\left(\frac{\partial \bar{u}_i}{\partial x_j} + \frac{\partial \bar{u}_j}{\partial x_i}\right) - \frac{2}{3}\left(\rho k + \mu_t \frac{\partial \bar{u}_k}{\partial x_k}\right)\delta_{ij} \qquad \text{EQUATION 24}$$

where $k = \frac{1}{2}\left(\overline{u'^2} + \overline{v'^2} + \overline{w'^2}\right)$ *is the turbulent kinetic energy per unit mass. The first term on the right hand side is the turbulent or eddy viscosity,* μ_t *(dimensions Pa s). The last term on the right side of Equation 24 is the Kronecker delta (*$\delta_{ij} = 1 \; if \; i = j \; and \; \delta_{ij} = 0 \; if \; i \neq j$*). This contribution ensures that the formula gives the correct result for the normal Reynolds stresses (those with* $i = j$*) (55).*

Examples of RANS-models using the Boussineq hypothesis are the Spalart-Allmaras model, the k-ε models, and the k-ω models. The advantage of this approach is the relatively low computational cost associated with the computation of the turbulent viscosity, μ_t *(8). The reason for this low computational cost is that the Boussineq hypothesis assumes* μ_t *to be an isotropic scalar quantity (same in all directions), in combination with the Random Walk Model (ref. section 3.6.2:II). However, this assumption is not completely true for real bubble plumes. One of the primary findings of Sheng and Irons (49) is that turbulence is not isotropic for gas-liquid plumes. Experiments showed that the turbulence is greater in the vertical direction. Moreover, Johansen et al. (68) found in all their experiments a higher radial turbulence relative to axial turbulence near the free surface. These findings may favor an anisotropic turbulence model, e.g., the RSM model which accounts for directional effect of the Reynolds stress fields, briefly explained below.*

B. Reynolds Stress Model (RSM)

An alternative approach in solving the Reynolds stresses is embodied in the Reynolds Stress Model (RSM), which is part of the RANS-family. In this approach the assumption of isotropic turbulent viscosity hypothesis is discarded and the Reynolds stresses are directly computed through additional transport equations, which account for the directional effect of the

Reynolds stress fields (28). As the assumption of isotropy for underwater plumes is not entirely true, the RSM may be a good alternative in future underwater simulations.

However, Cloete (5) found completely unrealistic results when simulating gas stirred ladles using the RSM. The flow pattern predicted by the RSM was found distinctly different from the literature presented in chapter 2. The buoyant plume region was predicted much narrower and more concentrated than recommended by Freidel & Fanneløp (22). See Cloete (5) for a more comprehensive discussion of these findings.

For further information about the RSM model see the Versteeg (55) and Crowe (8).

3.5.4 RANS-models using the Boussinesq Approach

Most of the literature sources found on CFD computations of gas-liquid plumes incorporate the assumption of turbulent isotropy, which imply the use of the Boussinesq approach. As noted above, the RANS-models using the Boussinesq hypothesis is the Spaltar-Allmaras model, the k-ε models, and the k-ω models.

- *The Spaltar-Allmaras model is a one-equation model that solves the modeled transport for the turbulent viscosity. It has been shown to provide good results for boundary subjected to adverse pressure gradients. The model was in principal developed for aerodynamic flows, and it is not calibrated for general industrial flows. In ANSYS FLUENT Theory Guide (28) it is reported that the model produces relatively large errors for plane and round jet flows. Moreover, turbulent dispersion of particles cannot be included when the Spalart-Allmaras turbulence model is activated (ref. section 3.6.2:II).*
- *The k-ω models incorporates modifications for low-Reynolds number effects, compressibility, and shear flow spreading. Their main influence lies in mimicking laminar-turbulent processes, and is well applicable to wall bounded flows and free shear flows (8).*

As neither the Spaltar-Allmaras model or the k-ω models are suited for the current application, the best choice is most likely the k-ε models for estimating the continuous phase turbulence. ANSYS FLUENT provides three k-ε models; the Standard, RNG and Realizable. As this thesis mainly is an extension of the work done by Cloete (5), Cloete et al. (11), and Skjetne & Olsen (2), the choice of turbulence model is based upon their findings. Cloete (5) did a detailed study of these three models for simulation of gas stirred ladles, and the most suited model was found to be the Standard k-ε model (SKE): "In the absence of the RSM, k-ε model is the only alternative. A detailed study of these models has shown that the SKE is the

only viable option when no turbulence modulation by bubbles is included in the model". The turbulence modulation by bubbles is briefly explained in chapter 6. Moreover, the SKE is applied in the models of Cloete et al. (11), Skjetne & Olsen (2) and Pan (47), as well.

A. Standard k-ε model (SKE)

The Standard k-ε model (SKE) is one of the most common turbulence models, and was proposed by Launder and Spalding (69). The model is known to be robust, economic, and to provide reasonable accuracy for a wide range of turbulent flows in industrial and heat transfer simulations.

The SKE is a two equation model, which means it includes two extra transport equations to represent the turbulent properties of the flow. The first transported variable is the turbulent kinetic energy, k (30). The second transported variable is the turbulent dissipation, ε, which can be viewed as the rate of loss of the kinetic energy of the turbulent motion through viscosity into thermal energy. It is these variables that determine the degree of turbulence in the continuous phase.

The k-ε model includes assumption of the flow to be fully turbulent, and therefore the effects of molecular viscosity (i.e. friction within a fluid where the velocities is laminar) are negligible. Thus, SKE is only valid for fully turbulent flows.

The turbulence kinetic energy (TKE) k, and its rate of dissipation ε, are obtained from the two transport equations below, where u_i (i = 1,2,3) are the Cartesian components of the velocities and x_i are the Cartesian coordinates (28):

$$\rho\left(\frac{\partial k}{\partial t}+\frac{\partial}{\partial x_i}(ku_i)\right)=\frac{\partial}{\partial x_j}\left[\left(\mu+\frac{\mu_t}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right] + G_k + G_b - \rho\varepsilon - Y_M + S_k$$

EQUATION 25

$$\rho\left(\frac{\partial \varepsilon}{\partial t}+\frac{\partial}{\partial x_i}(\varepsilon u_i)\right)=\frac{\partial}{\partial x_j}\left[\left(\mu+\frac{\mu_t}{\sigma_\varepsilon}\right)\frac{\partial \varepsilon}{\partial x_j}\right] + C_{1\varepsilon}\frac{\varepsilon}{k}(G_k + C_{3\varepsilon}G_b) - \rho C_{2\varepsilon}\frac{\varepsilon^2}{k} + S_\varepsilon$$

EQUATION 26

In the equations above, G_b represents the generation of turbulence kinetic energy due to buoyancy. This term is only included when gravity and temperature gradients are present simultaneously. However, in the current application the temperature is assumed constant. Y_M is the contribution of the fluctuating dilatation in compressible turbulence to the overall dissipation rate. This term is only included in case of high Mach-number flows. Further, σ_k and σ_ε are turbulent Prandtl numbers for k and ε, respectively. C_{1_ε}, C_{2_ε}, and C_{3_ε} are model constants. G_k represent the production of the kinetic turbulent energy due to mean velocity gradients, and may be defined as:

$$G_k = -\rho\overline{u_i' u_j'}\frac{\partial \overline{u_j}}{\partial x_i} \qquad \text{EQUATION 27}$$

By use of the Boussinesq hypothesis (ref. Equation 24), the Reynolds stresses in Equation 27 is estimated. However, first the turbulent viscosity needs to be computed by combining k and ε (28):

$$\mu_t = \rho C_\mu \frac{k^2}{\varepsilon} \qquad \text{EQUATION 28}$$

The model constants given in Equation 25, 26, and 28 have the following default values:

$$C_{1\varepsilon} = 1.44,\ C_{2\varepsilon} = 1.92,\ C_\mu = 0.09,\ \sigma_k = 1.0,\ \sigma_\varepsilon = 1.3$$

These model constants are determined from experiments for fundamental turbulent flows, for among other jets.

Various literature sources claims that the k- ε model assumes an isotropic turbulent viscosity. However, this is only true when the Random Walk Model (RWM) is activated. The RWM assumes, in combination with the k- ε model, that the root mean square (RMS) values are equal for all three velocity fluctuations. This is further discussed in section 3.6.2:II.

3.6 Multiphase Flow

Multiphase flow is the kind of flow that occurs most frequently in nature. A phase refers to the solid, liquid, or gas like state of matter. A multiphase flow is the flow of mixture of phases (7).

In simulation of multiphase flows the different phases are referred to as dispersed phase and continuous phase (ref. chapter 2). The dispersed phase can be computed as spherical particles or parcels, which typically represents bubbles or droplets. In ANSYS FLUENT, parcels

represent groups of particles which have similar properties (e.g. diameter, velocity, density etc.). The continuous phase will normally be either gas or liquid surrounding the dispersed phase. By this approach the dispersed phase can exchange momentum, mass, and energy with fluid phase, if desired.

The motion of particles can be caused by the continuous phase alone, wakes created by other particles, and collisions between particles. When modeling multiphase flows, this interaction is referred to as the coupling between the particle motion and its surroundings (8). For highly dispersed flows only the continuous fluid affects the particle motion (e.g. drag force), and the flow denotes a one-way coupling. For denser flows the particle motion will normally affect the continuous fluid motion and a two-way coupling has to be employed (ref. figure 10).

A three-way coupling is when particle wakes and other continuous-phase disturbances affect the motion of nearby particles. If it is likely to believe that particle collision (particle-particle interactions) influences the overall motion of particles, a four-way coupling should be considered. However, this particle/parcel approach of modeling the dispersed phase is made considerably simpler when particle-particle interactions can be neglected (ref. chapter 6) (28). In the current application a two-way coupling is employed, which is further discussed in section 3.6.2.

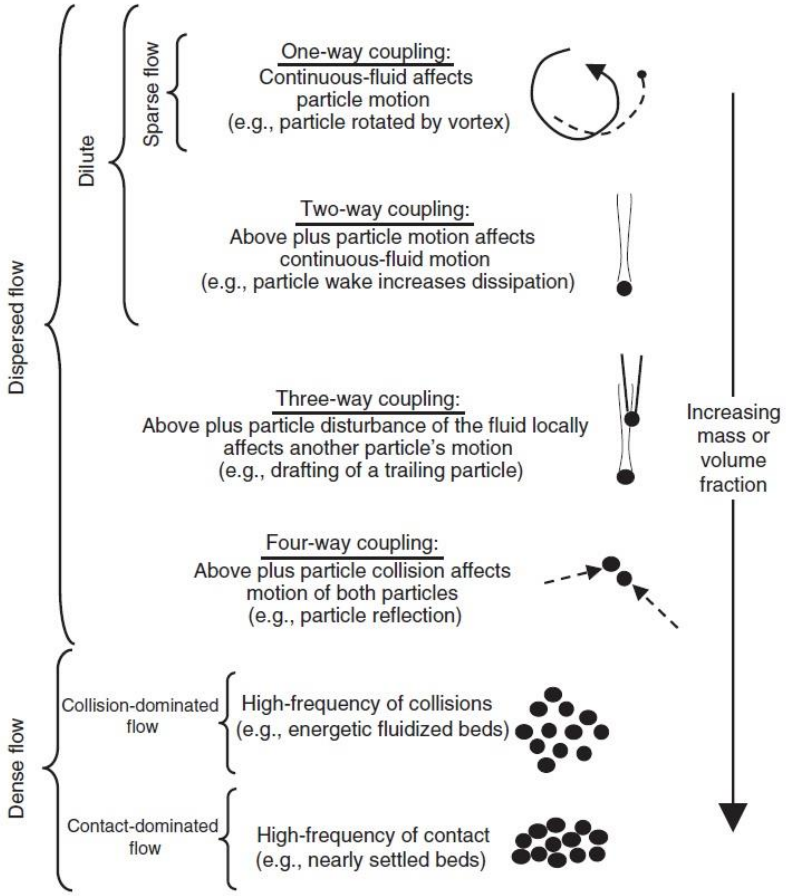

Figure 10: Dilute, dispersed, and dense flow conditions based on various interphase and intraphase coupling (8).

In order to simulate a multiphase underwater oil and gas blowout, different numerical formulations is necessary. In ANSYS FLUENT there are currently two approaches for the numerical calculations of multiphase flows: the Eulerian-Lagrangian approach and the Eulerian-Eulerian approach (28). Eulerian-Lagrangian (known as Lagrangian) models track the motion of each particle and the fluid phase is treated as a continuum by solving the Navier-Stokes equations (9). Thus, the dispersed phase can exchange momentum, mass, and energy with the continuous phase. The Eulerian-Eulerian models (called Eulerian), treat the different phases as interpenetrating continua and studying their dynamics by means of averaged equations of motion. In the Eulerian reference frame, the dispersed phase is treated as a cloud with continuum-like equations.

These two approaches are described and discussed in the following sections. For clearness, the choice of approaches is based upon the work of Smith (26), Domgin et al. (42), Cloete (5), Cloete et al. (11), and Skjetne & Olsen (2).

3.6.1 CFD Simulation of Multiphase Plumes

Historically, CFD simulations of multiphase plumes were typically two-dimensional axisymmetric, using either the Lagrangian or the Eulerian approach to compute the dispersed bubble phase (26; 42). However, these models did not account for the interaction between the water surface and the atmospheric air. Later research on gas stirred ladles shows a combination of the Lagrangian and the Eulerian method, which provide information on both the rising plume and the surface interactions (5). Cloete et al. (11) extended this to account for bubble plumes originating from a subsea gas release. In this thesis, the general model set-up in ANSYS FLUENT is inspired by this work.

As discussed in section 2.4, integral model provides a good representation of a buoyant plume. However, the method provides limited information on the behavior of the water surface. CFD simulations, on the other hand, are more fundamental and can generate information of both the bubble plume and the surface interactions. In simulations of oil and gas plumes it is essential to find the balance between accuracy and computational efficiency, since the geometry and the number of numerical grid cells can be enormous. The next sections explain the fundamental theory behind the models used.

I. Lagrangian Method

In the Lagrangian reference frame, the dispersed phase is treated as a set of discrete particles (ref. figure 11). The ambient fluid is treated as a continuum by solving the Navier-Stokes equations, while the dispersed phase is solved by tracking the particles (e.g. bubbles and droplets) and monitors the change in their properties (e.g. density, mass, size, etc.), through the calculated field (28). These discrete particles may move freely in the continuous phase, exchanging mass, momentum and energy with the continuum and other particles, if activated. Both drag- and buoyancy forces may affect each particle, as long as the two-way coupling is employed.

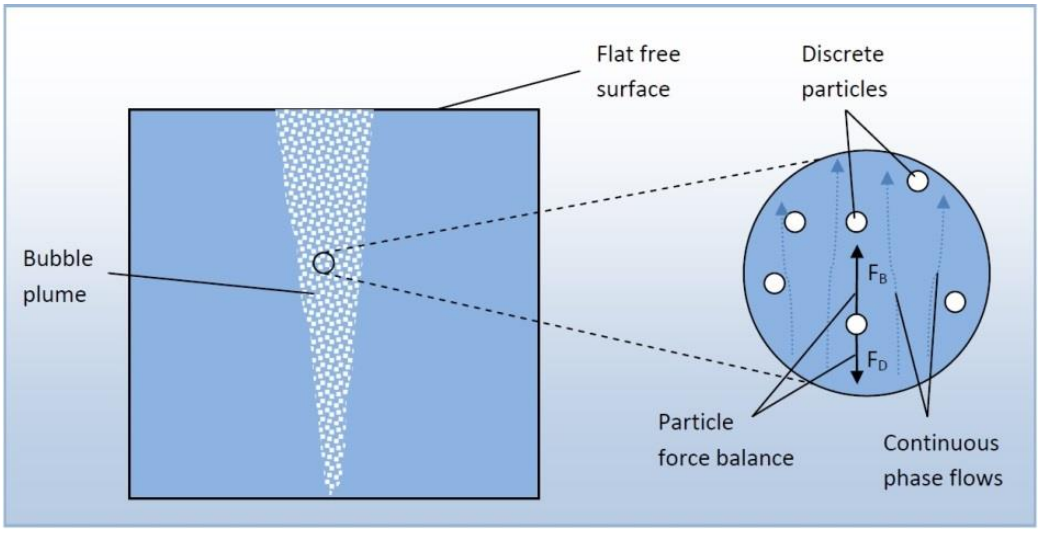

Figure 11: Sketch of the Lagrangian calculation procedure (5).

Figure 11 illustrates buoyant gas bubbles surrounded by water. In the presence of oil droplets, the concept is the same as the oil droplets can be modelled as particles/parcels as well. Moreover, multiple parameters require appropriate tuning and user-defined functions to provide realistic results (ref. chapter 4). Bubble/droplet size distribution, bubble density, correct drag law and activation of turbulent dispersion are some of the models and user-defined parameters required. As figure 11 indicates, the surface is assumed flat and frictionless as no multiphase interface model is employed, yet.

II. Eulerian Method

The Eulerain description of fluid flow does not include individual fluid particles. Instead, a control volume is defined to calculate the flow properties (e.g. pressure, velocity) (8). For multiphase modeling, the different phases are treated as interpenetrating continua, meaning their volume fractions are assumed to be continuous functions of space and time and their sum is equal to one (28). Conservation equations are derived for each dispersed phase on a control volume basis, which have similar form as that for the surrounding fluid. According to Cloete (5), pure Eulerian approaches are computationally expensive, due to high requirements of grid resolution.

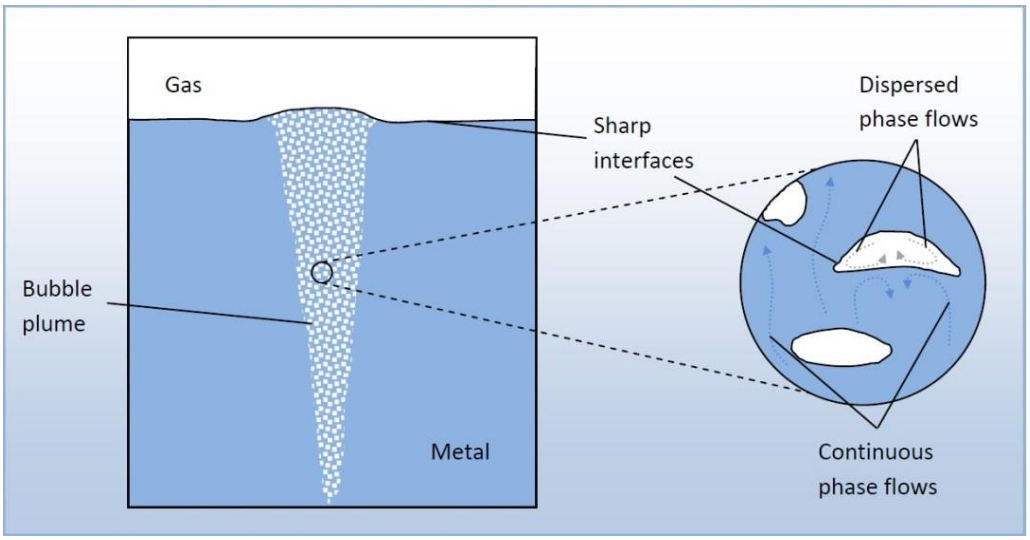

Figure 12: Sketch of the Eulerian calculation procedure (5).

Fortunately, volume-averaged Eulerian multiphase models are specifically designed for sharp interfaces between various phases (ref. figure 12). However, the sharp interphase tracking makes it impractical to model a deep water multiphase plume, as the number of bubbles and droplets are enormous.

III. Coupled Model – Lagrangian and Eulerian

Cloete (5) suggests a combination of the two methods; Eulerian-Eulerian-Lagrangian. The Lagrangian method represents and tracks the dispersed phase, while a specialized interface technique of the Eulerian model can capture the free surface, between water and atmospheric air (ref. figure 13). In ANSYS FLUENT, three different Eulerian multiphase

approaches are available: the volume of fluid (VOF) model, the mixture model, and the Eulerian model (ref. section 3.6.3).

As pointed out by Cloete (5), the Volume of Fluid (VOF) model is the most suitable approach for underwater hydrocarbons releases. The VOF model contains a surface-tracking scheme, named Geo-Reconstruction (section 3.6.3:A), which include transient tracking of any liquid-gas interface. As the interface of water and atmospheric air is the point of interest for this surface-tracking technique in the current application, the computational expenses should be affordable.

In ANSYS FLUENT, the Lagrangian method of tracking particles is represented by the Discrete Phase Model (DPM). The DPM performs Langrangian trajectory calculations for the dispersed phase, including coupling with the continuous phase, as discussed in the previous sections. This approach is computational efficient, as the bubbles and droplets is modeled as parcels.

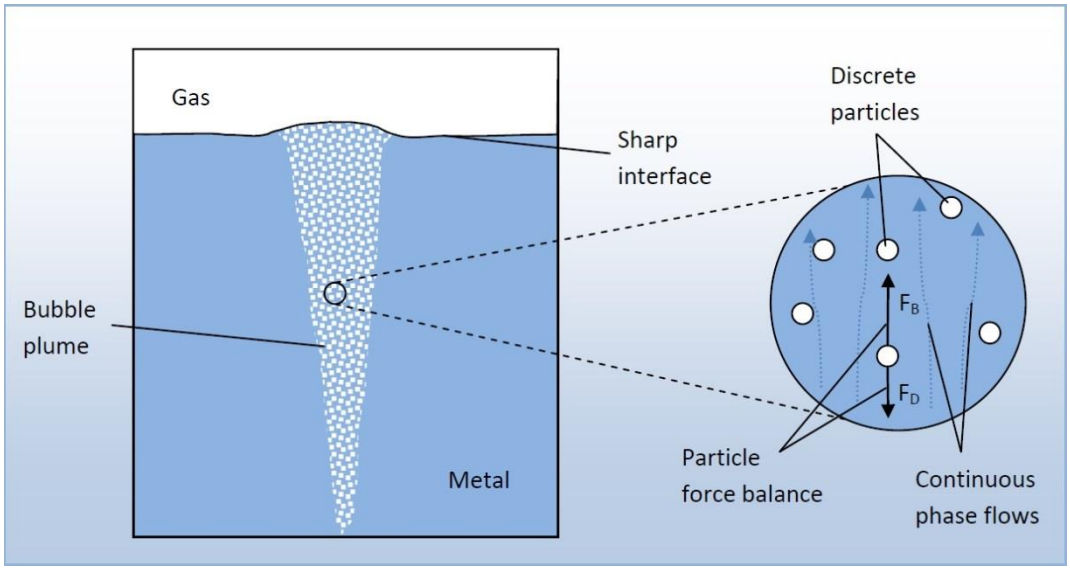

Figure 13: Sketch of the combined Eulerian-Eulerian-Lagrangian approach (5).

Figure 13 illustrate a combination of the Lagrangian and the Eulerian method. In the presence of two-way coupling, the discrete particles interact with the ambient water. A clearly visible fountain is developed right after the first parcels reach the water surface, as explained in chapter 2. In addition, the VOF model takes care of the multiphase interaction between the two continuous phases. A combination of the Lagrangian and the Eulerian reference frame was in 2001 tested out by Han et al. (70) in simulation of an oil-water model of a gas stirred ladle. However, the VOF model was only applied to track the interface between metal slag leaving the top gas surface horizontal and fixed.

3.6.2 Lagrangian Approaches: Point-Force vs. Resolved-Surface Treatment

As discussed in the previous sections, the DPM follows the Eulerian-Lagrange approach. This approach employs the Lagrangian frame of reference for the discrete particle phase, and the continuous phase is solved on an Eulerian grid. Each parcel is tracked in the flow domain by solving the force balance equation expressed in the Lagrangian formulation (28):

$$m_p \frac{d\boldsymbol{u_p}}{dt} = \boldsymbol{F}_{body} + \boldsymbol{F}_{surf} + \boldsymbol{F}_{coll} \qquad \text{EQUATION 29}$$

Equation 29 represents Newton's second law, where the left hand side is the particle mass multiplied by acceleration of the particle. The right hand side contains the body forces (e.g. gravity, buoyancy) acting on each particle, the surface forces (e.g. drag, lift), and the particle-particle or particle-wall collision forces. The last term on the right hand side is neglected in the current application.

Lagrangian approaches are divided into two different treatments of the particle phase; point-force and resolved-surface treatment (8). The main difference between the two methods is how the surface forces are calculated. The point-force represents a treatment of the particle surface to obtain the forces. Each particle is commonly described at a single point that moves with independent velocity, and individual particle trajectories are computed. Surface forces like lift, drag and stress are found from theoretical and empirical treatment of the relative velocity between the particle and continuous phase. The most essential assumption of the point-force treatment is that the method considers the surrounding phase as a hypothetical continuous flow having properties defined at the particles center of gravity (43). However, the method may fail in case of highly dense particle flows. Figure 14a illustrates the point-force treatment for Eulerian-Lagrangian methods.

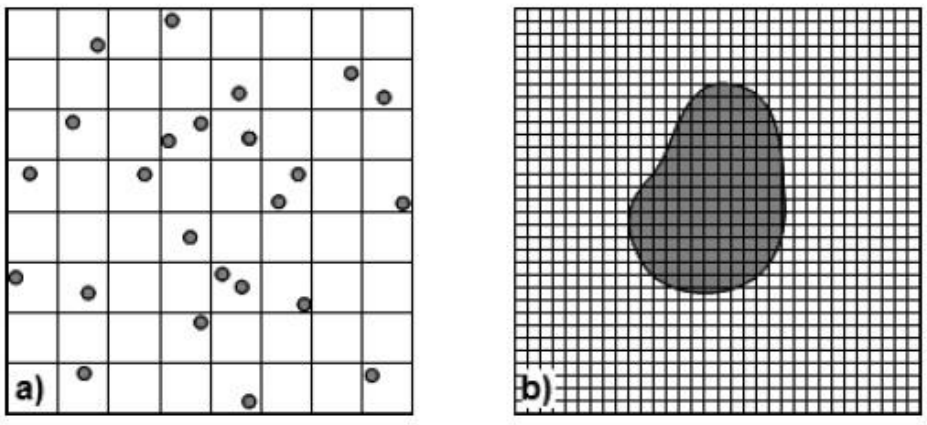

Figure 14: a) Point-force representation and b) resolved-surface representation (43).

In the method of resolved-surface (ref. figure 14b), the continuous flow surrounding each particle must be solved to a high resolution (8). In order to obtain the surface forces and the net momentum interactions, the flow solution is numerically integrated over the particle surface, with respect to continuous phase pressure and shear stresses. This usually results in more realistic particle surface forces, however, at the price of high numerical grid resolution over the particles.

The Discrete Phase Model (DPM) utilizes the point-force treatment. In simulation of deep water oil and gas releases, the resolved-surface treatment would require major amount of available computational resources due to the high number of parcels and the required fine grid resolution.

A. The Discrete Phase Model (DPM)

As discussed in the previous sections, the Eulerian methods for simulation of the dispersed phase are too computationally expensive, due to the high number of bubbles and droplets. The only choice of model left, in ANSYS FLUENT, is the Eulerian-Lagrangian Discrete Phase Model (DPM).This model seems like a reasonable choice, based on the previous sections and a short excerpt from the ANSYS FLUENT Theory Guide (28):

"In addition to solving the transport equations for the continuous phase, the DPM model allows the users to simulate a discrete phase in a Lagrangian frame of reference. This second phase consists of spherical particles (which may be taken to represent droplets or bubbles) dispersed in the continuous phase. ANSYS FLUENT computes the trajectories of these discrete phase entities, as well as heat and mass transfer to/from them. The coupling between the phases its impact on both the discrete phase trajectories and the continuous phase flow can be included."

In addition, through user-defined functions (UDFs) (ref. chapter 4), the drag force may be defined to fit the correct type of particle (e.g. bubble and drops). UDFs may also control parameters as particle density, particle size, and mass transfer through various macros, which can be connected to each particle and the continuous phase.

However, the DPM model contains one important assumption (28):

- *As the discrete phase particles does not occupy any volume, the dispersed phase should be sufficient dilute so that particle-particle interactions and the effects of the particle volume on the continuous phase are negligible. In practice, these issues imply that the discrete phase must be present at a fairly low volume fraction, usually less*

than 10-12%, based on the actual grid cell. This limit in terms of the dispersed phase is to provide reliable simulation results (5).

This assumption may cause problems when simulating the significantly smaller validation model (ref. chapter 7), due to fairly high flow rate of air and a quite dense grid resolution. However, when simulating the DeepSpill experiment, a significant coarser grid distribution is required to achieve appropriate computational time (ref. chapter 8).

I. Particle Force Balance

After selecting required initial values, streams of parcels are injected into the continuous phase. While the parcels are moving through the ambient fluid, trajectories of the discrete phase are computed. This is done by integrating the force balance over the particles, as noted in section 3.6.2. The force balance equates the particle inertia with the forces acting on the particle, and may be written as (28):

$$\frac{d\boldsymbol{u}_p}{dt} = \boldsymbol{F}_D\left(u - u_p\right) + \frac{\boldsymbol{g}(\rho_p - \rho)}{\rho_p} + \boldsymbol{F} \qquad \text{EQUATION 30}$$

As seen from Equation 30, the acceleration of a particle is influenced by drag effects and gravity, in addition to the term $\boldsymbol{F}$. Here $\boldsymbol{F}$ represent an additional acceleration (force/unit particle mass). According to Skjetne & Olsen (2), the bubble/droplet acceleration is influenced by buoyancy, drag, virtual (or added) mass, lift and turbulent dispersion. Thus, Equation 30 may be extended to account for buoyant bubbles/droplets:

$$\frac{d\boldsymbol{u}_b}{dt} = \boldsymbol{F}_D(u - u_b) + \frac{\boldsymbol{g}(\rho_b - \rho)}{\rho_b} + \boldsymbol{F}_{VM} + \boldsymbol{F}_L + \boldsymbol{F}_{TD} \qquad \text{EQUATION 31}$$

where $\boldsymbol{F}_{VM}$ is the virtual mass force and $\boldsymbol{F}_{TD}$ is the turbulent dispersion force. These forces are further discussed in the following sections. $\boldsymbol{F}_L$ is the lift force, which is neglected in the present work. According to Olsen & Cloete (52), comparisons without a lift force did provide good agreement with experiments. The term $\boldsymbol{F}_D(u - u_p)$ represents the drag term (drag force per unit particle mass). In Equation 31, the drag coefficient, C_D, is a function of the particle shape and have to fit the correct type of simulated particle (e.g. bubble and droplet). A user-defined drag function may be hooked to the DPM in ANSYS FLUENT. See section 4.1 for information about the drag coefficients used in the current application.

$$F_D = \frac{18\mu}{\rho_p d_p^2}\frac{C_D Re}{24}$$ EQUATION 32

Equation 32 shows the expression of the drag force calculated for each particle in ANSYS FLUENT. Here Re is the relative Reynolds number, which is defined as (28):

$$Re \equiv \frac{\rho_l d_p |\boldsymbol{u}_p - \boldsymbol{u}_l|}{\mu_l}$$ EQUATION 33

where μ_l is the molecular viscosity of the fluid, ρ_l is the fluid density, and d_p is the bubble/droplet diameter.

Virtual Mass Force

As mentioned above, the particle force equation (ref. Equation 30) incorporates additional forces that may be important under special circumstances. The virtual mass force is required to accelerate the fluid surrounding a particle in the Lagrangian frame of reference. This effect is more clearly described by Johansen et al. (51). When a particle rises through surrounding fluid with terminal velocity $v_{p,1}$, a steady drag force is exerted on the particle. However, if the particle is accelerated to a higher velocity $v_{p,2}$, the drag force exercised on the particle is slightly higher, and this additional force is defined as the virtual mass force. As acceleration of particles usually is caused by increased kinetic energy in the flow field, this is supplied by work on the particle during the acceleration. In ANSYS FLUENT the virtual mass force is implemented as:

$$\boldsymbol{F}_v = \frac{1}{2}\frac{\rho}{\rho_p}\frac{d}{dt}\left(\boldsymbol{u}_f - \boldsymbol{u}_p\right)$$ EQUATION 34

,where the constant $\frac{1}{2}$ indicates potential flow. $\boldsymbol{u}_f$ and $\boldsymbol{u}_p$ refers to the fluid velocity and the particle velocity, receptively. The virtual mass force is important when $\rho > \rho_p$, which e.g. refers to buoyant bubbles or droplets. The virtual mass force is implemented as an additional drag force in ANSYS FLUENT (ref. section 3.6.3:A).

II. Turbulent Dispersion of Particles

The jet zone developed by high velocity bubbles and droplets, explained in chapter 2, will form a turbulent velocity field with large concentrations of gas and oil. In this turbulent velocity field, the motions of particles are highly unpredictable. In regions where the void fraction is significant, an interfacial force associated with particle dispersion is of importance in subsea blowouts (44). This particle dispersion force is explained by the instantaneous drag on a fluid eddy is greatest, and hence the amplitude of the oscillation is least, when the eddy is moving in the direction of increased particle concentration. The amplitude will quickly start to oscillate more and result in a "saw blade" motion of entrained particles away from the region of high particle concentration (44). Therefore, the degree of particle dispersion are dependent on the particle concentration gradients in the ambient water, the frequency and amplitude of oscillation, and the instantaneous drag force, which in turn depends on the particle and fluid properties. The theory of particles moving away from regions of high concentration gradients are consistent with the assumption of Schmidtke & Lucas (45), which claims that turbulent dispersion forces causes an increase of the horizontal extension of underwater bubble plumes. Thus, the turbulent dispersion is an additional drag force originating from velocity fluctuations in a turbulent velocity field (11).

In ANSYS FLUENT the user can choose between two different turbulent dispersion models; the stochastic tracking model or the particle cloud model. Only the stochastic tracking is evaluated in this thesis.

Stochastic Tracking (Random Walk) Model

A random walk model consists of a large number of statistically independent steps, which is suitable to represent the chaotic nature of turbulent diffusion. The stochastic tracking (random walk) model includes this effect of instantaneous turbulent velocity fluctuations (i.e. u', v', and w') on the particle trajectories. This is done by insert the instantaneous velocities of the turbulent flow, $u = \bar{u} + u'$, into the particle force equation (ref. Equation 31) (28). The instantaneous fluid velocities were described in section 3.5.2 (ref. Equation 18).

For each individual particle the turbulent dispersion trajectory is calculated by integrating Equation 31, using the instantaneous fluid velocity along the particle path. By computing the trajectory in this manner, for a sufficient number of particles, the random effects of turbulence on the particle dispersion can be included. In ANSYS FLUENT, this is done by using the Discrete Random Walk (DRW) model. Here, the fluctuating components are discrete

piecewise constant functions of time, and their random is kept constant over an interval of time given by the characteristic lifetime of an eddy (28). Each eddy is characterized by:

- *The fluctuating fluid velocity components are randomly distributed Gaussian variables.*
- *A particle is assumed to interact with the fluid phase eddy over the smaller of the "eddy lifetime", defined as a time scale, τ_e.*

Since the velocity fluctuations are assumed to follow a Gaussian probability distribution:

$$u' = \zeta\sqrt{\overline{u'^2}}$$ EQUATION 35

, where ζ is a normally distributed random number. The number of injected parcels should therefore be sufficiently large, so the dispersion accounts for all direction (personal communication with Senior Scientist Jan Erik Olsen, SINTEF Materials & Chemistry, 17/04-2014). By assumption of isotropy, the root mean square (RMS) values are equal for all three velocity fluctuations, and may be deduced from the turbulent kinetic energy, k, when the k-ε model or the k-ω model is activated:

$$\sqrt{\overline{u'^2}} = \sqrt{\overline{v'^2}} = \sqrt{\overline{w'^2}} = \sqrt{\frac{2k}{3}}$$ EQUATION 36

If RSM is activated the velocity fluctuations are calculated individually, as RSM does not employ the assumption of isotropy (ref. section 3.5.3):

$$u' = \zeta\sqrt{\overline{u'^2}}$$
$$v' = \zeta\sqrt{\overline{v'^2}}$$ EQUATION 37
$$w' = \zeta\sqrt{\overline{w'^2}}$$

After the instantaneous velocity fluctuations are calculated, the maximum time for which a particle can be influenced by a specific turbulent eddy is required. ANSYS FLUENT uses the concept of integral time, T, in prediction of this "eddy lifetime". For small tracer particles that move with the fluid, the integral time becomes the fluid Lagrangian integral time, T_L. If the k-ε model is activated,

$$T_L = 0.15\frac{k}{\varepsilon}$$ EQUATION 38

, and for simulations where RSM is desired:

$$T_L = 0.30\frac{k}{\varepsilon}$$ EQUATION 39

After the Lagrangian integral time is measured, ANSYS FLUENT can predict the characteristic lifetime of the eddy, either as a constant:

$$\tau_e = 2T_L$$ EQUATION 40

, or as a random variation about T_L:

$$\tau_e = -T_L ln(r)$$ EQUATION 41

, where r is a uniform random number between 0 and 1.

In the current ANSYS FLUENT application "Random Eddie Lifetime" is activated (ref. Equation 41), which means that the characteristic lifetime of an eddy is to be random. This choice is based on the fact that ANSYS FLUENT Theory Guide (28) refers to the scale constant, C_L, as a "not well known" constant. In addition, the option of random calculation of τ_e yields a more realistic description of a jet.

However, the particle may cross the turbulent eddy in a shorter time than the eddy lifetime (51). The crossing time is implemented in ANSYS FLUENT as:

$$t_{cross} = -\tau\, ln\left[1 - \left(\frac{L_e}{\tau|u_l - u_p|}\right)\right]$$ EQUATION 42

, where τ is the particle relaxation time, L_e is the eddy length scale, and $|u_l - u_p|$ is the magnitude of the relative velocity.

A particle is now assumed to interact with a turbulent fluid eddy for the smaller of the eddy lifetime and the particle eddy crossing time. A new value of the instantaneous velocity fluctuations is obtained by a random new value of ζ,in Equation 35, when this time is reached.

III. Steady or Unsteady Particle Tracking

When the DPM particle tracking is employed, a decision between steady or unsteady particle tracking has to be made. This option can be chosen independent of the settings for the solver. Thus, steady state trajectory simulations can be performed even when selecting a transient solver for numerical reasons. The user may also specify unsteady particle tracking when solving the steady continuous phase equations.

Steady Particle Tracking

The steady state formulation tracks particle streams spatially only, independent of time. These particle streams are tracked from the DPM inlet to a certain termination condition, for each DPM-iteration. This condition can e.g. be a given boundary condition where the particles are allowed to escape. The strength of each particle stream is defined as the number of particles flowing along it per second (5), and is calculated according to the specified mass flow rate, particle density, diameter, and the number of streams.

Unsteady Particle Tracking

In case of unsteady particle tracking, the particles is to be treated as separate entities in order to give the particles a certain distance at every particle time step or fluid flow time step. Thus, the particles are tracked through space and time. In ANSYS FLUENT, this is achieved by updating the parcels position in the domain every time step (5). The number of parcels is determined by the flow rate and time between each injection, which is controlled by the user. As mentioned in previous sections, a parcel contains a number of particles having the same properties (e.g. velocity, density, diameter).

3.6.3 Eulerian Approaches: Mixed-Fluid vs. Separated-Fluid

This section is focusing on the interaction between the primary phase (i.e. atmospheric air) and the secondary phase (i.e. water), as the dispersed phase is calculated in the Lagrangian reference frame. Eulerian techniques can be divided into mixed-fluid ("one-fluid") and separated-fluid ("two-fluid") approaches. An important assumption for these models is that the secondary phase is treated as a continuum (8), which clearly fits the continuous water phase in the current application.

Mixed-Fluid Approach

The mixed-fluid approach assumes that the secondary phase (which may take form as a continuous or dispersed phase) and the primary phase are in local kinetic and thermal equilibrium. This implies that the relative velocities and temperatures of the phases are assumed to be much smaller compared to variations in the overall flow field. As a result, the two phases are treated as a homogenous mixture within each numerical cell (43). Depending on the phase concentration, the fluid properties may change from cell to cell, and a single set of momentum conservation equation is solved for the flow mixture (8). This approach is the one used in the current simulations, and are further discussed in section 3.6.3:A.

Separated-Fluid Approach

The separated-fluid approach describes the secondary phase with the point-force assumption, which assumes that both the primary fluid and the secondary phase constitute two separate, but intermixed, continua (8). As the phases are interpenetrating, the separated-fluid approach requires two sets of momentum equations for a two-phase flow; one for the primary phase and the other for the secondary phase. In addition, the relative phase velocities and temperatures are required as they are used to determine the coupling between each phase, in contrast to the mixed-fluid approach (43).

In ANSYS FLUENT, three different Eulerian multiphase approaches are available: the Volume of Fluid (VOF) model, the mixture model, and the Eulerian model. The VOF model is applied in the work of Cloete et al. (11), Skjetne & Olsen (2), and seems to be the best suited multiphase model to solve the current problem. The concept and fundamental theory of the VOF model is discussed in the following section.

A. The Volume of Fluid (VOF) Model

The VOF model uses the mixed-fluid ("one-fluid") approach. This implies that the VOF model may simulate two or more immiscible fluids by solving a single set of momentum equations, and relies on the fact that the different phases are not interpenetrating. The position of the interface between the fluids is of interest, and the volume fraction of each phase is tracked throughout the domain (28). In every computational cell the volume fraction of all phases sums unity. Thus the variables and properties of each phase in a given computational cell are either purely representative of one of the phases, or representative of a mixture of the

phases, depending upon the volume fraction values. In other words, there are three possible conditions for each control volume:

- *α_q= 0: The cell is empty (of the q^{th} fluid).*
- *α_q= 1: The cell is full (of the q^{th} fluid).*
- *0 < α_q < 1: The cell contatins the interface between the q^{th} fluid and one or more other fluids.*

Here α_q denotes the volume fraction in one cell, and q^{th} the type of fluid.

After the volume fractions of phases in each cell are determined, the tracking of the interface between fluids are accomplished by solving a continuity equation, referred to as the "volume fraction equation":

$$\frac{1}{\rho_q}\left[\frac{\partial}{\partial t}\left(\alpha_q \rho_q\right) + \nabla \cdot \left(\alpha_q \rho_q \boldsymbol{u}_q\right) = S_{\alpha_q} + \sum_{p=1}^{n}\left(\dot{m}_{pq} - \dot{m}_{qp}\right)\right] \qquad \text{EQUATION 43}$$

where $\dot{m}_{qp}$ is the mass transfer from phase q to phase p, and $\dot{m}_{pq}$ is the mass transfer from p to phase q. As there is negligible mass transfer between atmospheric air and ocean water, $\dot{m}_{qp} = \dot{m}_{pq} = 0$ in the current application. The source term, S_{α_q}, on the right hand side, is zero by default. However, the user can specify this source term as a constant or a user-defined mass source term for each phase. In the presence work, this source term is by default as there is negligible mass added and removed from both the primary and the secondary phase. Equation 36 may be rewritten as follow (by assuming constant density in the surface region):

$$\frac{\partial}{\partial t}\alpha_q + \nabla \cdot \left(\alpha_q \overline{\boldsymbol{u}_q}\right) = 0 \qquad \text{EQUATION 44}$$

The continuity equation above (ref. Equation 44), is the same as the RANS continuity equation presented in section 3.5.2, only the volume fraction is added and the assumption of incompressibility is applied. This volume fraction equation is only solved for the secondary phase. The volume fraction of the primary phase (i.e. atmospheric air) is computed by:

$$\sum_{q=1}^{n} \alpha_q = 1 \qquad \text{EQUATION 45}$$

Moreover, a "coupling-force" between the discrete phase, computed in the Lagrangian frame of reference, and the continuous phase is required to account for the two-way coupling, discussed in section 3.6. This force, f_i^c, is now implemented in the Reynold-averaged momentum equations (ref. Equation 21-23), which may be written as (47):

$$\frac{\partial}{\partial t}(\rho\overline{u_\iota}) + \frac{\partial}{\partial x_j}(\rho\overline{u_\iota}\,\overline{u_\jmath}) = -\frac{\partial \bar{p}}{\partial x_i} + \frac{\partial}{\partial x_j}\left[\mu\left(\frac{\partial \overline{u_\iota}}{\partial x_j} + \frac{\partial \overline{u_j}}{\partial x_i}\right)\right] - \frac{\partial}{\partial x_j}(\rho\overline{u_i'u_j'}) + \rho g_i + f_i^c$$

EQUATION 46

where density and viscosity are mixture properties $\rho = \sum \alpha_k \rho_k$ *and* $\mu = \sum \alpha_k \mu_k$*, where* $k = q, p$.

The coupling-force is given by the sum of drag forces working on the rising bubbles and droplets, which may take form as force per unit volume:

$$f^c = -\frac{\sum_b (F_{drag} + F_{virtual\,mass})}{V_{cell}}$$

EQUATION 47

Equation 47 expresses the momentum transfer from the continuous phase to the discrete phase, where ANSYS FLUENT examining the change in momentum of a particle as it passes through each control volume in the domain. This is further discussed in chapter 8.

Solution Scheme

Both implicit and explicit discretization may be used in solving the volume fraction equation (ref. Equation 43). In ANSYS FLUENT, the explicit schemes uses the volume fractions that were calculated at the previous time step to obtain the face fluxes for all cells, while the implicit schemes requires the volume fraction values at the current time step. Fortunately, one scheme fits the current simulation problem better than others. The Geometric Reconstruction Scheme, which uses explicit discretization, represents the interface between fluids using a piecewise-linear approach (28). The interface between two fluids is assumed to have a linear slope within each cell, and uses this linear shape for calculation of the advection of fluid through the cell faces (ref. figure 15). This scheme provides a shaper interface compared to other discretization schemes, at the price of some additional computational time.

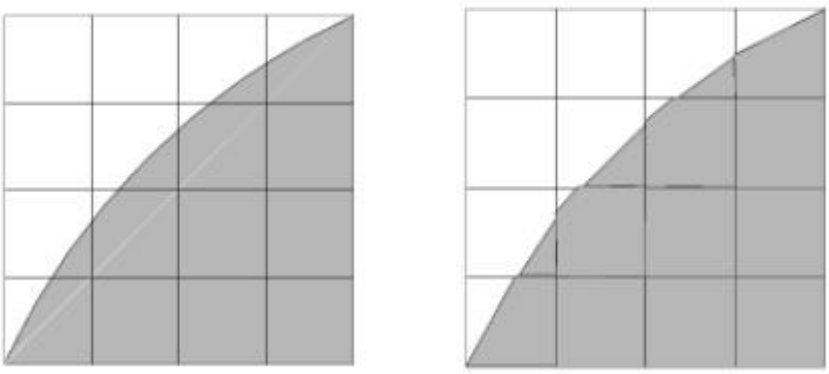

Figure 15: Interface calculations (l.s.) actual interface shape and (r.s.) interface shape represented by the geometric reconstruction scheme (28).

Chapter 4

User-defined Modeling

A user-defined function (UDF) is provided by C functions that dynamically are loaded with the ANSYS FLUENT solver. In order to make any application as realistic as possible, UDFs can be compiled or interpreted.

UDF uses "DEFINE" macros to achieve access to various models in the ANSYS FLUENT solver. The user may among other implement UDFs to allocate memory for post-process purposes, set particle properties or define various source terms. In the current application, UDFs plays an important role to achieve realistic simulation results. Bubble size and density distribution, gas dissolution, and ocean currents are only some of the subjects covered by UDFs.

4.1 Drag Force Gas Bubbles and Oil Droplets

For more than a century ago, equations of motion for rigid bodies moving through fluid at rest were established by Kirchhoff (12). Forces like drag and lift, gravity and buoyancy were found to be the most essential. This is consistent with the particle forces listed by Skjetne & Olsen (2). The lift represents those forces acting perpendicular to the bubble/droplet trajectory. However, the lift force was found to make only minor contributions to the overall force balance (ref. section 3.6.2:I). In ANSYS FLUENT, forces of gravity and buoyancy are automatically accounted for, while calculation of correct drag force requires an additional model.

4.1.1 Drag Force Gas Bubbles

As noted in section 3.6.2:I, the drag coefficient, C_D, defines the shape of the discrete particles. Xia et al. (10) presents a drag coefficient which is applied to account for larger bubbles that are deformed from the standard spherical shape. Olsen & Cloete (52) claims that: "The drag coefficient for bubbles in a plume is not necessarily the same as the coefficient for a single bubble. We use the expression of Xia et al. which represents the behavior of a bubble plume." Thus, the drag coefficient of bubbles may be defined as:

$$C_D = \frac{2}{3}\sqrt{\frac{E_0}{3}}$$ EQUATION 48

where E_0 represents the Eotvos number, which is a dimensionless number describing the characteristic shape of the bubble (11):

$$E_0 = \frac{g(\rho_l - \rho_g)d_b^2}{\sigma}$$ EQUATION 49

As seen from Equation 49, the Eotvos number is a function of the gas and liquid densities, the bubble diameter and the liquid viscosity. To account for bubble shape, the drag coefficient (ref. Equation 48) is inserted into the expression of the drag force (ref. Equation 32).

4.1.2 Drag Force Oil Droplets

Feng & Michaelides (71) presents a drag correlation that accounts for interaction between viscous oil and less viscous carrier fluid. However, the use of the standard sphere drag correlation rather than Feng and Michaelides drag correlation is found by Snyder (53) to not significantly alter simulation results. Moreover, no literature concerning experimental correlation for drag of slightly buoyant oil droplets in water is found. Thus, a spherical drag law is applied to account for drag forces exercised on the discrete buoyant oil droplets. The spherical drag law is implemented in ANSYS FLUENT as:

$$C_D = a_1 + \frac{a_2}{Re} + \frac{a_3}{Re^2}$$ EQUATION 50

where the constants a_1, a_2 and a_3 are applied over several ranges of Re, which is given by Equation 33.

4.2 Density and Size Distribution

4.2.1 Density Distribution of Gas Bubbles

A compressible flow describes the behavior of fluids experiencing significant variations in density (14). For shallow underwater gas blowouts, the density change occurs mainly due to variation in temperature and loss of hydrostatic pressure, as seen from the ideal gas law below:

$$\rho_g = \frac{P_l}{R_g T_l}$$ EQUATION 51

In the equation above, ρ_g is the gas density and P_l is the hydrostatic pressure of the surrounding water. R_g represents the individual gas constant and T_l is the temperature of the amient fluid. The hydrostatic pressure working on each bubble/droplet is given by:

$$P_l = P_{atm} + \boldsymbol{g}\rho_l(H_z - H_p)$$ EQUATION 52

where H_z is the water depth and H_p is the depth of each buoyant bubble or droplet.

However, in case of deep water gas releases the compressibility equation of state (15) should be employed:

$$\rho_g = \frac{P_l}{ZR_g T_l}$$ EQUATION 53

where the Z-factor is the compressibility of the gas, and represents the deviation of the gas density from the one computed by the ideal gas law (Z = 1), in Equation 51. Z is dependent of the gas composition, pressure, and temperature in the surrounding environment. At shallow depths (200 –300 m) and normal temperature conditions, Z tends to be close to Z = 1 for most gases (15). However, at higher pressures, Z tends to be reduced (Z < 1). The effect of the compressibility factor relative to ocean depth is illustrated in figure 16.

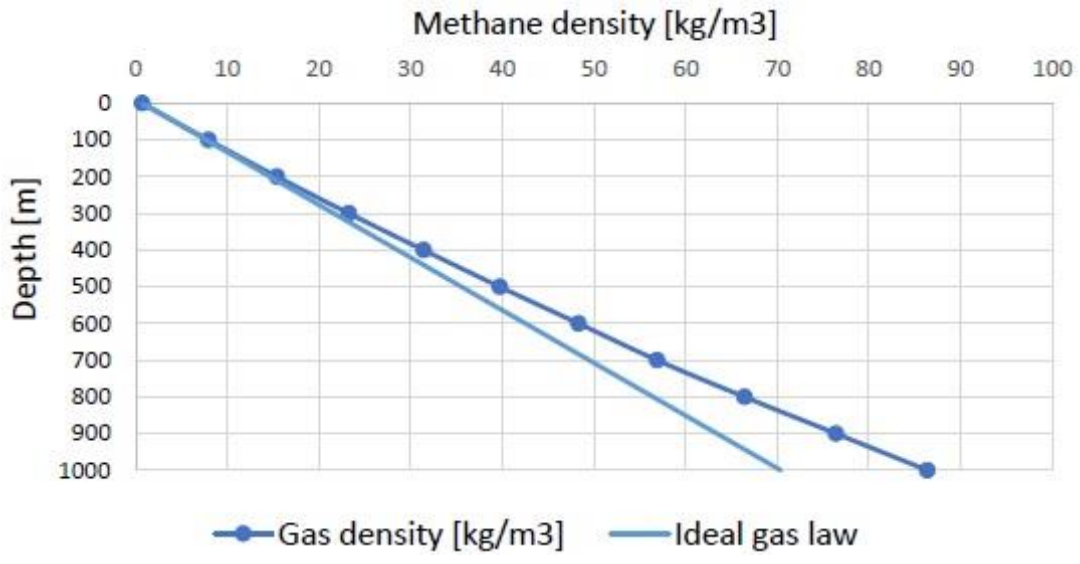

Figure 16: Methane gas density as a function of depth - Ideal gas law vs. Non-ideal gas behavior (72).

Figure 16 shows the methane gas density as a function of depth. At great depths, the non-ideal gas behavior deviates significantly form the ideal gas law. It should be noted that the

non-ideal gas behavior is taken from the generalized compressibility chart of Moran & Shapiro, which is valid for light hydrocarbon gases.

The effect of the compressibility factor is, nevertheless, neglected in this thesis. As can be seen from figure 16, this effect should be accounted for in future work. Skjetne & Olsen (2) applied, however, the Peng-Robinson equation of state to account for non-ideal gas behavior. The physical impact of assuming ideal gas behavior for deep water gas releases is further discussed in chapter 6.

4.2.2 Density Distribution of Oil Droplets

In the primary model (ref. chapter 8), the oil droplets are assumed to be incompressible, which involves a constant density distribution. Although there is no such thing as an incompressible fluid, the term may be used when the change in density with pressure is so small that it can be neglected (14).

4.2.3 Oil Droplet Size Distribution

During the DeepSpill experiment, 4 elevations were selected for measuring the oil (i.e. marine diesel) droplet size distributions. The distribution of the droplet size was observed at various distances from the release point, and a theoretical size distribution was found to be approximated by a Rosin-Rammler distribution (34). Moreover, these findings of a Rosin-Rammler distributed oil size are consistent with the experimental data provided by Karabelas (73).

In a Rosin-Rammler distribution, the complete range of droplet sizes are divided into an adequate number of discrete intervals, each represented by a mean diameter for which trajectory calculations are performed. The fraction of particles greater than a given diameter, d, is given by Equation 54, which is based on the assumption that an exponential relationship exists between the droplet diameter, d, and the mass fraction of droplets with diameter greater than d (28):

$$Y_d = e^{-(d/\bar{d})^n} \qquad \text{EQUATION 54}$$

where $\bar{d}$ is the mean diameter and n is the spread parameter. The input values (i.e. min, max, mean and spread diameter) are obtained from Johansen et al. (34), and presented in chapter 8.

4.2.4 Bubble Size Distribution

As seen from Equation 49, the bubble shape, which is characterized by the Eotvos number, is a function of the bubble size and thus the bubble diameter will influence the drag force of a rising bubble. In addition, the amount of mass transferred from a rising bubble to the surrounding water, is dependent on the bubble surface area (ref. Equation 70). A correct bubble size distribution is necessary to achieve sufficient simulation results for underwater gas plumes.

The bubble size model is governed by turbulence parameters and loss of hydrostatic pressure, which affects the material properties of a buoyant bubble. A bubble size model is implemented (ref. appendix A.1 and B.1) to account for the effect of variation in bubble diameters. The fundamental theory is based on the work of Laux and Johansen (25), and additional modifications are provided by Cloete et al. (11) and Pan (47).

A instantaneous local mean bubble diameter, d_b, is described by a transport equation (ref. Equation 55) which accounts for:

- *Loss of bubbles to downstream cells.*
- *Gain of bubbles from upstream cells.*
- *Breakup.*
- *Coalescence.*

In the work of Laux & Johansen (25) the transport equation is developed for dispersed phase calculated in the Eulerian formulation. However, this transport equation is simplified by Cloete et al. (11) to account for bubbles tracked in the Lagrangian framework:

$$\frac{\partial \rho_b d_b}{\partial t} = \rho_b \frac{d_b^{eq} - d_b}{\tau_{rel}}$$ EQUATION 55

Here $\rho_b = \alpha_b \overline{\rho_b}$ is the bubble bulk density. The relaxation time τ_{rel} is controlled by the speed of breakup or the coalescence process, and d_b^{eq} is the mean equilibrium diameter. The equilibrium diameter is the diameter a bubble achieves if it resides sufficiently long at the same flow conditions. The term at the right hand side forces the local mean bubble diameter towards its equilibrium diameter during a time frame given by the relaxation time (11). The relaxation time is given by the turbulent kinetic energy (k) and its turbulent dissipation rate (ε).

The equilibrium diameter for bubbles dispersed in a turbulent flow field, is given by Laux & Johansen (25):

$$d_b^{eq} = C_1 \alpha_b^{0.5} \frac{(\sigma/\rho)^{0.6}}{\varepsilon^{0.4}} (\mu_b/\mu)^{0.25} + C_2$$ EQUATION 56

Here, ε is the dissipation of turbulent kinetic energy, μ_b is the viscosity of the bubble phase, and σ is the surface tension between bubbles and the ambient fluid, α_b is the bubble void fraction.C_1 is a dimensionless constant ($C_1 = 4.0$), while C_2 refers to the minimum bubble size ($C_2 = 0.0001m$) (47).

The relaxation time refers to the time needed to relax to the equilibrium mean diameter, given in Equation 55. This is controlled by the speed of the breakup or the coalescence process. If the instantaneous mean diameter is smaller than its equilibrium value, Laux & Johansen (25) assumes coalescence occurs more frequently than breakup and the relaxation timescale is modelled as a characteristic timescale for the coalescence process. However, if the mean diameter is larger than its equilibrium value, breakup is expected to occur more frequently and the relaxation time becomes a characteristic timescale for the breakup process:

$$\tau_{rel} = \begin{cases} \tau_B & if\ d_b > d_b^{eq} \\ \tau_C & if\ d_b < d_b^{eq} \end{cases}$$ EQUATION 57

Laux & Johansen (25) restricts the relaxation time by a turbulent micro scale that represents the smallest time scale in a turbulent flow:

$$\tau_{rel} = |\tau_{rel}, \tau_k|_{max}$$ EQUATION 58

where the turbulent microscale is given by:

$$\tau_k = 6\sqrt{\frac{v}{\varepsilon}}$$ EQUATION 59

, and v is the kinematic viscosity of the fluid.

The breakup of a bubble occurs if the turbulent shear forces exceeds the resistive surface tension forces, where bubbles are sheared by the turbulent eddies they are exposed to. Turbulent eddies that are large compared to the bubble size is expected to not contribute to breakup, but to move the bubble around. Moreover, smaller eddies are assumed to be too

small to shear bubbles. However, eddies of comparable size may cause breakup due to their rotational frequency. Pan (47) assumes eddies which contributes most to breakup is located in the intertial subrange, and the breakup time scale is modelled as:

$$\tau_B = {d_b}^{2/3} \varepsilon^{-1/3}$$ EQUATION 60

The breakup time scale is here calculated by using an estimate for the dissipation (25): $\varepsilon = u_{rms}^3/l$ *, where* l *is an appropriate turbulent length scale and* $u_{rms} = l/\tau_e$ *is the turbulent velocity scale of eddies of size* l*. As discussed above, eddies with length scale of the same order as a bubble contributes most to breakup; thus* $l = d_b$*.*

Laux & Johansen (25) assumes that coalescing bubbles are brought into contact by turbulent velocity fluctuations. However, the bubble size model presented in the current application is based on the work of Pan (47). Here a coalescence time scale is presented, but no literature source is found:

$$\tau_C = \frac{d_b}{0.2 \cdot 6 \cdot \sqrt{\alpha_b k}}$$ EQUATION 61

4.3 Slip Velocity

Plumes consisting of gas bubbles and oil droplets will have a relative velocity to the surrounding water. Various bubble plume models (63; 61; 24; 66) assume a constant slip velocity ranging from 0.23 to 0.35 m/s. The slip velocity is defined as the velocity difference between the buoyancy driven bubble/droplet and the surrounding liquid:

$$u_{slip} = |(u_p - u_l)|$$ EQUATION 62

The implementation of the slip velocity can be found in appendix A.1 and B.1.

4.4 Gas Dissolution

In the primary model (i.e. simulation of the DeepSpill experiment), the residence time of gas in the ambient water are expected be sufficiently long to make the effect of gas dissolution significant. As the solubility of gas is increased due to high ambient pressures, dissolution of gas into the sea water may cause a significant reduction in the buoyancy flux (15). Additionally, the bubble surface area is assumed to have great impact on the mass transfer rate. The mass transfer rate is defined as the mass transferred from the rising dispersed bubble phase into the surrounding ocean, in kg/s. This section is essentially based on the work of Skjetne & Olsen (2).

4.4.1 Governing Equations and Assumptions

Natural gas is in principal a multi-component gas. From a simulation point of view, this is a much more complex problem than to consider the bubbles as pure methane. In reality, the natural gas is often dominated by a large ratio of methane (2). Thus, release of pure methane may in principle be valid for release of natural gas.

The process of gas dissolution into the ambient water is governed by the concentration difference of the specific gas component at the bubble surface and in the surrounding liquid. The concentration at the bubble surface is given by the solubility of the gas species. Thus the mass flux J, through the bubble surface, may be expressed by the following expression (2):

$$J_i = k_i(C_i^{sol} - C_i^w) \qquad \text{EQUATION 63}$$

In Equation 63, $k_i(m/s)$ refers to the mass transfer coefficient of species i, $C_i^{sol}(kg/m^3)$ is the solubility of gas in seawater, and C_i^w (kg/m^3) is the ambient ocean concentration of dissolved gas.

Results presented by Skjetne and Olsen (2), reveals that deep water plumes lose most of their buoyancy through dissolution of gas into the surrounding ocean. This is mainly due to the fact that seawater is under saturated with gas, and thus has a large capacity to dissolve gas. However, in the present work, the transient concentration of methane transferred to the surrounding water is set to zero ($C_i^w = 0$). This assumption may provide a higher rate of gas dissolution compared to real gas plumes. There is, however, possible to add the fraction of the dissolved methane to the secondary phase (i.e. sea water) through the species model in ANSYS FLUENT and user-defined functions. Impacts of this assumption is further discussed in chapter 6.

As noted above, the natural gas is assumed to have the same properties as pure methane, and Equation 63 might be written as follows (2):

$$J = k_{CH_4}\rho_{H_20}n_{CH_4}^{sol}\frac{M_{CH_4}}{M_{H_20}}$$ EQUATION 64

Here n denotes solubility, M denotes molar weights, and the ambient ocean concentration of dissolved gas is neglected ($C_i^{sol} = Y_{CH_4}^{w} = 0$).

Lekvam & Bishnoi (16) did several solubility experiments in which concerned dissolution of methane in pure water. The ratio of solubility was measured at low temperatures and high pressures. However, the solubility of gas varies with pressure, temperature and salinity of the sea. To account for the salinity, adjustments in relation to Lekvam & Bishnoi are required. Millero (17) studied the activity coefficients of non-electrolytes in sea water by using the Setschenow equation:

$$ln(S^0/S) = ln\gamma_N = k_s m$$ EQUATION 65

where S^0 and S is the solubility in water and solution, k_s is the salting coefficient of the non-electrolyte (N), γ_N is the activity coefficient, and m is the molality. Millero (17) presents the value of the salting coefficient ($k_s = 0.319$). In the present work, k_s is the salting in coefficient (negative sign). Thus, Equation 65 may be rewritten to account for low temperature ($5°C$), loss in hydrostatic pressure, and salinity (2):

$$n_{CH_4}^{sol} = n_{CH_4}^{sol_0}e^{-0.319s/M_{NaCl}}$$ EQUATION 66

In the above equation, s is the salinity of sea water which in the North Sea typical is 35 moles/kg. The solubility, n_{CH4}^{sol0}, is given for different pressures and temperatures presented in Lekvam and Bishnoi (16). A rounding of these values is implemented in the user-defined function, "Primary_Model.c" (ref. appendix B.1). The solubility of lower hydrostatic pressures than presented by Lekvam & Bishnoi, is tuned after experimental results presented in Johansen et al. (34).

4.4.2 Clean Bubbles vs. Bubbles Contaminated by Surfactants

The mass transfer rate is expected to be highly dependent of the bubble surface area. However, the bubble surface might be affected by surfactants and thus decrease the mass transfer rate. The composition of the bubble surface is therefore of interest.

In the present work, surfactant refers to compounds that lower the surface tension between the dispersed phase and the ambient fluid (14). These substances interact with the bubble surface to change its properties (18). Surfactants might immobilize the surface and decrease the internal circulation inside the bubble. As the rate of mass transfer is dependent of sufficient internal circulation, contaminated bubbles tends to be less affected by gas dissolution relative to clean bubbles. This effect makes itself evident in the journal article of Skjetne & Olsen (2), figure 17.

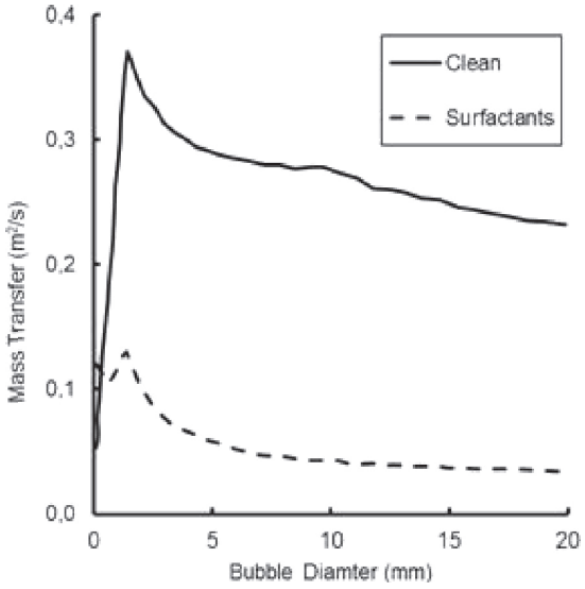

Figure 17: Mass transfer coefficient for methane bubbles for clean and contaminated surface (2).

In the current application the bubbles are assumed to be influenced by surfactants. Clift, Grace, & Weber (18) describes the phenomena of surfactants as follows: "One must accept the presence of surface-active contaminants in most systems of particle importance. Even though the amount of impurity is so small that there is no measurable change in the bulk fluid properties, a contaminant can eliminate internal circulation."

Zhang & Xu (74) developed an expression of the mass transfer coefficient for bubbles with rigid surface contaminated by surfactants. The expression is based on the theory for convective crystal dissolution and parameterization of the Sherwood number under various conditions by Clift, Grace, & Weber (18). The mass transfer coefficient was found to be as follow:

$$k_{CH_4} = \left[1 + (1 + Pe_c)^{1/3}\left(1 + \frac{0.096Re^{1/3}}{1+7Re^{-2}}\right)\right]\frac{D_{CH_4}}{d_b}$$ EQUATION 67

,where d_b is the bubble diameter and the Reynolds number (Re) is defined as:

$$Re = \frac{(2r_b)u_{slip}\rho_f}{\mu}$$ EQUATION 68

In Equation 68, r_b is the radius of the dissolving bubble, u_{slip} is the slip velocity (ref. section 4.3), ρ_f is the ocean density and μ the viscosity of seawater. The compositional Peclet number (Pe_c) (ref. Equation 67), characterizing the relative importance of flow versus diffusion, and is defined as:

$$Pe_c = 2r_b/D_{CH_4}$$ EQUATION 69

The last property, on the right hand side of Equation 67, is the diffusivity of methane (D_{CH_4}). Diffusivity is defined as a measure of the ability of a substance to transmit a difference in temperature (14). The mechanism of diffusion of gases and liquids is extremely complicated and generalized theories are not available. Furthermore, only limited experimental results are available in the literature. However, since the temperature of sea water is constant in the present work, the diffusivity of methane is assumed to be a constant value, $D_{CH_4} = 1.3 \cdot 10^{-9}\, m^2/s$ (19).

4.4.3 Mass Transfer Rate

The final expression of gas dissolution is the product of the mass flux (ref. Equation 64) and the bubble surface area. The characteristic bubble shape is given by the Eotvos number (ref. section 4.1.1), but for simplicity the bubble shape is approximated as a sphere (20). The mass transfer rate can thus be calculated by the following expression, in kg/s:

$$\dot{m} = \pi d_b^2 k_{CH_4} \rho_{H_2O} n_{CH_4}^{sol} \frac{M_{CH_4}}{M_{H_2O}}$$ EQUATION 70

The effect of gas dissolution is implemented in the UDF "Primary_Model.c" (ref. appendix B.1).

4.5 Ocean Current

Sea water in motion is often referred to as currents. The primary generating factors of currents are wind and differences in water density, caused by variations in temperature and salinity. Ocean currents generated by these factors are affected by the extent and location of land, ocean depth, underwater topography, and by rotation of the Earth (80).

4.5.1 Wind Driven Currents

The stress of wind blowing across the sea causes a surface layer of water to move (80). Waves or ripples provide enough surface roughness necessary for the wind to interact with the sea surface. Without rotation of the Earth, frictional coupling between the moving air and the ocean surface would push only a thin line of water in the same direction as the wind (75). The surface layer of water would drag the layer beneath it, putting it into motion due to internal water friction. This interaction may decrease with ocean depth as each ocean layer moving at a slower speed than the layer above. As the Earth rotates the moving ocean layers are deflected. This effect is commonly known as the Coriolis effect. Each layer of water is put into motion by the layer above and slowly shifts direction due to rotation of the Earth.

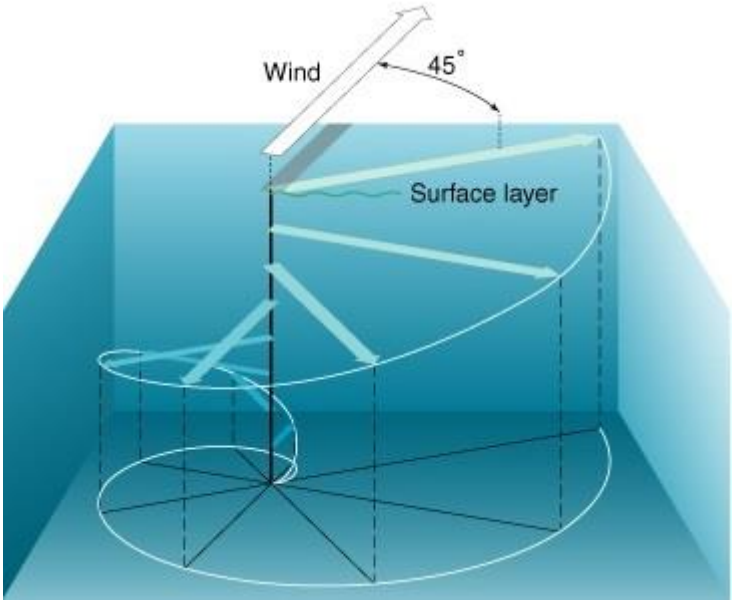

Figure 18: The Ekman spiral describes how the horizontal wind sets surface water in motion (75).

In figure 18, the Ekman spiral indicates that each moving layer is deflected to the right of the overlying layers movement; hence, the direction of water movement differs with increasing depth.

4.5.2 Density Driven Currents

The density of water varies with salinity, temperature, and pressure. At a given depth, the differences in density are essentially due to differences in temperature and salinity. In regions of high density, the water surface is lower than in regions of low density. Due to these density differences, water masses may flow from an area of higher water (low density) to one of lower water (high density). As the Earth is rotating, the flow is deflected by the Coriolis force and toward the right in the Northern Hemisphere, or toward the left in the Southern Hemisphere (30). This movement of water masses between subsurface density fields that are deflected by the Coriolis effect, is called geostrophic currents. The larger density gradients (rate of change with distance) the stronger the geostrophic currents are.

4.5.3 Modelling Ocean Currents

Oil droplets are expected to be strongly affected by ambient ocean currents due to their relative low buoyancy. One of the main objectives of this thesis is to create a reasonable method of simulating the effect ocean currents. Three different approaches are tested out, discussed and presented in the following sections.

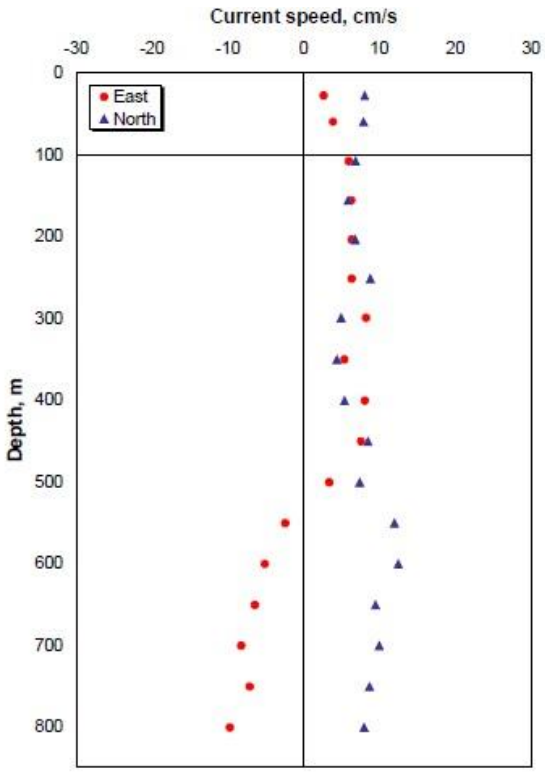

Figure 19: Ocean current profile from the start of marine diesel (oil) discharge (34).

During the DeepSpill experiment, ocean current data were obtained from two instruments, one upward looking ADCP (Acoustic Doppler Current Profiler) mounted in a rig anchored on the seabed, and one downwards looking ADCP (34). The current profile from start of the marine diesel (oil) experiment is presented in figure 19, where red circles indicates east direction and blue triangles indicates north direction.

For more information about how the ocean currents were measured, see Johansen et al. (34).

Method 1 – Velocity Inlet as Boundary Condition

The first attempt in recreating the ocean currents measured during the DeepSpill experiment (ref. figure 19), was to define the vertical water boundaries as velocity inlet. The various current velocities were given from two different UDFs, in which represents the velocities in x-direction (east/west) and y-direction (north/south) (ref. appendix B.3). The exact current velocities are obtained from ScanIt, which is software for extracting data from scanned graphs.

The geometry was in this method divided into four bodies, two water bodies and two air bodies (ref. figure 20). The idea behind this division was to achieve different boundary conditions (i.e. walls, interior, and velocity inlet) for the air and water bodies. The inner water body, in figure 20, defines the region for where the multiphase plume was expected to rise, where a denser grid was given. A large volume of air and water was placed around the inner body, in order to maintain the mass balance. The boundaries of the inner water body were defined as velocity inlets controlled by the UDFs. Below, the geometry created in ANSYS Workbench is presented.

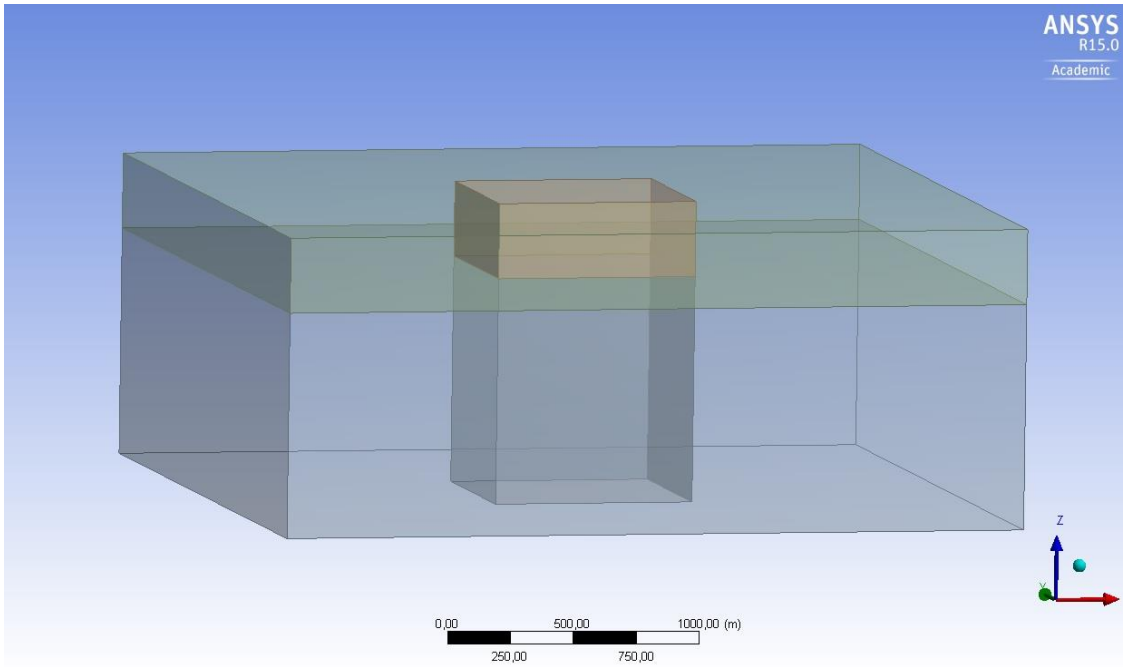

Figure 20: Geometry for Method 1 and 2.

The idea was to run the simulation, without any injection of oil and gas particles, until steady state ocean currents had developed. However, this method was not working as first expected and were quickly discarded based on observations from the velocity contour plot, figure 21. Figure 21 indicates that the momentum of the simulated ocean currents is too small relative to the internal friction of ocean water. For this reason, the sea current velocities are quickly reduced and "dead-zones" containing no water movement occurs.

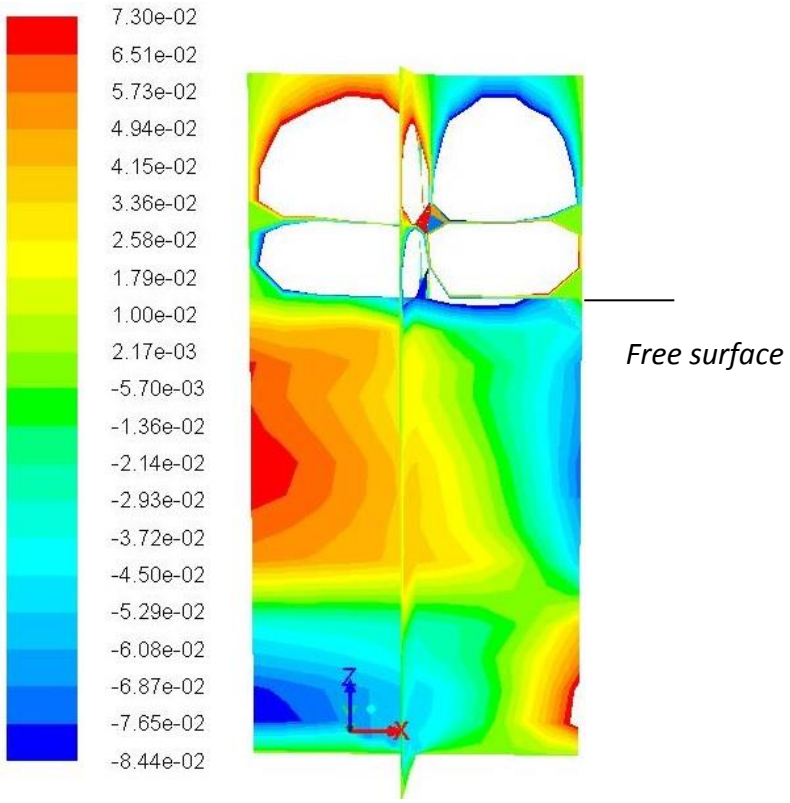

Figure 21: Method 1 – Velocity contour plot in x-direction.

Method 2 – Patching in Velocity Regions

ANSYS FLUENT allows the user to patch in different initial values of flow variables into various regions of the domain. The different ocean current velocities (ref. figure 19) were patched into the plume region (i.e. inner water body of figure 20). However, this method was not expected to provide sufficient results. As seen from figure 22, the different layers of ocean currents contribute with different velocities and directions. Movement of water masses through the ocean is slowed by internal friction in the water, by the surrounding fluid moving at different velocities.

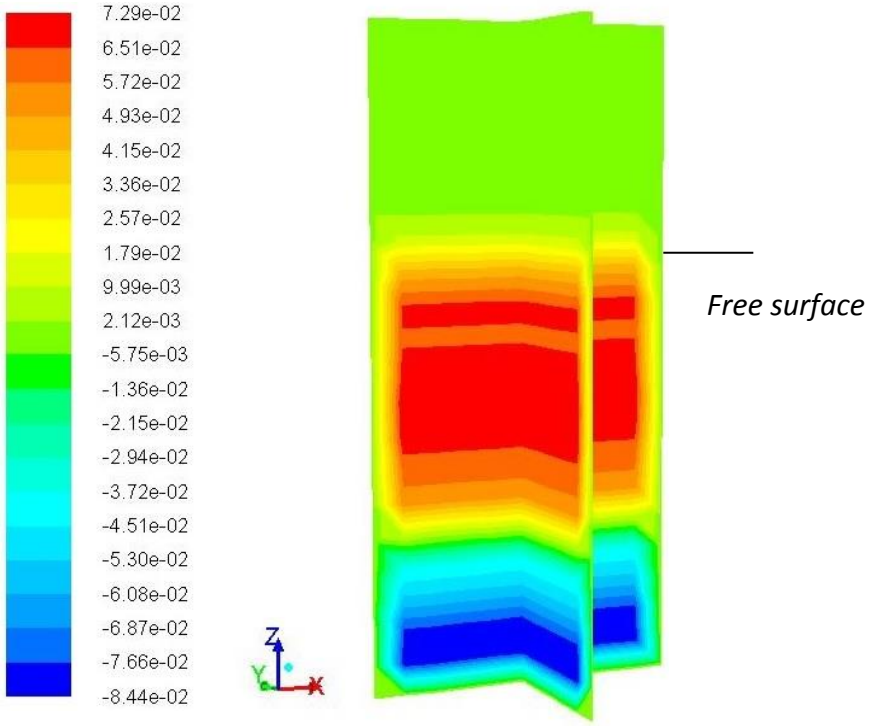

Figure 22: Method 2 - Initial patched ocean currents.

A faster-moving fluid layer tends to drag along a slower-moving layer, and a slower-moving layer will tend to reduce the speed of a faster-moving layer. This momentum transfer is a product of turbulence that moves kinetic energy to smaller scales, and increases the turbulence in the surrounding water (30). Figure 23 illustrates this effect. The various ocean layer velocities are almost effaced due to this momentum transfer:

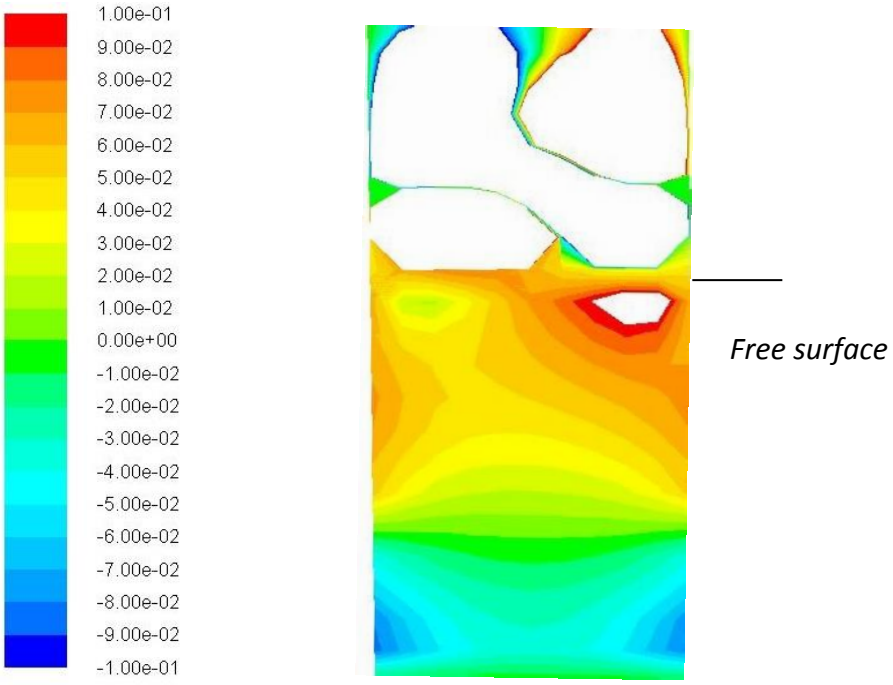

Figure 23: Method 2 - Ocean currents after quasi steady state conditions.

Method 3 – Particle Acceleration

In the presence of ambient ocean currents, the motion of bubbles and droplets are affected by additional body forces (ref. section 3.6.2:I). In this approach a user-defined drag force estimates the effect of ocean currents on each bubble/droplet. An acceleration term is returned to each particle, in which replaces the effect of a moving continuous phase. Thus, ocean currents are not simulated by movement in the continuous phase. However, the effect of cross-flow is estimated by a user-defined drag force, which in turn determines the degree of acceleration returned to each bubble/droplet. In order to implement this in ANSYS FLUENT, the macro "DEFINE_DPM_BODY_FORCE" may be used. This macro returns the acceleration term due to the drag force (in m/s^2) to the ANSYS FLUENT solver (76).

As discussed in section 3.6.2:I, ANSYS FLUENT predicts the trajectory of a discrete phase particle (e.g. droplet or bubble) by integrating the force balance on the particle, which is tracked in a Lagrangian reference frame. This force balance equates the particle inertia with the forces acting on the particle, and may be written in x-direction (ref. Equation 30):

$$\frac{du_p}{dt} = F_D(u - u_p) + \frac{g_x(\rho_p - \rho)}{\rho_p} + F_x$$

In the current application, the drag force exercised on buoyant gas bubbles is dependent on the Eotvos number (ref. section 4.1.1). For simplicity, this drag force is assumed to be valid for both oil droplets and gas bubbles under the influence of ocean currents. This assumption is not expected to alter the simulation results to a large extent, as a test simulation for which the oil droplets applied the spherical drag law (ref. section 4.1.2), under the influence of sea currents, provided almost identical results. The drag force and the related drag coefficient are given in section 4.1.1.

As the drag term is a function of the relative Reynolds number (ref. Equation 33), an expression of slip velocity is necessary. This slip velocity is the relative velocity between the bubble/droplet and the computed continuous phase velocity in addition to the specified ocean current velocity (ref. figure 19), at different elevations in the domain. The slip velocity is given by Equation 71, which indicates the ocean currents contributes to particle movement in x (east/west) and y (north/south) direction.

$$u_{slip} = |\boldsymbol{u}_p - \boldsymbol{u}_l| = \sqrt{(u_p - u_w - u_c)^2 + (v_p - v_w - v_c)^2 + (w_p - w_w)^2} \qquad \text{EQUATION 71}$$

, where u_p is the bubble/droplet velocity in x-direction, v_w is the surrounding water velocity in y-direction, and v_c defines the ocean current drift velocity in y-direction (north/south). Thus, for each bubble/droplet rising through the domain, acceleration terms on the following form are returned, in x- and y-direction:

$$accel_x = F_D \cdot u_c \quad \text{and} \quad accel_y = F_D \cdot v_c \qquad \text{EQUATION 72}$$

As explained above, this method of simualting ocean currents are not based upon movement in the continuous phase, as the case are for real cross-flows. However, the effect of ocean currents on the discrete particles are estimated by a user-defined drag force. This is further discussed in section 6.3.4 and section 8.3.4.

See appendix B.2 for implementation of the user-defined ocean current function.

Chapter 5

Numerical Accuracy and Computational Efficiency

The main goal of a CFD simulation should be the achievement of reliable and accurate results. However, this could result in a computational expensive model that requires additional simulation time. Therefore, a balance between computational time and numerical accuracy should be achieved. In this chapter, various factors with significant impacts on this subject are evaluated.

5.1 Grid

The gridding process is usually decisive for both the accuracy of the results and the computational time. Generation of numerical grid can be quite complicated and time consuming, dependent on the method available in the preferred CFD-package. In ANSYS FLUENT the user may choose between three mesh generation approaches.

5.1.2 Methods of Mesh Generation

I. Generating Mesh in ANSYS Workbench

The first approach is to create the complete grid refinements in ANSYS Workbench. Here the user may play around with different element sizes, growth rates, and bias ratios to achieve a desired mesh. This process can be quite complicated due to the required preliminary knowledge about the expected solution, as regions of large gradients (e.g. pressure and velocity) should be covered by a denser grid. In the current application, this method was found very time consuming and not very efficient.

II. Isovalue Mesh Adaptation

A second choice is to refine the mesh relative to specific flow variables selected in the solver ANSYS FLUENT. User-defined macros or TUI-commands are created to obtain transient mesh refinements in regions with large gradients (77). For instance, multi-phase plumes are identifiable by regions of relatively high particle concentration and high velocity fluids. As these regions contain large gradients of important flow quantities, isovalue mesh adaptation may be appropriate in simulation of a rising multi-phase plume affected by ambient ocean currents.

Various methods of isovalue mesh refinement exist. The simplest strategy is to subdivide the cells, for which every "parent cell" is divided into "child cells" (ref. figure 24). For every parent cell four child cells are created, which implies the overall mesh topology remains the same (78). This method of refinement is referred to as h-refinement. In the present work, isovalue adaptation is tested out using the h-refinement strategy (ref. figure 25).

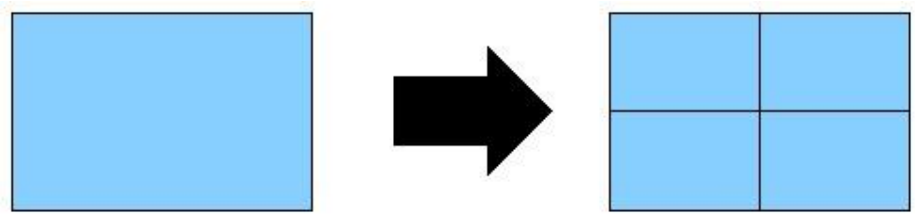

Figure 24: The strategy of h-Refinement for 2-D mesh (78).

This process might be computationally expensive dependent on the number of required refinements. However, in contrast to earlier versions of ANSYS FLUENT, mesh adaptation is now supposed to work in parallel simulations (i.e. simulation with multiple CPUs in ANSYS FLUENT 15.0.0). This approach of generating mesh was the first choice in the current application, as preliminary knowledge about the expected plume path is not required. Figure 25 illustrates the isovalue mesh adaptation based on the void fraction in each control volume, for a rising bubble plume.

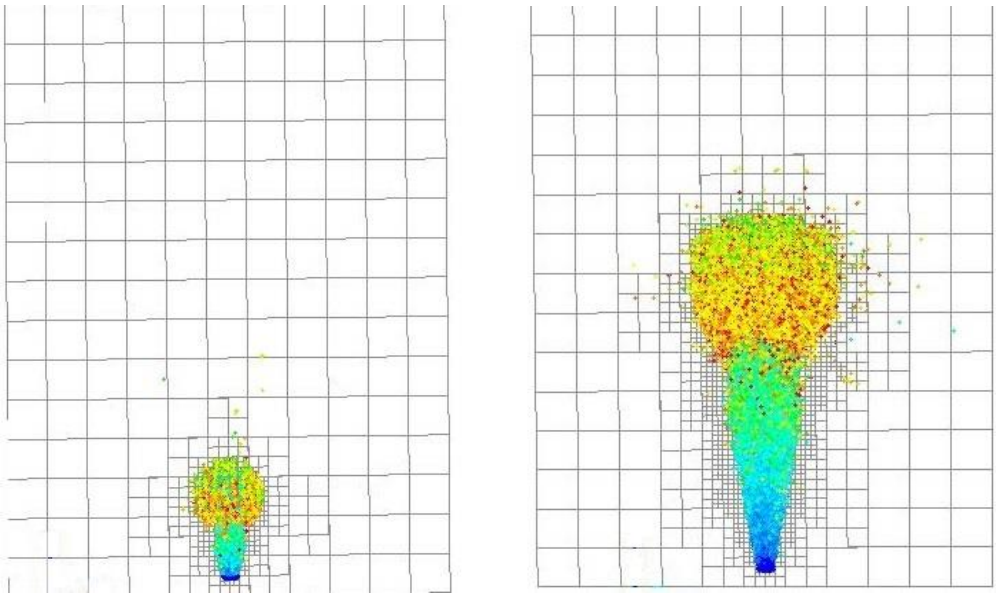

Figure 25: Isovalue mesh adaptation based upon the DPM concentrations in each control volume, for a rising bubble plume.

Figure 25 shows a test simulation of the Rotvoll experiment, in which showed similar results as simulations applying a fixed mesh (ref. chapter 7).

However, the situation was different when simulating the DeepSpill experiment. Some minutes (in flow time) after the gas phase reached the water surface divergence through momentum was detected and the simulation was crashed. The problem may indicate the presence of numerical instabilities that creates artificial velocities in regions where the grid size differences of two neighboring cells are too large. This generation of velocities seems to be associated with the Geo-Reconstruction Scheme and its piecewise-linear approach, mentioned in section 3.6.3:A. Moreover, it is later reported that a uniform grid size distribution should be applied to interfaces captured by the VOF model (personal communication with Senior Scientist Jan Erik Olsen, SINTEF Materials & Chemistry, 30/04-2014). The effect of these artificial velocities is illustrated in figure 26.

Left hand side of figure 26 shows the water velocity in z-direction, right after the first gas bubbles has penetrated the water surface. A negative z-velocity is starting to develop, which seems to increase with time (right hand side of figure 26). As the simulations are calculated through external computers (ref. section 7.1.6), the user can only analyze the solution for specified time steps, which makes it difficult to point out the exact impact of this numerical instability. However, the water masses may start to push the discrete phase towards the seabed, making the numerical solution unstable. However, this issue did not affected the Rotvoll model due to significant shorter flow time.

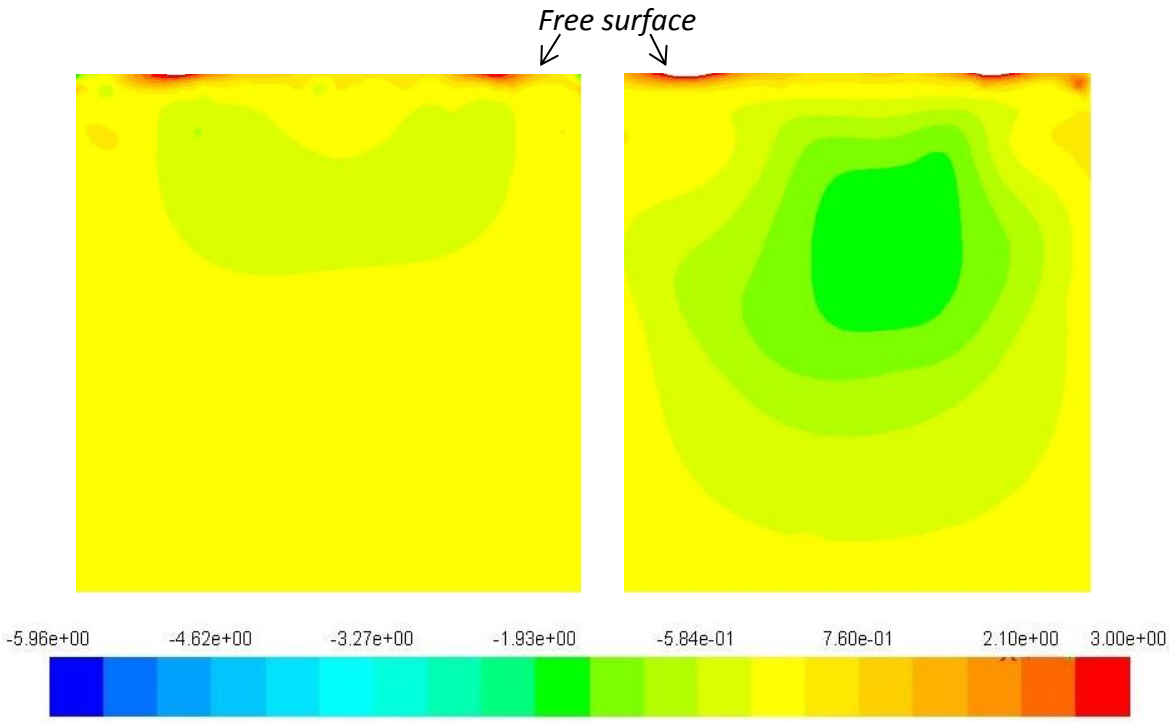

Figure 26: Possible effect of non-uniform grid distribution in the interface region.

Fortunately, there might be a solution to this problem in combining isovalue mesh adaptation and uniform grid size distribution of the water surface. In ANSYS FLUENT, the user has the opportunity of selecting regions where isovalue mesh adaptation is preferred to be active. By dividing the water surface into one separate body and then manually refine it, the generation of artificial velocities should disappear. However, this solution was recently discovered and is not tested out for the current application.

III. Region Refinement in ANSYS FLUENT

The third method concerns manually refinements of desired domain. This is done in the solver, ANSYS FLUENT, before any calculations are started. A trial and error process are necessary due to required preliminary knowledge of the expected solution. In the current application, this process was found significantly easier and less time consuming than the gridding process in ANSYS Workbench (ref. section 5.1.2:I). Region refinement in ANSYS FLUENT is a type of h-refinement, discussed in the section above. The only difference is that the refinements are done manually for regions were the dispersed phase is expected to rise. However, this might increase the computational time compared to isovalue mesh adaptation, due to redundant grid cells. This method requires a uniform grid size distribution in the water surface region, as well (ref. section 5.1.2:II).

5.1.3 Grid Size and Mesh Independence

The grid size is crucial in case of accurate and computational economic simulations. The primary procedure to ensure numerical accuracy is to check for mesh independence. This implies that the mesh should be sufficiently dense so additional refinements do not offer significant improvements in the solution (5). One way to test for mesh independence is to run cases of different grid size and compare the flow variables of interest for each case. However, the computational time may exceed the desired simulation time when additional refinements are desired/required. A mesh independence study is carried out by Pan (47), when simulating the Rotvoll experiment. This study is based upon an initial grid size of 5 cm, which is the same grid size used in the work of Cloete et al. (11).

In the primary model (i.e. DeepSpill experiment), a finer grid size is given in the jet zone region, as this region contains flow variables of large gradients. In the buoyant plume region (ref. chapter 2), a significant larger grid size is sufficient, as the change in ambient flow variables is expected to be almost constant. A complete mesh independence study is not conducted for the primary model, due to the major amount of computational time required. However, an unpublished SINTEF-report and personal communication with Senior Scientist Jan Erik Olsen (SINTEF Materials & Chemistry, 03/04-2014) revealed that an amount of 80 cells along the plume center line is enough for sufficient results, when simulating deep water gas releases applying the coupled DPM and VOF model approach. Grid size is further discussed in chapter 8.

5.2 Solver Algorithm

As noted in section 3.4, the PISO algorithm is applied in the current application. This algorithm is part of the SIMPLE family of algorithms. One limitation of the SIMPLE algorithm is that new velocities do not satisfy the momentum balance, and the calculations must be repeated until the balance is satisfied (28). However, in the PISO algorithm the repeated calculations are moved inside a solution stage where the pressure is corrected. After a few PISO loops, the corrected velocities satisfy the continuity and momentum equations more closely. This method is called "neighbor correction", and may dramatically decrease the number of iterations required for convergence for transient problems. The PISO algorithm is based on higher degree of the approximated relations between the correction for pressure and velocity, which in most cases provides improved accuracy. However, higher order schemes can be less stable, and they may increase computational time (55). For more information about the PISO algorithm see Versteeg (55).

Chapter 6

Discussion of Assumptions

In most practical engineering applications, a large number of simplifications and assumptions need to be made to enable the computational fluid dynamic solution of the differential equations to fit a realistic physical situation. Moreover, commercial CFD packages make it possible to develop models without fully understanding a large number of assumptions. These assumptions can lead to incomprehensible results which can be difficult to interpret properly.

6.1 Assumptions Discrete Phase Model (DPM)

The Discrete Phase Model (DPM) includes a number of assumptions. In the following sections the most important assumptions are discussed.

6.1.1 No Volume Occupied by the Discrete Phase

DPM particles does not take up any volume within the domain (ref. section 3.6.2:A). As a consequence, the mass of water within the plume itself is larger than it might be for real plumes, and this increased mass will have to be accelerated by the correct volume of gas/oil (11). Additionally, according to ANSYS FLUENT Theory Guide (28), the DPM method is only valid for void fractions smaller than 10-12%. In case of denser void fractions, the momentum transfer between the discrete phase and the surrounding fluid can be over-predicted (5).

Cloete (5) calculated the total volume violating the DPM criterion, in which was found small enough to be negligible. This assumption is further discussed in section 7.3, where the void fractions along the plume center line for different flow rates are presented.

6.1.2 No Turbulent Wake Effects and Particle-Particle Interaction

In real blowouts, the rising bubbles and droplets interact through collisions and various wake effects (29). As the dispersed phase and the continuous phase employs a two-way coupling, the discrete particles leave no turbulent vortex structures in their wakes, and can only transfer momentum as a source term to the continuous phase (ref. section 3.6). As mentioned above, the DPM particles do not occupy any volume, which implies that two particles would simply pass straight through each other if they were to cross (5). The effect of no particle-particle interaction may provide a more dispersed plume, due to less coalescence of bubbles/droplets.

Pan (47) included a user-defined bubble-wake model, which resulted in a more dispersed and lower velocity plume that showed quite consistent results compared with experiments. This bubble-wake model is not included in the present work, thus the simulation results are expected to provide a slightly higher plume center velocity due to lower degree of horizontal extension.

6.1.3 Bubble Size Model

The bubble size model, accounting for breakup and coalescence mechanisms, is the same as the one applied by Cloete et al. (11) and Pan (47). The model was developed by Laux and Johansen (25), where the bubble size is governed by material properties and turbulence parameters (ref. section 4.2.4).

It is reported that the first bubble observed reaching the surface, during the Rotvoll experiment (ref. chapter 7), was one single large bubble (personal communication with Senior Research Scientist Paal Skjetne, SINTEF Materials & Chemistry, 20/03-2014). However, in the current bubble size model this effect is not possible to account for. In cases where the instantaneous mean diameter is smaller than the equilibrium diameter, coalescence occurs. Moreover, if the instantaneous local diameter is larger than the equilibrium diameter, the bubble size model accounts for breakup. This may affect the rise time of the dispersed bubbles as large diameter bubbles may raise faster, due to larger buoyancy force. This is further discussed in chapter 7.3.

6.2 Turbulence

In the current application, the Standard $k - \varepsilon$ model is used for modeling the effect of turbulence in the atmospheric air and ambient water. This is possible not an ideal turbulence model, especially due to the assumption of isotropic turbulent viscosity (ref. section 3.5.4:A). Cloete (5) claims that results obtained with the RSM were completely unrealistic, and that the Standard k-ε model is the most suited turbulence model when simulating underwater bubble plumes.

6.2.1 Assumptions of the Standard k-ε Model (SKE)

A. Isotropic Turbulence

The main assumption of the SKE model, in combination with the Random Walk Model, is the modeling of isotropic turbulence discussed in chapter 3. Different literature sources (ref. section 3.5.3) claims that the assumption of isotropic turbulence is incorrect for underwater bubble plumes. However, isotropic turbulence is incorporated in the majority of the available literature. The exact impact of this assumption is hard to determine, but it may increase the horizontal extension of a bubble/droplet plume and thus decrease the plume rise time, due to the effect of isentropic turbulent viscosity. This is further discussed in section 7.3.

B. No Surface Damping

It is reported by Soga & Rehmann (79) that there is an increase in turbulent dissipation at the free water surface for surfacing bubble plumes. Moreover, Cloete et al. (11) points out that: "When turbulent eddies approach and locally lift a free surface, there is an increase in the rate of the turbulent energy cascade, which ultimately leads to increase TKE dissipation rate". However, the Standard k-ε model does not account for turbulence damping in the vicinity of a free surface. On the left hand side of figure 27, turbulent eddies are illustrated without the effect of surface damping. The turbulent eddies are modelled straight through the water surface, without any increase of the turbulent dissipation (ε) and decrease in the turbulent kinetic energy (k). On the right hand side, the effect of surface damping is presented.

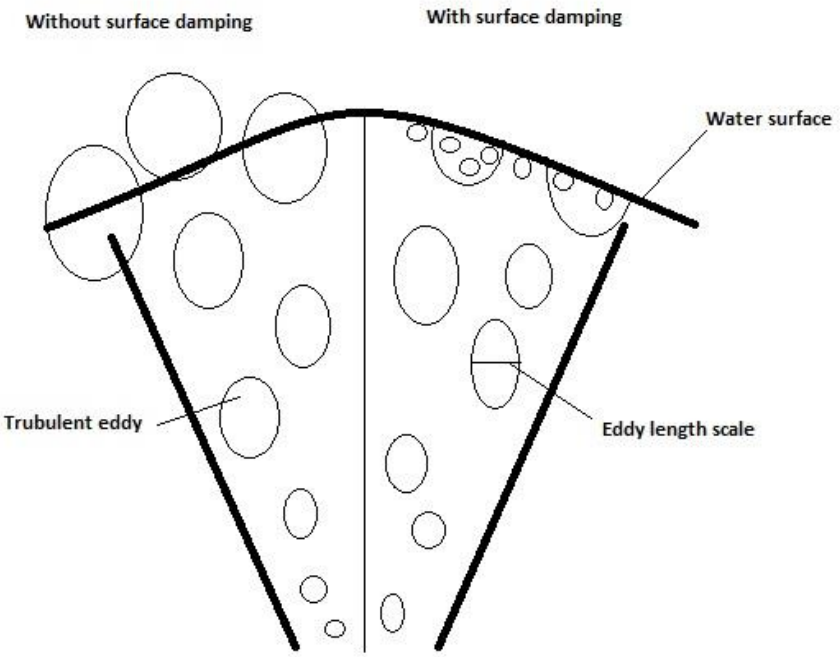

Figure 27: Effect of surface damping vs. no surface damping.

Being aware that the epsilon transport equation (ref. Equation 26) is actually the eddy length scale equation, a model may be implemented to assure the length scale is zero at the free surface. Pan (47) presents a proportional relationship between the length scale and the physical distance to the free surface. Thus, a new value of ε may be presented to override the ε in a computational cell near the free surface.

Including the effect of surface damping may affect the surface flow velocity, due to the effect of lower viscosity fluid. As the turbulent kinetic energy is "killed" and the turbulent dissipation is increased by calculation of a new value of ε, the turbulent viscosity is decreased (ref. Equation 28).

The effect of surface damping is presented in section 7.3, provided by Pan (47). Surface damping effects is, however, not implemented in the current application. As the gas phase are expected to be completely dissolved in the surrounding ocean and the turbulence of oil droplets is much less pronounced near the free surface, this effect is assumed to be not as serious. However, in case of shallow water and/or higher gas flow rates the effect of surface damping should be applied.

C. No Bubble Induced Turbulence

The Standard k-ε model is developed for constant density flows only. As bubbles and/or droplets are driven by buoyancy forces due to different densities, large gradients in volume fraction of the discrete parcels may occur. This effect is not accounted for in the SKE. Pan (47), however, explored the contributions of buoyancy force to the production and destruction of turbulence. By adding turbulence intensity in unstable stratified flow regions and suppress turbulence in stable stratified flows, the effects of buoyancy may be accounted for. Pan (47) included the effect of bubble induced turbulence to simulations of gas-stirred ladle. The results provided a slightly higher and more accurate prediction of the radial and axial water velocity.

Due to the small differences in velocities when the buoyancy-modified turbulence model was included, this model is neglected in the current simulations. Moreover, the large numerical cells necessary to simulate the DeepSpill experiment may contribute to even less effects of the bubble induced turbulence. Grid size and turbulence is further discussed in chapter 8.

6.3 Additional DeepSpill Assumptions

6.3.1 Ideal Gas Behavior

As discussed in section 4.2.1, the effects of non-ideal gas behavior are neglected in this thesis. As seen from Figure 16, the methane gas behavior deviate significantly from the ideal gas law at large depths.

When a blowout takes place in deep water, the pressure and temperature dependent compressibility factor, Z, should be introduced. The Z-factor contributes to higher bubble density, which may lead to lower specific volume of the discharged gas than predicted by the ideal gas law (34). Further, the lack of non-ideal gas behavior may decrease the mass of each bubble parcel and thus lowering the rate of gas dissolution (ref. appendix B.1). The effects of ideal gas behavior for deep water blowouts are further discussed in chapter 8.

6.3.2 No Species Transport

As mentioned in section 4.4, the transient concentration of methane transferred to the surrounding ocean is set to zero in the current simulations ($C_i^{sol} = Y_{CH_4}^w = 0$) (ref. Equation 64). This means the sea water surrounding each bubble are not experiencing any saturation. In case of water saturation, the gas dissolution rate of a bubble is stopped and thus the overall mass transfer rate is lowered. However, it is expected that the effect of no ocean saturation is small.

6.3.3 Underwater Oil Droplet Behavior

The turbulent zone created at the release point (ref. chapter 2) may cause the oil to fragment into droplets, and the size distribution of the oil drops is found to be approximated by a Rosin-Rammler distribution (ref. section 4.2.3). However, no literature is found regarding the frequency of breakup and coalescences of oil droplets. As oil is a highly viscos fluid, the breakup of oil droplets due to turbulence is expected to be at a much lower frequency than gas bubbles. Therefore, the effect of breakup and coalescence of oil droplets are neglected due to the lack of good literature sources on the subject.

In addition, the effect of density stratification is disregarded (ref. section 2.3.1). Even small stable density gradients in the ambient sea water may cause trapping of the plume. This effect is expected to increase the residence time of oil in the surrounding water. The oil may finally arrive at the sea surface due to the buoyancy of individual droplets (34).

6.3.4 Including Ocean Currents

The most appropriate way of modeling ocean currents was found by implementation of a user-defined function in which returns an acceleration term to each bubble/droplet, estimated by a drag force and the surrounding ocean current velocities (ref. section 4.5.3:Method 3). One assumption of this modeling choice is that water surrounding the entrained water plume actually is stationary. This implies that the discrete phase determines the path of the entrained water plume all the way towards the ocean surface. In a real blowout situation (ref. figure 4 and figure 5), ambient ocean currents force the entrained water plume to bend, and as the ambient ocean currents increases gas bubbles and oil droplets start to separate from it. This assumption may affect the point of oil and gas separation, due to higher turbulent viscosity in moving water. However, it is difficult to determine the exact impact of this assumption at the present time.

Chapter 7

Validation Model: Rotvoll Experiment

At Statoil's Research center in Norway, an experiment of underwater gas releases was conducted in 1997. The experiment is named "Rotvoll" and presented in the report of Engebretsen et al. (54). The main objectives were to investigate the surface currents near the bubble plume and gas dispersion above the surface (54). A series of experiments were conducted in a rectangular 7 m deep basin with a surface area of $6 * 9$ m. The basin was filled with water and air was released at the bottom, at gas rates of 83, 170 and 750 Nl/s (11). In order to reduce the vertical momentum created by the bubble plume, an arrangement in front of the release point was installed. Due to this momentum breaker, the fluctuations in the gas flow and the length of the inlet jet was minimized. In 2009, Cloete et al. (11) used a coupled DPM and VOF model to recreate the experimental data obtained by Engebretsen et al. (54). In the report of Cloete et al. (11), these simulation results are compared with the empirical data obtained by Engebretsen et al. (54).

In this chapter, the Rotvoll experiment is simulated in order to validate the general model set-up of the primary model (ref. chapter 8). Simulation results of this validation model are compared with experimental data from Engebretsen et al. (54), in addition to some references of the simulation results obtained by Cloete et al. (11). Even though the primary model includes release of both oil and gas into high pressure water, gas dissolution and the effect of ambient ocean currents, the basic model set-up is expected to be much of the same. Both models employs a coupled DPM and VOF model, the turbulence of water is calculated by the Standard k-ε model, in addition to the same bubble size model and bubble drag force. Section 7.3 and 7.4 contains a comprehensive discussion including conclusion of why the validation model can verify the slightly different primary model set-up, described in chapter 8.

As this thesis mainly is an extension of the work done by Cloete et al. (11) the simulation results presented in this chapter are expected to be quite similar to theirs.

7.1 Model Description

The Rotvoll model is carried out using transient, three dimensional flow computations. The Standard k-ε model is activated to account for turbulence in the surrounding water. In order to capture the surface interactions, the VOF model with its surface tracking technique is employed (ref. section 3.6.3:A). The Discrete Phase Model (DPM) is used to track the rising bubbles through the surrounding water, in the Lagrangian reference frame.

The residence time of one single bubble in the ambient water are expected to be quite short, thus the dissolution of air is neglected. The density distribution of the dispersed bubbles is following the ideal gas law (ref. section 4.2.1), and the bubble size distribution is given from section 4.2.4. The drag force exercised from the surrounding water onto the buoyant bubbles is given in section 4.1.1.

7.1.1 Geometry

*The geometry created in ANSYS Workbench has identical dimensions as the experimental basin, with a height of 7 m and surface area of 6 * 9 m. An air layer is placed on top of the basin to account for fountain effects and backflow effects, created by the pressure-outlet boundary (ref. section 7.1.3).*

7.1.2 Grid

The geometry is divided into a uniform grid size distribution of 40 cm, before it is exported into the ANSYS FLUENT solver. As the gradients of interest are found in the plume region, the grid size is refined three times using region refinement, explained in section 5.1.2:III. A gird size of 5 cm, in the plume region, is the same as Cloete et al. (11) applied. A mesh independency study (ref. section 5.1.3) was carried out by Pan (47), were additional finer grid did not offer improvements in the solution. The simulations are performed on a mesh with 832345 hexahedral cells. Figure 28 shows the grid of the z-center plane and bottom plane.

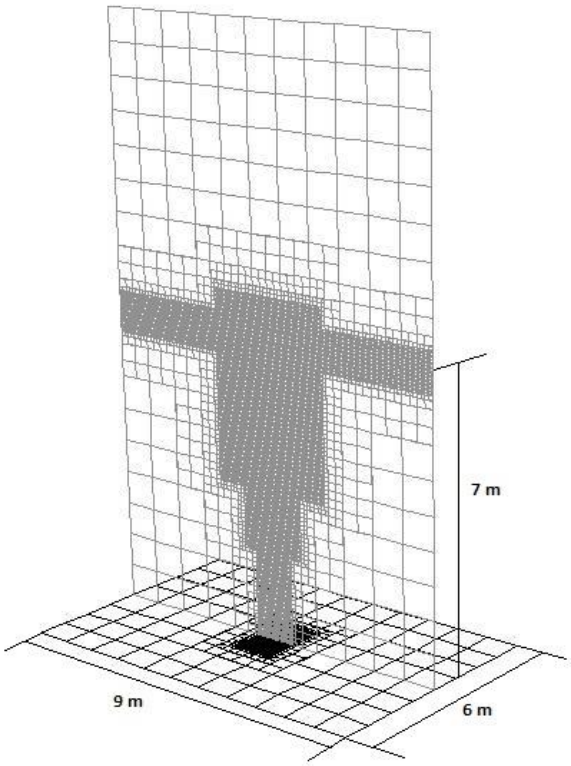

Figure 28: Numerical grid size distribution Rotvoll model.

7.1.3 Physical Properties and Boundary Conditions

Continuous Phase

The density and viscosity of the water phase is specified as 998.2 kg/m^3 and 0.001003 kg/m-s, respectively. A uniform temperature distribution of 15 ℃ is assumed for both the air and water phase. The atmospheric air is given by a density of 1.225 kg/m^3 and a viscosity of 1.7894e-05 kg/m-s. These values are default in ANSYS FLUENT.

The boundaries surrounding the continuous phases, except the pressure-outlet at the top boundary, are specified as walls, were the DPM particles are reflected in case of any contact. Additionally, the shear conditions of the walls are assumed no slip. A maximum of 20 iterations per time step is applied to provide sufficient flow variables at the grid faces.

Discrete Phase

*The density of the air-bubbles is specified to follow the ideal gas law (ref. Equation 44), with an individual gas constant (R_g) of 286.9 J/Kg*K and a water temperature (T_l) of 15 ℃. The pressure (P_l) is given by the hydrostatic pressure in the basin (ref. Equation 52). The bubbles are released at flow rates of 83, 170 and 750 Nl/s, from 100 equally spaced point sources*

located in a radius of 0.17 m. The number of injected parcels must be sufficiently large to account for the turbulent dispersion of bubbles (ref. section 3.6.2:II). The bubbles are tracked every 0.01 second, which corresponds to the calculation time step size. The bubble size model (ref. section 4.2.4), accounts for breakup and coalescence of the air-bubbles. The bubble shape and drag coefficient (C_D) is given by the expression of Xia et al. (10), in section 4.1.1.

Density variation, bubble size distribution, and slip velocity is given by the user-defined function "Validation_Model.c" (ref. appendix A.1), which is executed at the end of each time step. The drag force is found in the user-defined function "Bubble_force.c" (ref. appendix A.2). This UDF is hooked to the DPM in ANSYS FLUENT. The position of the bubbles leaving the water phase is written to a file, before the discrete particles are deleted due to computational efficiency and the fact that the current objective only considering the interactions between the bubbles and the surrounding water. This is accomplished by accessing the particle stream index through a user-defined macro (ref. appendix A.1 and B.1).

7.1.4 Initial Conditions

The domain is initialized with zero-values for all flow variables, except for the turbulent kinetic energy, k, and its dissipation rate, ε. Initial values of k and ε represents pressure and temperature differences that may create water movements before any bubbles are injected. The initial values of k and ε is not known from the work of Engebretsen et al. (54). However, according to Pan (47), the simulations was found to provide sufficient results with initial values of k = 0.007 m^2/s^2 and ε = 0.001 m^2/s^3. The effect of the turbulent initial values is discussed in section 7.3.3:A.

Due to installation of the momentum breaker, right above the release point, the initial velocity is set to zero. The effect of this momentum breaker is further discussed in section 7.3.4. The atmospheric air is defined as primary phase and water is patched into the lower region of the geometry (ref. figure 28).

7.1.5 Solution Method

The continuity, momentum and turbulence are derived from the Second-Order Upwind Scheme, which is based on higher order schemes (ref. section 6.3). In upwind schemes the face values are calculated from quantities in the cell upstream (28). Further, the discretization of the pressure is captured by the PRESTO! scheme (5), and the water surface is

captured by the Geo-Reconstruct Scheme (ref. 3.6.3:A) As discussed in section 3.4 and 5.3, the PISO scheme is used for the pressure-velocity coupling.

For more information about these schemes, see the ANSYS FLUENT Theory Guide (28).

7.1.6 Hardware and Software

The calculations are achieved by submitting the ANSYS FLUENT case and data file onto a high performance cluster, named Kongull. The Kongull cluster is a CentOS 5.3 Linux cluster running Rocks on HP servers with AMD processors. For more information see https://www.hpc.ntnu.no/display/hpc/Kongull.

7.2 Results

This section presents the current simulation results in comparison with experimental data presented in the work of Engebretsen et al. (54), in addition to some references to the simulation results presented in Cloete et al. (11). The calculations are stopped after 20 seconds, when quasi steady state is assumed (11). Fountain height, rise time and various velocity profiles are measured, compared, and presented in the following sections.

7.2.1 The Free Surface

As noted in section 2.2, the rising water is deflected outwards in a radial surface flow, and an elevation of the water surface occurs, due to momentum of the entrained water plume. In context of the current CFD simulation, this elevation is a visible evidence of the two-way coupling employed (ref. section 3.6).

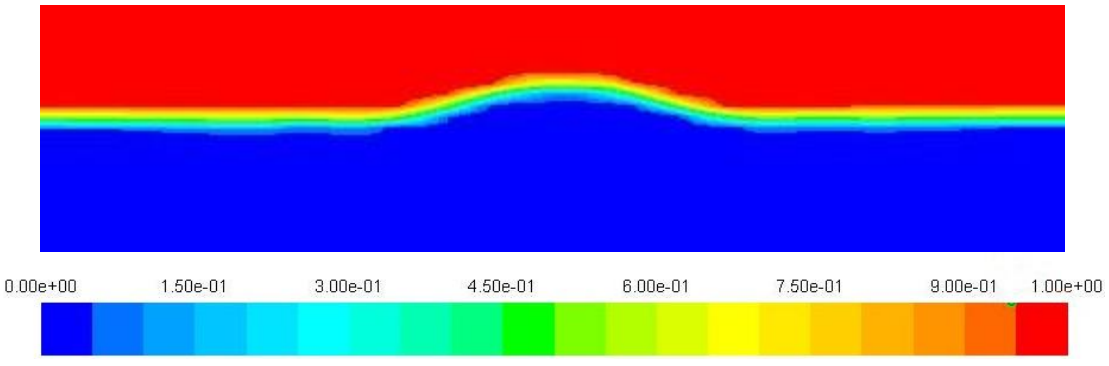

Figure 29: Contour plot colored by the volume fractions at the free water surface.

The contoure plot of volume fractions, showed in figure 29, illustrates the elevation of the water surface. The blue color indicates the water phase and red color indicates the atmospheric air. The yellow line represents the interface between the two phases, which is captured by the Geo Reconstruction Scheme (ref. section 3.6.3:A).

Figure 30 is a vector plot of water velocities in the interaction zone (ref. chapter 2), after quasi steady state conditions are reached. The water is radially deflected and the atmospheric air moves in the same direction, due to shear forces between water and atmospheric air.

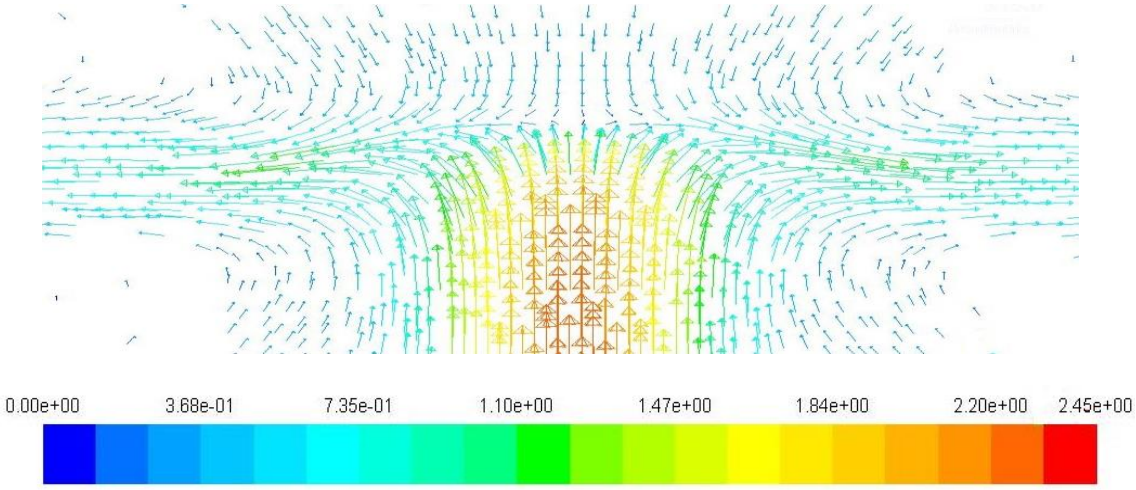

Figure 30: Velocity vector plot in the free surface region.

The founatin height was found by plotting the volume fractions against the position above the initial water surface (i.e. 7 m above the basin bottom), along the plume center line. A volume fraction of 1 indicates the water phase, while 0 indicates atmospheric air, in figure 31.

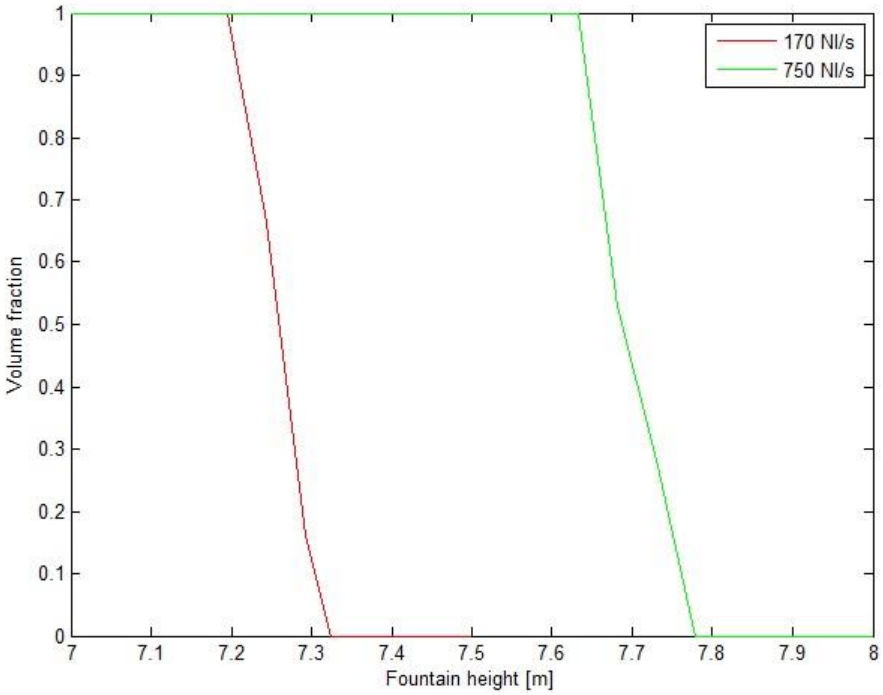

Figure 31: Plot of the fountain heights for 170 Nl/s and 750 Nl/s.

	Fountain height (cm)	
Flow rate:	170 Nl/s	750Nl/s
Experiment	30	45
Simulation	32.4	77.8

Table 1: Fountain height - Experiment vs. Simulation

As seen from table 1, the simulated fountain height yields quite good agreement with experimental data for the middle flow rate (i.e. 170 Nl/s), only slightly over-predicted. However, the highest flow rate shows a major discrepancy of 32.8 cm. Cloete et al. (11) presents, on the other hand, a slightly under predicted fountain height for the middle flow rate, while an even greater discrepancy, than showed in table 1, is observed for the highest flow rate. This is further discussed in section 7.3.

7.2.2 Rise time

The rise time is defined as the time to initial surface burst, i.e. the time it takes for the first bubbles to reach the water surface. As noted in section 7.1.3, bubbles are deleted when they reaches the interphase of water and atmospheric air. When a bubble is taken out of the system, the particle position is written to a file toghether with its residence time. For further details see appendix A.1. The residence time of the first bubble leaving the system defines the rise time of the discrete bubble plume.

	Rise time (s)		
Flow rate:	83 Nl/s	170 Nl/s	750Nl/s
Experiment	6.0	4.8	3.1
Simulation	6.14	5.15	3.47

Table 2: Rise Time - Experiment vs. Simulation

As seen from table 2, the rise time is over-predicted for all three flow rates. The higher flow rate the greater discrepancy is observed. Cloete et al. (11) presents a slightly lower simulated rise time compared with the current simulation results, presented in table 2. Possible factors of this over-predicted rise time is discussed in section 7.3.

7.2.3 Various Plume Velocities

As noted in chapter 2, the drag force exercised from the ambient water will quickly reduce the rise velocity of a bubble plume. A net momentum force is transferred from the bubbles to the ambient water, and a motion of water is developed alongside the bubbles. This entrainment effect makes the surrounding water to move in the same direction as the dispersed bubble plume, and a velocity profile of the entrained water is developing. For three different elevations the velocity profile generated by the bubble plume is experimental measured, for the middle flow rate (170 Nl/s) (11).

Figure 32 shows the experimental data compared with the current simulation results. For the two lower heights (i.e. 1.75m and 3.80m) the simulation results matches the experimental data quite well. However, a slightly under prediction of the simulated velocities are observed right outside of the plume center. This effect becomes significantly clearer at 5.88 m height. It is, however, reported that the vertical and horizontal velocities was measured with Höntzsch turbine flow meters, during the experiments (47). These types of turbine flow meters are suited to measure mono-directional flow, but may overestimate the vertical velocity component when the flows start to bend, when approaching the surface.

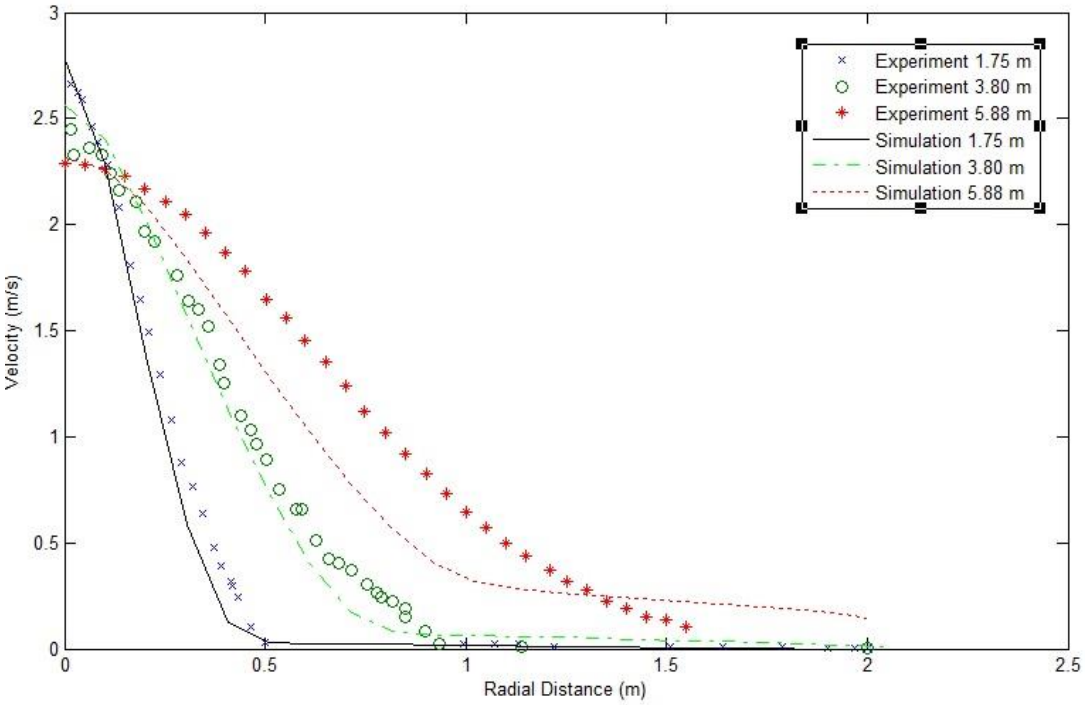

Figure 32: Velocity magnitude of water velocities at three different elevations at a gas flow rate of 170 Nl/s.

The center velocities (i.e. radial distance = 0) is observed to be slightly overestimated for the two lower heights, while the center velocity at 5.88 m height matches the experimental data very well. A comprehensive discussion of the results presented in figure 21 is presented in section 7.3. Simulation results of Cloete et al. (11) shows, however, a slightly under-predicted center velocity, while the velocities right outside the plume center matches very well in the work of Cloete et al. (11).

7.2.4 Velocity Magnitudes near the Top Surface

As the plume of entrained water approaches the surface, water is deflected outwards in a radial surface flow (figure 1 and figure 30). When this turning process occurs, the water velocity at the horizontal surface will increase until it eventually obtain steady state conditions. Engebretsen et al. (54) presents experimental data showing the velocity magnitude of water near the top surface for gas flow rates of 83 and 170 Nl/s, 1.75 m from the plume center. Below, in figure 33, the experimental surface flow data is compared with the current simulation results, which indicates a major discrepancy for the middle flow rate (170 Nl/s). The experimental data is shown to be predominantly in radial direction compared with the simulation result, which is more directed vertically. However, the plots match quite well for heights below 6.6 m, which may indicate that the SKE model does not account for turbulence damping in the vicinity of a free surface (ref. 6.2.1:B). This is further discussed in section 7.3.3:B.

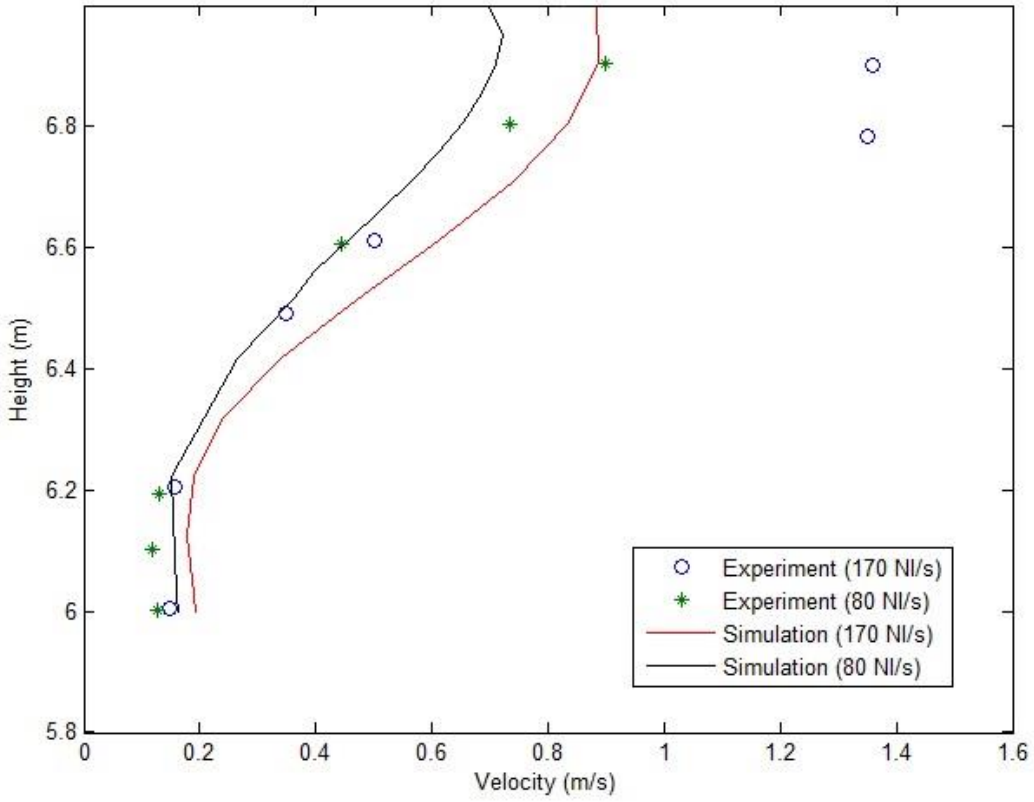

Figure 33: Comparison between experimental and simulated velocity magnitude of water near the top surface for the gas flow rates of 83 and 170 Nl/s, 1.75 m from the plume center.

7.3 Discussion

This section contains a comprehensive discussion of the various factors that may affect the results presented in the previous section, which indicated quite good accordance with experimental data. In CFD, and especially a system of such complexity as the one studied herein, there are a large number of simplifications and approximations being made through turbulence modeling, choice of algorithms, and small differences in initial conditions, that can have large and unpredictable impact on the results. When presenting CFD data it is therefore exceedingly important to be critical, thorough and specific.

7.3.1 Void Fraction

As noted in section 3.6.2:A and section 6.1, the DPM model contains one important assumption. As the discrete phase particles does not occupy any volume, the dispersed phase should be sufficient dilute so that particle-particle interactions and the effects of the particle volume on the continuous phase are negligible. ANSYS FLUENT Theory Guide (28) claims that this in practice implies that the discrete phase must be present at a fairly low volume fraction, usually less than 10-12%, based on each control volume. Violation of this volume fraction limit may provide unrealistic simulation results. In figure 34, the void fraction of gas is plotted along the plume center line after quasi steady state conditions is reached.

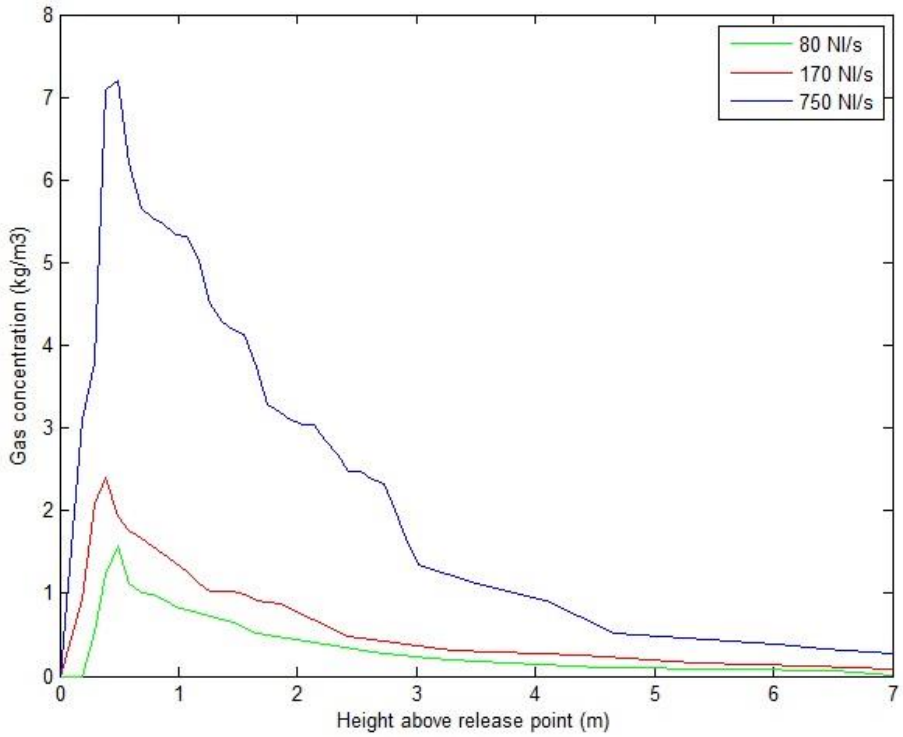

Figure 34: Void fraction of gas plotted along the plume center line after quasi steady state conditions are reached (20 sec).

As seen from figure 34, the gas concentration limit of 10-12% is strongly violated. A gas concentration above 1 is physical impossible, and may have major influence on the simulation results. Plume regions dominated by dense DPM concentrations (i.e. void fraction) are reported to overestimate the momentum transfer from the dispersed bubbles (5). Thus, this effect may increase the center velocity of the entrained water plume, in addition to overestimate the fountain height. Table 1 and figure 32 support this effect of over-predicted momentum transfer. Table 1 shows a fountain height that increases its discrepancy to experimental data with flow rate, where higher flow rate indicates stronger violation of the discrete phase model (ref. figure 34). Moreover, the simulated center velocity is slightly higher at the two lower heights (ref. figure 32), which may imply overestimated momentum transfer. However, the rise time (ref. section 7.2.2) is not expected to be influenced by the violation of the discrete phase model, as the first bubbles will experience control volumes dominated by water. The discrepancy in rise time (ref. table 2) is discussed in the next sections.

As noted in Chapter 3.6.2:II, the horizontal extension of bubble plumes is affected by the surrounding gas concentration. As the gas concentration exceeds the physical limit of 1, it is likely to believe that the dispersion of bubbles should be greater in order to decrease the gas concentration in the plume center region. Higher degree of bubble dispersion may decreased the center velocity plotted in figure 32 and increase the velocities in radial direction.

Pan (47) implemented a bubble-wake induced turbulence model, discussed in section 6.1.2. The model was found to provide a slightly more dispersed bubble plume with lower plume center velocity. Thus, by implementation of the bubble-wake model the void fractions (ref. figure 34) might be reduced somewhat.

7.3.2 Bubble Size Model

The bubble size model (ref. section 4.2.4), accounting for breakup and coalescence mechanisms, is the same as the one applied by Cloete et al. (11) and Pan (47).

As noted in section 6.1.3, it is reported that the first bubble observed reaching the surface, during the Rotvoll experiment, was one single large diameter bubble. One large diameter bubble rising towards the water surface will experience higher drag force from the ambient water, but even stronger buoyancy force. Since the bubble size model does not account for bubbles of large diameter, this may be a possible source of error when considering the simulated rise time (ref. table 2).

In figure 35, a parcel plot of the bubble size distribution is plotted after quasi steady state is reached, which is quite consistent with the theory from chapter 2 and the simulation results of Pan (2014). The larger bubbles exist in the outer boundary of the plume, where coalescence dominates due to low level of turbulence. As the bubble size model accounts for breakup in regions were the level of turbulence is high, the plume region is dominated by smaller bubbles.

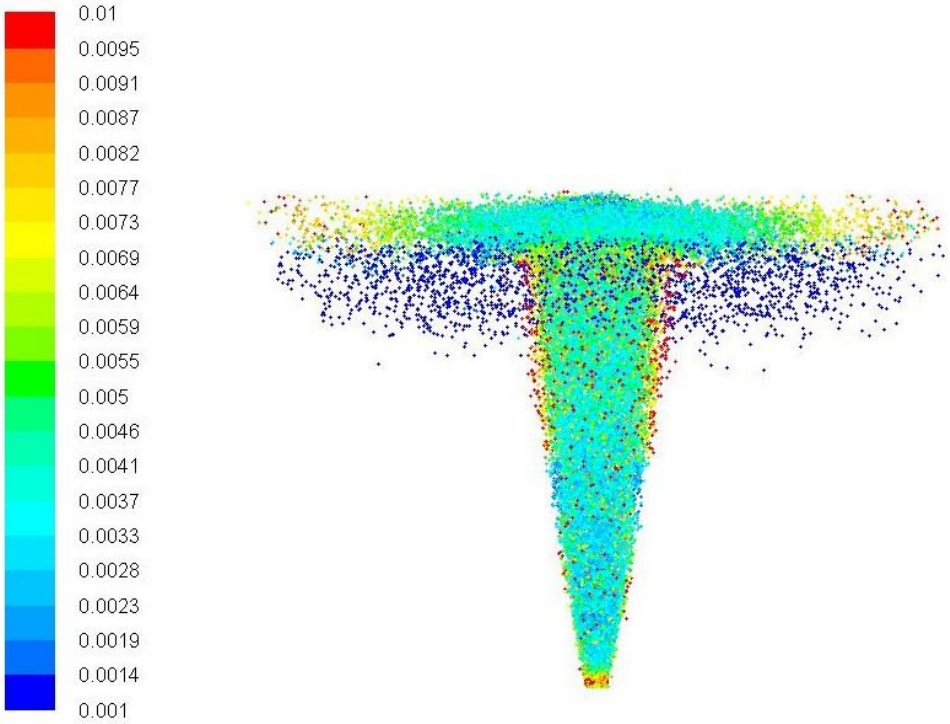

Figure 35: Parcel plot of the bubble diameter after quasi steady state conditions are reached (20 sec).

7.3.3 Turbulence

In the current application, the k-ε model is used to account for turbulence in the surrounding water. The k-ε turbulence model in combination with the Random Walk Model assumes that the turbulent viscosity is isotropic, which means it is treated as a scalar quantity with similar effects in all directions. However, this is probably not the complete truth. One of the primary findings of Sheng and Irons (13) was that turbulence is not isotropic for gas-liquid plumes. Their experiments showed that the turbulence is greater in the vertical direction. Moreover, Johansen et al. (68) found in all their experiments a higher radial turbulence relative to axial turbulence near the free surface. These results may favor an anisotropic turbulence model, which possible contributes to less dispersion of the bubble plume and thus a lower rise time.

A. Initial Values of k and ε

The initial values of k and ε seems to have major effects on the bubble plume properties. These values are tuned after experimental results in the work of Cloete et al. (11) and Pan (47), which indicates they are not experimental measured data. These values refer to the initial turbulence level in the surrounding water, and the ratio between them may influence simulation results. When the turbulent kinetic energy (k) is increased, relative to ε, the horizontal extension of the bubble plume may increase, which might contribute to a longer rise time. In the table 3, the rise time for different initial values of k is showed. The initial dissipation rate is assumed constant (ε = 0.001 m^2/s^3) for all three cases.

Type: Flow Rate 170 Nl/s	Initial Turbulent Kinetic Energy (m2/s2):	Rise time (s):
Current Simulation 1	0.001	3.98
Current Simulation 2	0.007	5.15
Current Simulation 3	0.014	5.88

Table 3: Rise time for various initial values of the turbulent kinetic energy (170 Nl/s).

Table 3 indicates that higher initial values of k provides a longer rise time. Possible causes of this discrepancy in rise times are discussed below.

I. Eddy Time Scale and Turbulent Dispersion

The eddy time scale, or "eddy lifetime", is a measure for the time spent by a parcel inside a turbulent eddy. Equation 38, presented below, is the fluid Lagrangian integral time (ref. section 3.6.2:II):

$$T_L = 0.15\frac{k}{\varepsilon}$$

As seen from the equation above, a higher value of the turbulent kinetic energy k, will increase the eddy time scale. This may increase the turbulent dispersion of the bubble plume, and decrease the plume rise velocity. One way to think of it is that a parcel are spending longer time in one specific turbulent eddy, moving the motion of bubbles away from the plume center. In case of smaller eddy time scales, the parcels may be affected by multiple eddies working in different directions. This mindset is illustrated in figure 36, where one single bubble is tracked for high and low eddy time scale.

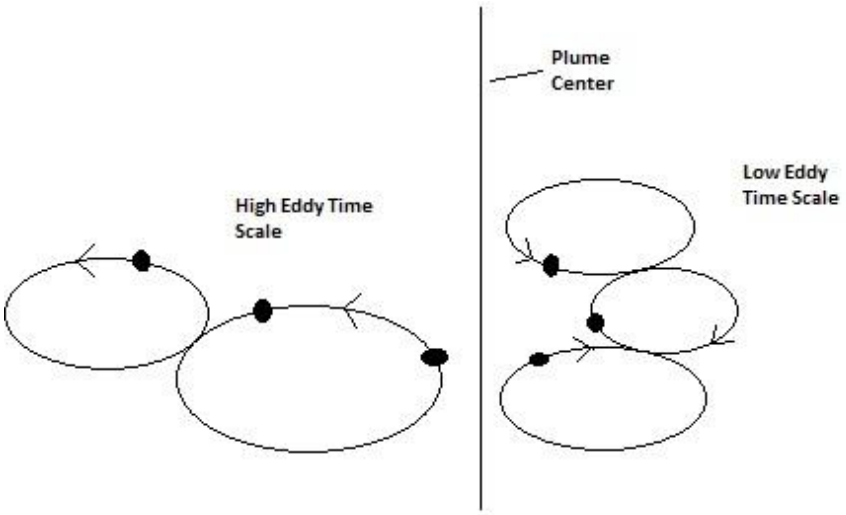

Figure 36: Motion of a particle for high eddy time scale vs. low eddy time scale.

II. Velocity Fluctuations

By the assumption of isotropy, the root mean square (RMS) values are equal for all three velocity fluctuations, and may be deduced from the turbulent kinetic energy (k), as noted in section 3.6.2:II.

$$\sqrt{\overline{u'^2}} = \sqrt{\overline{v'^2}} = \sqrt{\overline{w'^2}} = \sqrt{\frac{2k}{3}}$$

An increase in k may increase the horizontal extension of the bubble plume, due to increased velocity fluctuations in all directions. Velocity fluctuations are presented in figure 8.

III. Turbulent Viscosity

The isotropic turbulent viscosity, μ_t, is a function of both k and ε, as discussed in section 3.5.4:

$$\mu_t = \rho C_\mu \frac{k^2}{\varepsilon}$$

Viscosity in turbulent flows is defined as the internal fluid resistance. Turbulent eddies are transferring momentum, in which creates internal fluid friction. Increased turbulent viscosity may thus force turbulent eddies to drag more on the surrounding water, which may increase the dispersion of the entrained water plume and decrease the plume center velocity of the

water. Subsequently, this increased viscosity may decrease the rise velocity of a bubble plume.

B. Free Surface Damping

As discussed throughout this thesis, turbulence modeling is critical for accurate prediction of the mean and turbulent velocity field, bubble dispersion, bubble sizes, and bubble-fluid momentum transfer. In figure 37, the experimental surface flow data is plotted against the current simulation results, which indicates a major discrepancy near the surface. The main reason of this discrepancy is that the Standard k-ε turbulence model does not account for turbulence damping in the vicinity of a free surface (ref. section 6.2.1:B). This effect is illustrated in Figure 37. Soga and Rehmann (79) reported that there is an increase in turbulent dissipation at the free surface in bubble plumes. However, this effect is not captured by the Standard k-ε turbulence model. In figure 37, the effect of free surface damping is plotted against the experimental data from Engebretsen et al. (54) and the current simulation results from section 7.2.4.

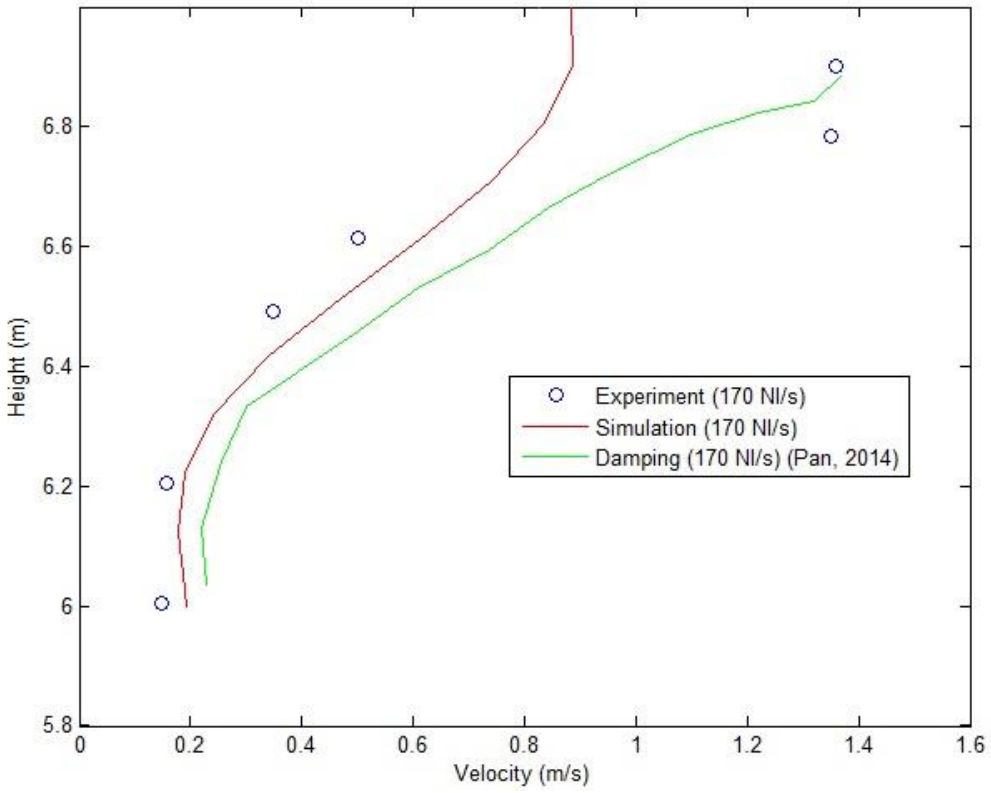

Figure 37: Comparison between experiment, simulation without and without surface damping effects for flow rate of 170 Nl/s.

Seen from figure 37, including free surface damping provides a good match with experimental data near the free surface. The effect of turbulent surface damping is presented in the work of Pan (47). Here the Standard k-ε turbulence model is extended to account for

the proximity of a free surface, by supplying the correct characteristic length to treat the near surface turbulence (ref. section 6.2.1:B). For more information about the free surface damping, see Pan (47).

The effect of the free surface damping might get more clear by examine the expression of turbulent viscosity, μ_t, once more (ref. Equation 28). As the turbulent kinetic energy, k, is reduced and rate of turbulent dissipation (ε) is increased, the turbulent viscosity is decreased. A less viscos fluid will flow more easily, due to lower internal friction.

7.3.4 Momentum Breaker

As noted in section 7.1.4, the inlet velocity of the discrete bubbles is assumed to be zero. This is to account for the arrangement installed in front of the release point. However, this assumption is likely to affect the simulation results. When a jet of gas bubbles strikes a horizontal plate, the bubbles are dispersed in the radial direction and may increase the horizontal extension of the bubble plume. Thus, bubbles may start their rise towards the water surface before they passes the momentum breaker on the outside. It is, however, hard to determine the exact impact of this assumption.

7.4 Concluding Remarks

Clearly, the most concerning assumption of the validation model is the violation of the Discrete Phase Model (DPM) (ref. section 7.3.1). The author finds it unlikely that such high and unphysical void fractions can be neglected, and that the possible error of this violation might be "covered" by appropriate tuned initial values of the turbulent kinetic energy, k, and its dissipation rate, ε. However, as the DPM concentration is calculated from the bubble concentration of each control volume, the DPM concentration is expected to be significant lower for the primary model, due to:

- *Significant larger control volumes (i.e. numerical grid cells).*
- *Possible violation of the DPM concentration limit will most likely influence only small regions of the total plume volume, which implies it can be neglected.*
- *As oil droplets and gas bubbles are affected by ocean currents and significantly longer residence time, the horizontal extension of the discrete phase is expected to decrease the gas concentration in the control volumes of interest.*

Moreover, the VOF model seems to capture the interaction between the continuous phases and the discrete parcels very well (ref. section 7.2.1). However, no gas bubbles are expected to surface during the DeepSpill simulation, due the effect of gas dissolution. Thus, only a surface slick of oil droplets is captured by the Geo-Reconstruct Scheme. As the turbulence of oil droplets is expected to be much less pronounced near the free surface, the effect of free surface damping is neglected in the primary model.

In section 7.2.3, the water velocities in the lower regions match the experimental data quite well. Thus, it is likely to believe that the Standard k-ε model predicts the degree of water turbulence for underwater blowouts in a sufficient way, under the assumption of appropriate initial values of k and ε. Moreover, the hydrostatic pressure is, in both models, accounted for by Equation 52.

It seems like the general set-up of the validation model fits the primary model quite well. A significantly larger geometry, inclusion of oil droplets and the presence of gas dissolution and ocean currents is the most distinctive differences between the two models.

Chapter 8

Primary Model: DeepSpill Experiment

The DeepSpill experiment included four controlled discharges of oil and gas from a ocean depth of 844 m, conducted in the Norwegian Sea. The main objectives were to calibrate numerical models for modeling blowouts in deep waters. In addition, testing surveillance and monitoring equipment, together with evaluation of the safety aspects of accidental releases of gas and oil in deep waters was of high interest (34). Extensive observations and documentation were acquired during the experiments by use of wind and current meters, aircraft surveillance, sampling of oil from the surface slicks, mapping of subsurface plumes with remotely operated vehicles (ROV) and echo sounder, as well as by chemical and biological sampling in the water column. Figure 38 illustrates this massive operation.

.

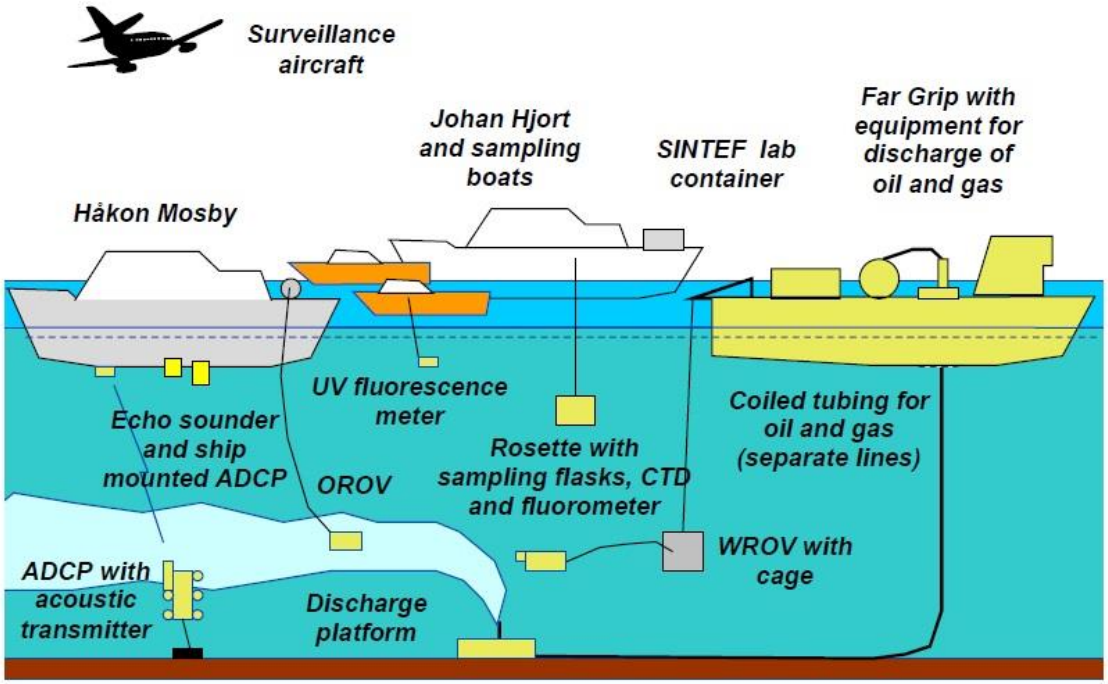

Figure 38: Schematic overview of participating units at the DeepSpill experiment (34).

The four controlled discharges consisted of:

- *Nitrogen gas and dyed sea water.*
- *Marine diesel (oil) and LNG.*
- *Crude oil and LNG.*
- *LNG.*

In the current application, two of the discharges is evaluated; marine diesel and LNG, and LNG. Throughout this chapter, marine diesel is often referred to as oil, while LNG is referred to as gas or methane (ref. section 4.4).

As a part of the analysis of the experimental observations and measurements, the DeepBlow model (ref. section 2.4) developed by SINTEF was compared with field data. This model is a highly complex integral plume model based on a Lagrangian concept, and was designed with special emphasis on deep water conditions. In this chapter, simulation results are presented, discussed and compared with chosen experimental data obtained from the report of Johansen et al. (34) and relevant modeling results provided by the DeepBlow model. As noted in Chapter 1, the main objective is to measure and evaluate the following:

- *The mean cloud path of oil*
- *Point of complete gas dissolution*
- *Rise time of oil*

8.1 Model Description

The primary model is carried out using transient, three dimensional flow computations, as the case was for the validation model. The Standard k-ε model is activated to account for turbulence in the surrounding ocean. In order to capture the surface interactions between ocean surface, atmospheric air and discrete phase, the VOF model with its interface tracking technique is employed (ref. section 3.6.3:A). The Discrete Phase Model (DPM) is used to track the rising gas bubbles and oil droplets through the ambient ocean.

The residence time of one single bubble is expected to be long enough to make gas dissolution effects dominant (ref. section 4.4). To account for ocean currents, an acceleration term is exerted on each bubble/droplet at the end of each time step, as discussed in section 4.5.3:Method 3. The density distribution of the dispersed bubbles is assumed to following the ideal gas law (ref. section 4.2.1) and the bubble size distribution is given by section 4.2.4. The oil droplets are assumed incompressible (i.e. no change in density) and the droplet size is found to be approximated by a Rosin-Rammler distribution (ref. section 4.2.3). The bubble shape is given by the Eotvos number and the drag force exercised from the ambient ocean on

the rising bubbles is given in section 4.1.1. The oil droplets are assumed to be spherical, and their drag coefficient is given by Equation 50.

8.1.1 Geometry

*The geometry is created in ANSYS Workbench, with a height of 1200 m and a surface area of 1000 * 1000 m. As the depth of the ocean is 844 m, the atmospheric air layer on top measures 356 m in height.*

8.1.2 Grid

*The same gridding procedure is conducted for the primary model as the validation model in section 7.1.2. A uniform grid size of 83.33 m is first created in ANSYS Workbench, before a refinement is performed in the domain for where the discrete phase is expected to rise. This region is refined three additional times, making each control surface area approximately 10 m * 10 m. This is consistent with the unpublished SINTEF report which claims a number of 80 cells are sufficient along the vertical center line, for deep water blowouts using the coupled DPM and VOF approach (ref. section 5.1.3). As discussed in section 6.1.3, the water surface area should be covered by a uniform grid size distribution to avoid artificial velocity generations and numerical instabilities. The simulations are performed on a mesh with 253169 hexahedral cells. The z-center plane of the grid (in x- and y-direction) is presented in section 8.2.1. As seen from Figure 40, the grid is refined for much larger regions than necessary for that specific simulation. This is done so the same geometry and mesh may account for greater plume dispersion (ref. section 8.3.1). A reasonable presentation of the grid is difficult; however, an attempt is presented below.*

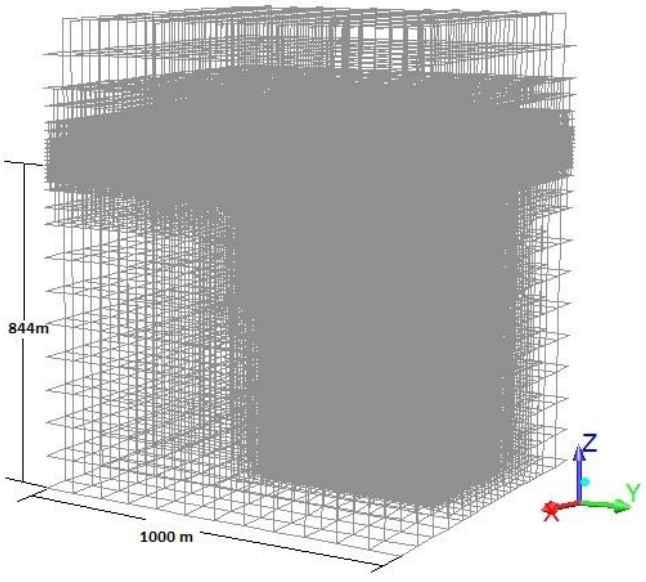

Figure 39: Geometry and mesh distribution - Primary model.

8.1.3 Physical Properties and Boundary Conditions

Continuous Phase

The density and viscosity of the sea water is specified as 1027 kg/m^3 and 0.001003 kg/m-s, respectively. A uniform temperature distribution of 5 ℃ is assumed for both the atmospheric air and sea water. The density of the air phase is 1.225 kg/m^3, while the viscosity is specified as 1.7894e-05 kg/m-s.

The boundaries surrounding the continuous phases are specified as walls, except the pressure-outlet at the top boundary. The DPM parcels are reflected in case of any contact with the surrounding walls. The shear conditions of the walls are assumed no slip, and a maximum of 20 iterations per time step is applied to provide sufficient flow variables at the grid faces.

Discrete Phase

*The oil is assumed incompressible, as noted in section 4.2.2. The density of methane is specified to follow the ideal gas law (ref. section 4.2.1), with an individual gas constant (R_g) of 518.3 J/Kg*K. The pressure (P_l) is given by the hydrostatic pressure along the vertical center line in the ocean domain (ref. Equation 52). The methane bubbles and oil droplets are injected from 10 equally spaced point sources located in a diameter of 0.12 m. The number of injected parcels is further discussed in section 9.3.1. Table 4 shows additional input data for the DPM.*

	Oil and Gas	Gas
Outlet diamter (m)	0.12	0.12
Gas flow rate (Sm^3/s)	0.6	0.7
Oil flow rate (m^3/hour)	60	-
Density of gas (kg/m^3)	0.67	0.67
Density of oil (kg/m^3)	854.8	-

Table 4: Input data DPM – Primary model (34).

The bubble shape and drag coefficient (C_D) is given by the expression of Xia el al. (10) (ref. section 4.1.1), while the oil droplets are assumed spherical (ref. section 4.1.2), which is by default in ANSYS FLUENT. Density variation, bubble size distribution, and gas dissolution is given by the user-defined function "Primary_Model.c" (ref. appendix B.1), which is executed at end of each time step. The bubble drag force is found in the user-defined function "Bubble_drag.c" (ref. appendix A.2). This UDF is hooked to the DPM in ANSYS FLUENT. The

oil size is approximated by a Rosin-Rammler distribution (ref. section 4.2.3).The input values are shown in table 5:

Rosin-Rammler Parameters for oil:	Values:
Min. Diamter (m):	0.002
Max Diameter (m):	0.007
Mean Diameter (m):	0.005
Spread Parameter (n):	2.5

Table 5: Input values – Rosin-Rammler Distribution (34).

The ocean currents are modelled as acceleration terms returned to each rising particle, where the implementation is found in the UDF named "Ocean_currents.c" (ref. appendix B.2). The position of the discrete particles is tracked as they rise through the continuous phase, which is implemented in the UDF "Primary_Model.c" (ref. appendix B.1). The gas bubbles are deleted when reaching the water surface, as explained in section 7.1.3.

8.1.4 Initial Conditions

The initial values of the turbulent kinetic energy (k) and its dissipation rate (ε) is highly uncertain for the current application. These values are not specified in Johansen et al. (34), and are therefore based on experimental data from other literature sources. So, in sea waters dominated by ocean currents, Dewey (58) found an initial turbulent intensity of about I = 5% of the mean turbulent kinetic energy. The initial turbulent intensity may be expressed as follows (14):

$$I = \frac{u'}{\bar{u}} \qquad \text{EQUATION 73}$$

where u' is the root-mean-square of the turbulent velocity fluctuations (ref. Equation 35 and 36), and $\bar{u}$ represents the initial mean velocity magnitude, which may be computed from the three mean velocity components:

$$\bar{u} = \sqrt{\bar{u}_x^2 + \bar{u}_y^2 + \bar{u}_z^2} \qquad \text{EQUATION 74}$$

The initial mean velocity is assumed to be represented by the ocean current velocities (in x- and y-direction) nearest the release point, given in Figure 19 and the UDF "Ocean_currents.c" (ref. appendix A.1). By the assumption of isotropy (ref. section 3.5.4:A) and combining Equation 35 and 36, the turbulent kinetic energy may be expressed as:

$$k = \frac{3}{2}u'^2$$ EQUATION 75

,where the initial turbulent kinetic energy is found to be $k \sim 6.09 \cdot 10^{-5} m^2/s^2$.

The rate of dissipation of turbulent kinetic energy, ε,is obtained from Thorpe (30). Here the range and observed variation of ε is presented at three different straits; the Florida Straits, the Strait of Gibraltar, and the Equator. An approximation of these values provides an assumed dissipation rate of $\varepsilon = 7.3 \cdot 10^{-6}\ m^2/s^3$. *The validity of these turbulent initial values is hard to determine and are highly unsafe. The ratio between them is more or less the same as the turbulent initial ratio used in the validation model (ref. section 7.1.4). However, the initial values of* k *and* ε *is expected to have minor effects on the overall simulation results as the flow time of the discrete phase are significantly longer compared with the validation model.*

Further, the release velocities of the discrete phases are calculated from the flow rates and the outlet diameter, presented in table 4. The rest of the domain is initialized with zero-values for all flow variables.

8.1.5 Solution Method

The continuity, momentum and turbulence are derived from the Second-Order Upwind Scheme, which is based on higher order schemes (ref. section 5.3). In upwind schemes the face values are calculated from quantities in the cell upstream (28). Further, the discretization of the pressure is captured by the PRESTO! scheme (5), and the volume fractions is captured by the Geo-Reconstruct Scheme (ref. 3.6.3:A) As discussed in section 3.4 and 5.3, the PISO scheme is used for the pressure-velocity coupling.

For more information about these schemes, see the ANSYS FLUENT Theory Guide (28).

8.1.6 Hardware and Software

The calculations are achieved by submitting the ANSYS FLUENT case and data file onto a high performance cluster, named Kongull. The Kongull cluster is a CentOS 5.3 Linux cluster running Rocks on HP servers with AMD processors. For more information see https://www.hpc.ntnu.no/display/hpc/Kongull.

8.2 Results

This section presents the primary simulation results in comparison with chosen experimental data from the DeepSpill experiment (34). In addition, relevant modeling results obtained from the DeepBlow model are to be presented. The mean cloud path of oil, point of complete gas dissolution and the rise time of oil is measured, presented and compared in the following sections.

8.2.1 The Mean Cloud Path of Oil

The second discharge, during the DeepSpill experiment, was release of marine diesel (oil) and natural gas. The ocean current profile is obtained from figure 19. Figure 40 shows a parcel plot colored by the discrete phase density after 4000 seconds of flow time. The red color indicates oil and the methane gas has taken the blue color. The rise time of oil is presented in section 8.2.3.

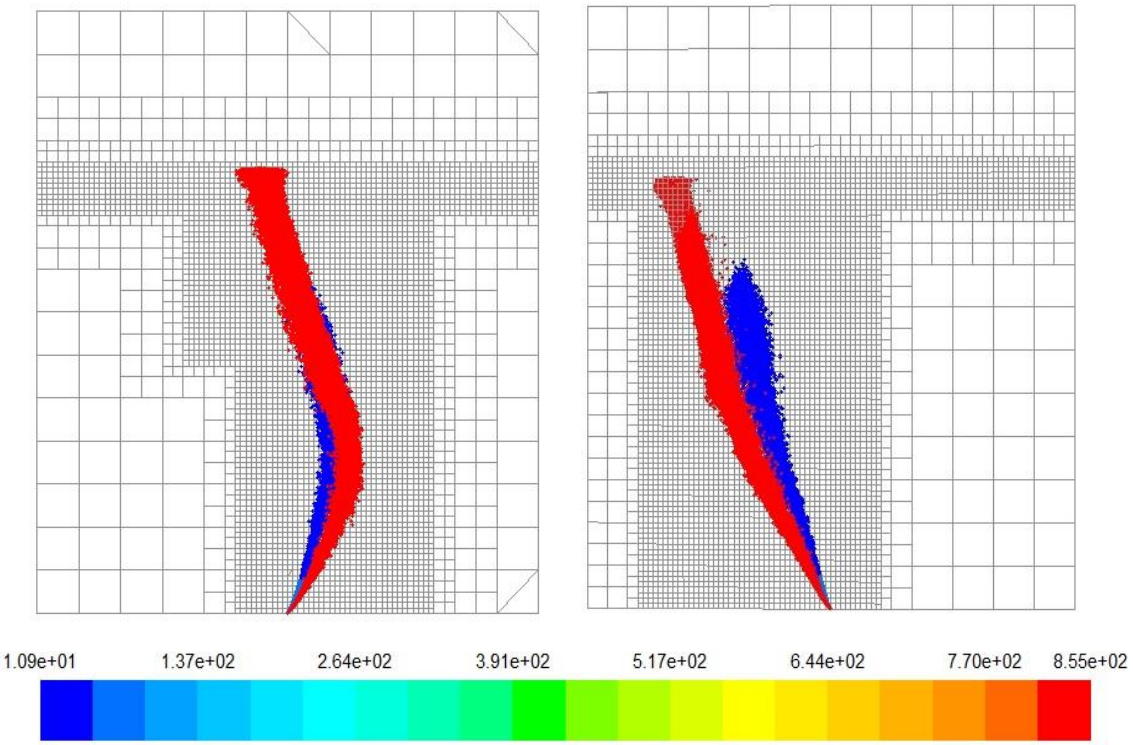

Figure 40: Oil and gas movement in east/west (l.s) and north/south (r.s).

Left hand side of figure 40 shows the bubble/droplet movement in east/west direction, while the right hand side of figure 40 illustrates the discrete phase movement in north/south

direction. The horizontal extinction of the gas and oil plume is something smaller than expected. This is further discussed in section 8.3. As can be seen from figure 40, the gas phase is trapped below the ocean surface due to dissolution effects, as explained in chapter 2. From this point, the oil droplets will start to rise slowly under buoyancy forces alone without any assistance from the more buoyant gas bubbles. The gas phase is further discussed in the next section. As indicated by figure 40, oil droplets have started to surface.

During the DeepSpill experiment, echo-sounder images showed a clear response on the underwater plumes generated. Higher degree of reflected sound waves indicates a more concentrated cloud. In the current simulation, the oil droplets are tracked while they rise towards the water surface. The position (x, y and z) of an oil droplet is written to a file for various elevations below the seabed (ref. appendix B.2). Figure 41 shows a comparison between the computed mean path of the oil droplet cloud and the data from the corresponding echo-sounder images.

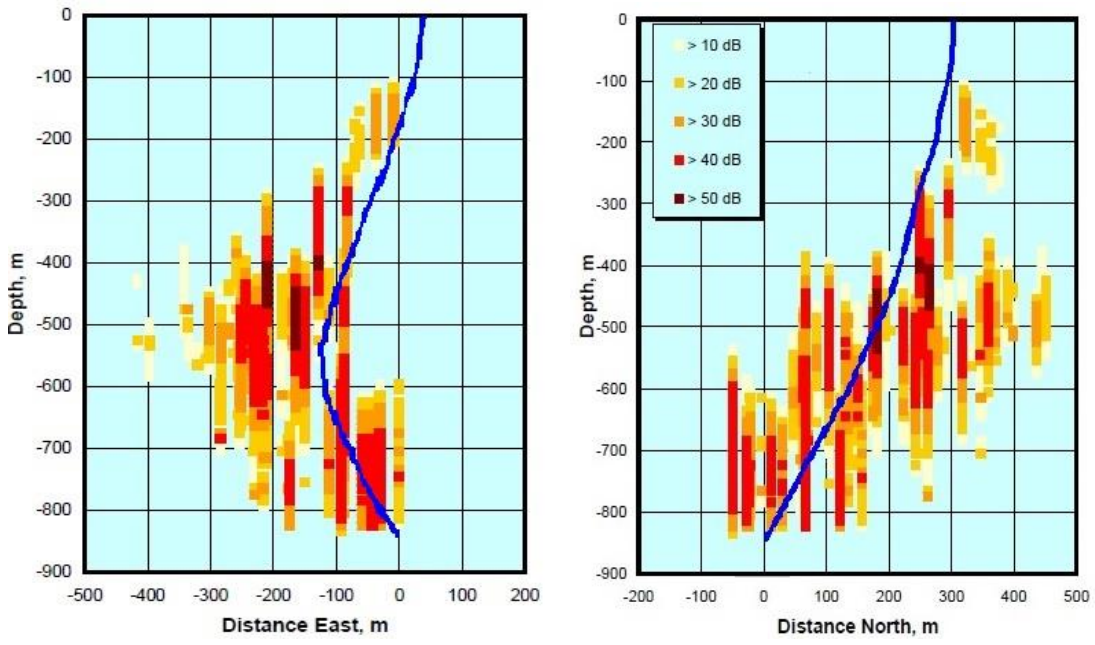

Figure 41: Mean path of oil (blue line) plotted together with echo-sound images taken during the DeepSpill experiment (34).

Figure 41 indicates that the computed mean path (blue line) corresponds fairly well with the overall shape of the cloud observed with the echo-sounder during the DeepSpill experiment. Moreover, the DeepBlow model did achieve quite similar results, as presented in figure 41.

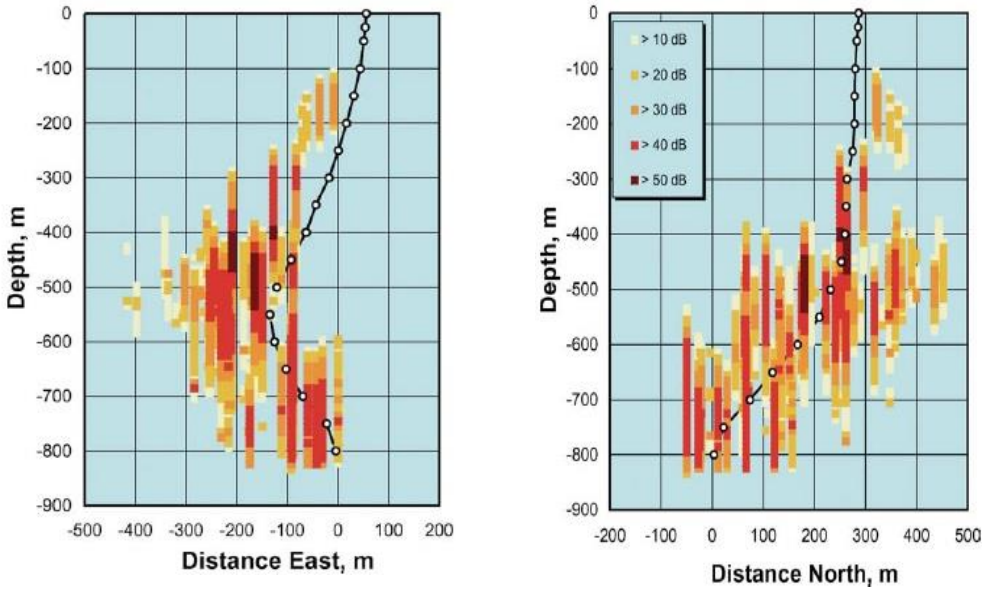

Figure 42: Mean cloud path of oil (DeepBlow model) plotted together with echo-sound images taken during the DeepSpill experiment (37).

8.2.2 Point of Complete Gas Dissolution

The fourth discharge during the DeepSpill experiment involved natural gas release, without any presence of oil droplets. The duration of this experiment was approximately 2 hours. The ocean current profile in east/west and north/south direction is presented in the report Johansen et al. (34). Figure 43 shows a parcel plot of methane after 2 hours (7200 sec). The parcels are colored by the parcel mass, in kg.

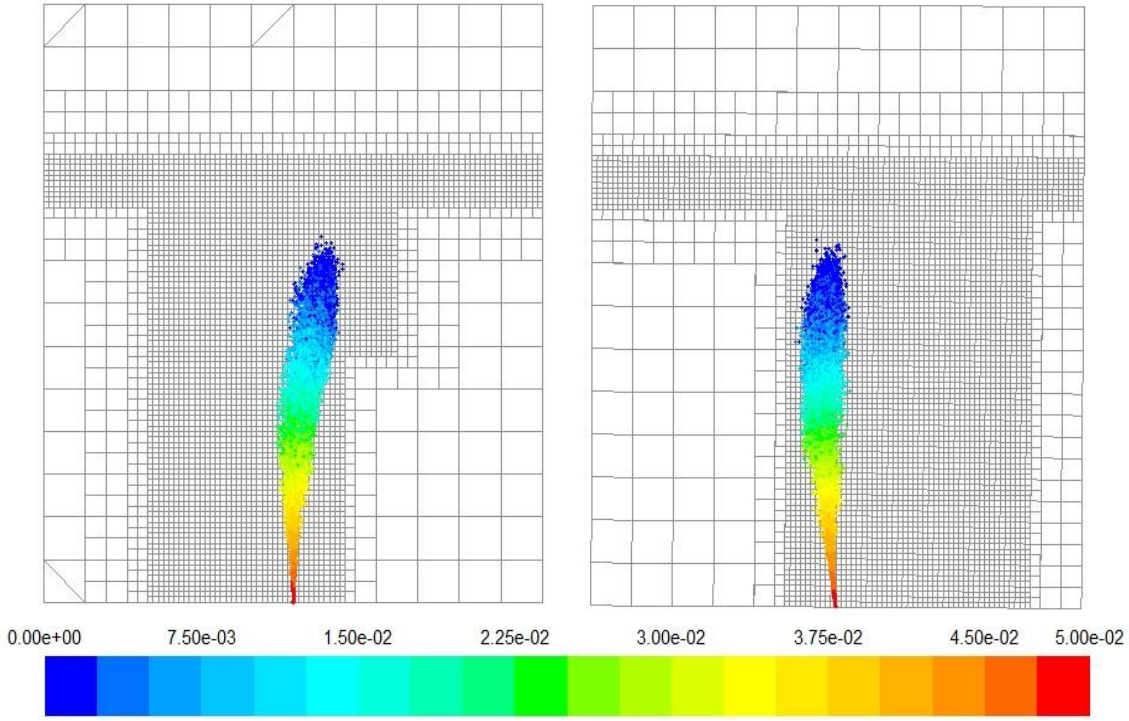

Figure 43: Bubble movement in east/west, north/south, and point of complete gas dissolution.

Left hand side of figure 43 shows bubble movements in east/west direction, and the right hand side of figure 43 illustrates the methane gas movements in north/south direction. Also here the dispersion of the bubble plume is something smaller than expected. Plume dispersion is, however, further discussed in section 8.3. Figure 40 indicates not only that the bubbles do not reach the surface, but also that the mass of each bubble tends to be reduced with distance from the release point. Such loss of gas is very likely to be due to dissolution of gas into the ambient sea water (34).

In the current simulation, the gas bubbles are tracked while they rise through the domain, as explained in the section above. Figure 44 shows a comparison of the corresponding echo-sounder data and the computed mean path of the bubble cloud, which is found to be centered well within the echo-sounder signal. Moreover, the gas is seen to disappearing at about 120 m depth, while echo-sounder images show a trapping depth of approximately 150 m. This indicates a very good match between the experimental data and the current simulation. As discussed in section 4.4, the solubility of methane gas in seawater is in principle based on the solubility coefficients provided by Lekvam & Bishnoi (16) and the mass transfer coefficient given by Zhang & Xu (74) (ref. section 4.4). However, by inserting these values into the mass transfer equation (ref. Equation 70), significantly higher rate of mass is transferred to the surrounding sea water. In order to match the echo-sound data showed in figure 44, a reduction factor of 0.38 (tuned by author) had to be introduced in the mass transfer equation. Simulation results without this reduction factor is presented and discussed in section 8.3.

The DeepBlow model observed the same problem of rapid gas dissolution, for which a reduction factor of 0.25 had to be employed to match the DeepSpill experiment.

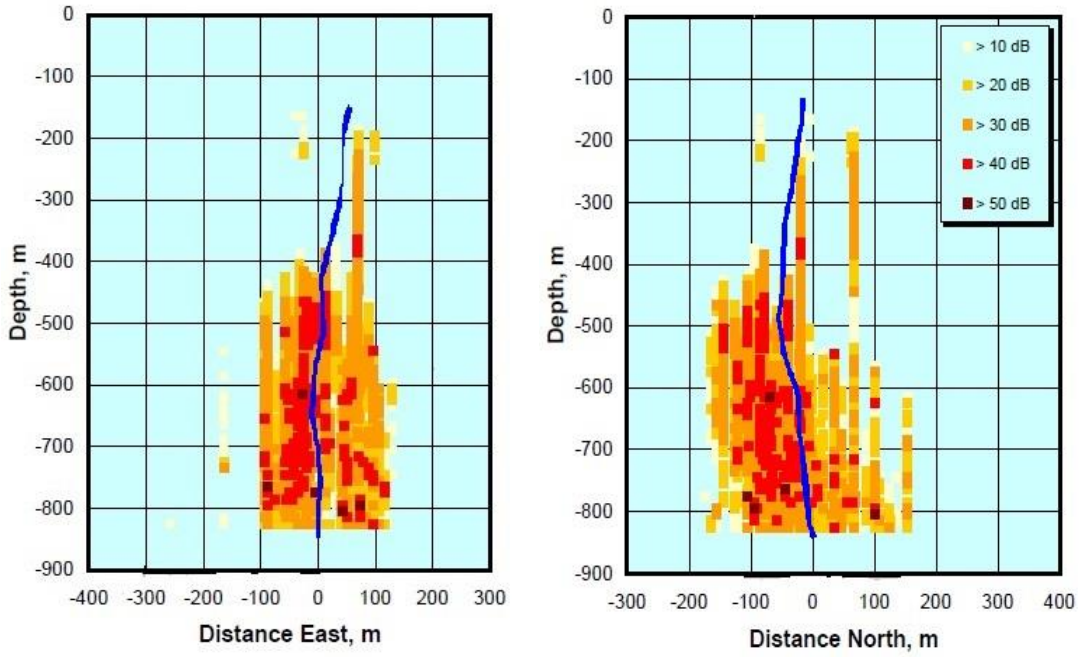

Figure 44: Mean cloud path of LNG and point of complete gas dissolution compared with echo-sound images taken during the DeepSpill experiment (34).

8.2.3 Rise Time of Oil

As explained in section 7.2.2, the rise time is the time it takes for the first droplets to reach the water surface. The rise time of oil droplets are expected to be influenced by the more buoyant gas bubbles and the surrounding ocean currents (ref. chapter 2). In case of significant ocean currents, the gas bubbles may separate quickly from the oil droplets, making the drops rise slowly alone under buoyancy forces. However, when ambient currents is too weak to completely separate the two dispersed phases, the oil droplets are entrained into the bubble flow and transported more quickly towards the water surface, until the gas bubbles eventually have lost all their buoyancy due to dissolution into the surrounding water.

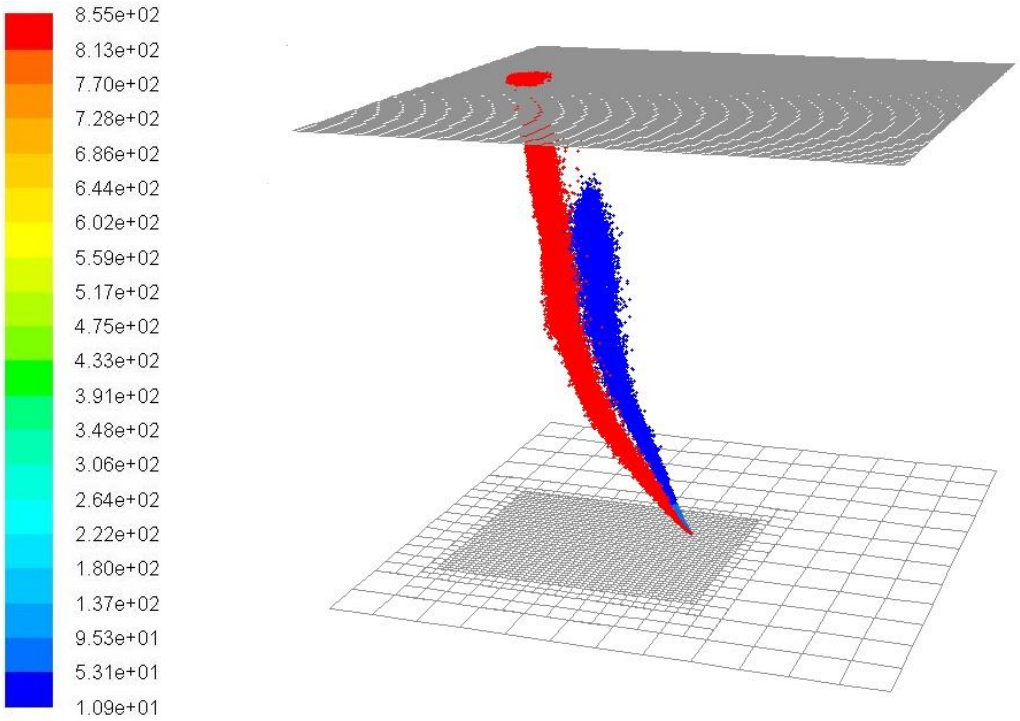

Figure 45: Parcel plot of oil droplets and gas bubbles colored by density.

As can be seen from figure 45, the oil cloud and gas bubbles are completely separated for only small regions, which may indicate the gas bubbles contributes to a higher rise velocity for the oil nearby droplets. The simulated rise time is in table 6 compared with the empirical rise time and the modelled rise time provided by the DeepBlow model.

Type:	Rise Time (s):
Experiment	3600 (1 hour)
DeepBlow	5400 (1 hour 30 min)
Current simulation	4758 (1 hour 19 min 12 sec)

Table 6: Rise time of oil - experiment, DeepBlow model, and current simulation.

As seen from table 6, both computational models over-predict the rise time of the oil release, but the current simulation provides something shorter and more accurate rise time compared with the DeepBlow model. This is further discussed in section 8.3.

8.3 Discussion

This section contains a comprehensive discussion of the various factors that may affect the results presented in the previous section, which in general indicated quite good agreement with the experimental data. CFD, and especially system of such complexity as the one studied herein, there are a large number of simplifications and approximations being made through numerical grid size, turbulence modeling, and user-defined functions that can have large and unpredictable consequences. As ocean currents dominate the path of the discrete phases, examination of the DPM concentration along the plume center line is difficult. The DPM concentration is, however, expected to be significantly lower than observed for the validation model, as discussed in section 7.4.

8.3.1 Numerical Grid Size

As explained in chapter 3, the grid determines the control volumes on which all governing equations are resolved, and thus the numbers of cells directly affects the solution accuracy and the required CPU time. A CFD model should capture any small velocity fluctuations and other features of the domain of interest by an efficient mesh scheme. However, a complete mesh independence study (ref. section 5.1.3) is not possible at the present time due to the major amount computational time required. Anyway, the largest flow gradients are localized in the jet zone (ref. chapter 2). Therefore, a refinement of the first 30 m above the release point is applied in order to analyze the effect of denser control volumes. In Figure 46, the simulation is performed on a grid with 430738 hexahedral cells, where the cells are refined to approximately 0.65 m in the region around the release point.

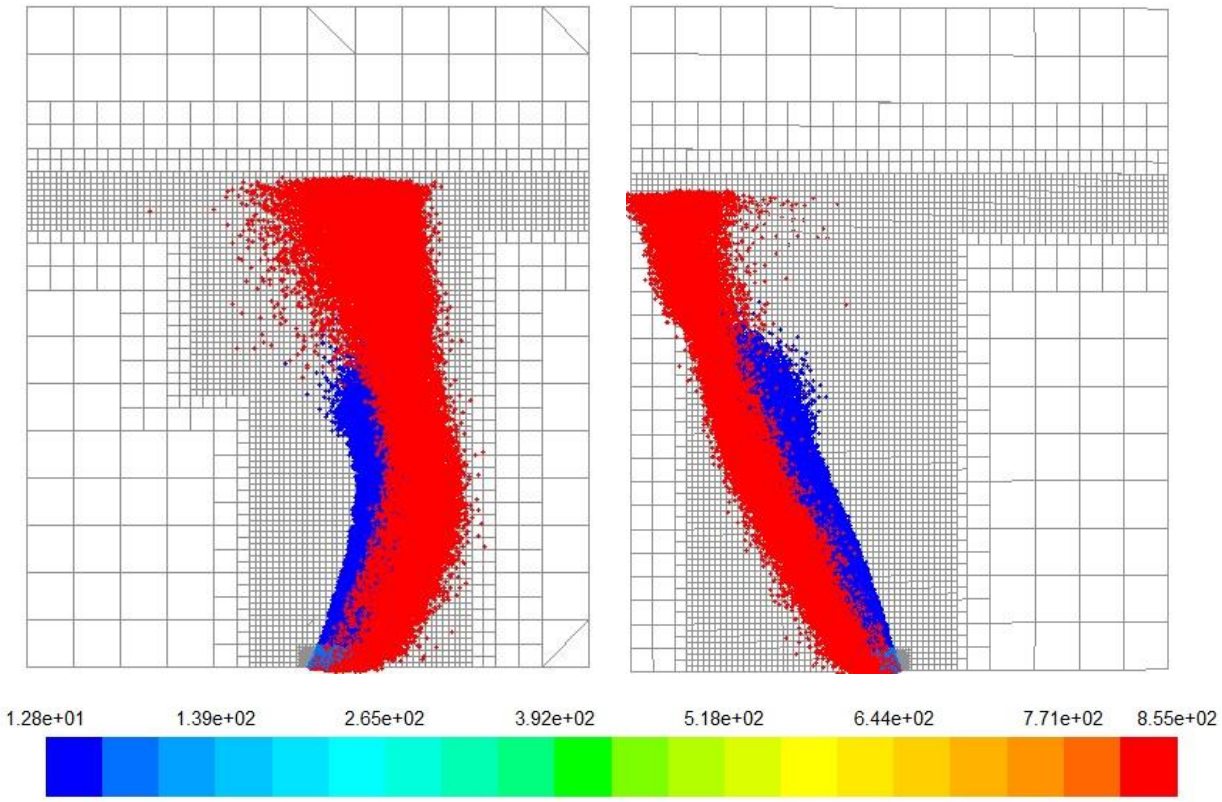

Figure 46: Parcel plot of oil and gas colored by density with a refined release zone.

Left hand side of figure 46 shows the bubble/droplet movement in east/west direction, while the right hand side of figure 46 illustrates the discrete phase movement in north/south direction. As seen from figure 46, both plumes are significantly more dispersed compared with figure 40. A greater horizontal extension of the plumes seem realistic (ref. chapter 2), while the effect of the refinement is further discussed below. In figure 47, the mean path of the oil cloud is plotted against the experimental echo-sound data and the first simulation result (ref. section 8.2.1).

In figure 47 the green line indicates the mean path of oil, where the first 30 m of the plume path is refined. As seen from figure 46, the horizontal extensions of the oil cloud are significantly greater, meaning the total plume volume is covering much larger regions of the echo-sound plot. The results presented in this section may indicate a denser grid distribution should be applied to even larger regions of the plume path. Moreover, this is supported by the calculated rise time, presented in table 7. However, an attempt of additional refinements (the first 60 m of the plume path) did provide an estimated computational time of approximately 2 weeks.

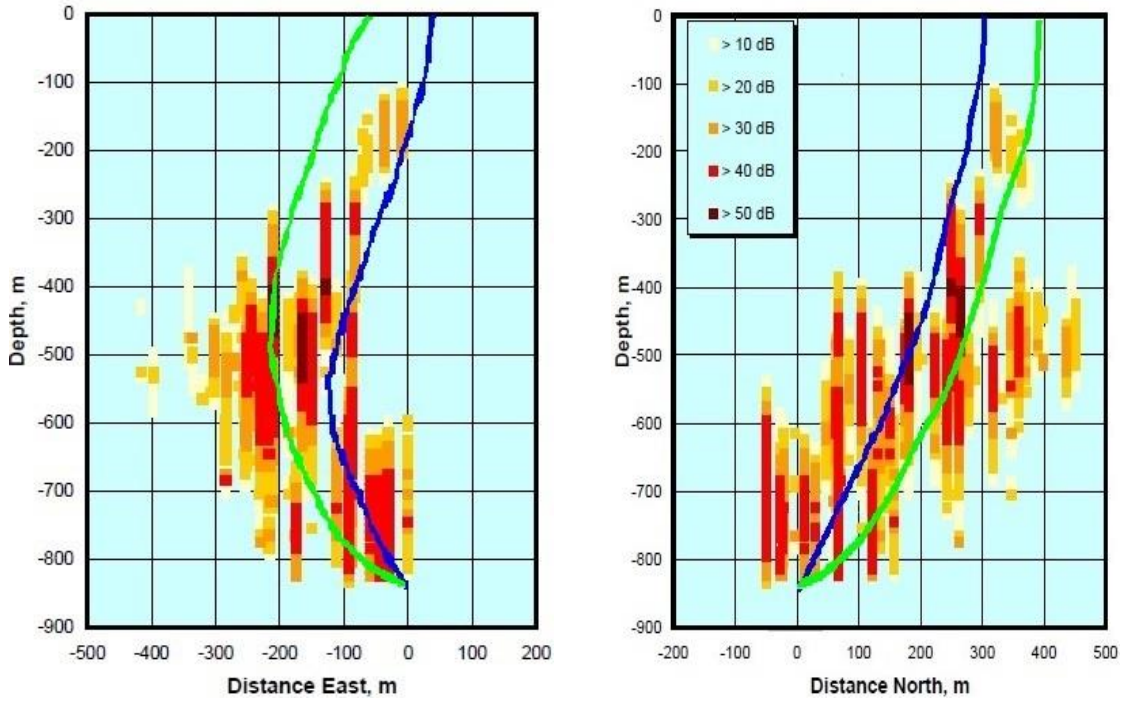

Figure 47: The mean path of oil with refined release zone (green line) plotted against experimental echo-sound data and the first simulation result from section 8.2.1.

Table 7 shows almost a 5 min faster rise time after the release zone is refined. Possible sources of the differences in simulation results presented in this section are further discussed below.

Marine Diesel Release:	Rise Time (s):
Experiment	3600 (1 hour)
DeepBlow	5400 (1 hour 30 min)
Simulation (uniform grid)	4758 (1 hour 19 min 12 sec)
Simulation (refined release zone)	4474 (1 hour 14 min 24 sec)

Table 7: Rise time – experiment, DeepBlow model, and current simulations.

I. Two-Way Coupling and Control Volume

As discussed in section 3.6.3:A, there is a two-way coupling between the continuous phase and the discrete particles. As the trajectory of a particle is computed, ANSYS FLUENT keeps track of the momentum gained or lost by the particle stream that follows that trajectory, where these momentum transfers are incorporated in the subsequent continuous phase calculations. Thus, as the continuous phase always impacts the discrete phase, the effects of the discrete phase trajectories on the continuum are incorporated, when a two-way coupling is activated. The change of momentum for the continuous phase is in ANSYS FLUENT found

by examining the drag forces exerted on a particle as it passes through each control volume of interest in the domain (ref. Equation 47). For clearness, the momentum change equation is given below:

$$f^c = -\frac{\sum_b (F_{drag} + F_{virtual\ mass})}{V_{cell}}$$

As can be seen from the equation above, the continuous phase may experience less change in momentum when surrounded by large control volumes relative to smaller volumes, by assuming the number of particles is the same. In large grid cells the forces are distributed over a greater volume, making the velocity gradients for the continuous phase smaller. In figure 48, three examples of control volumes are given.

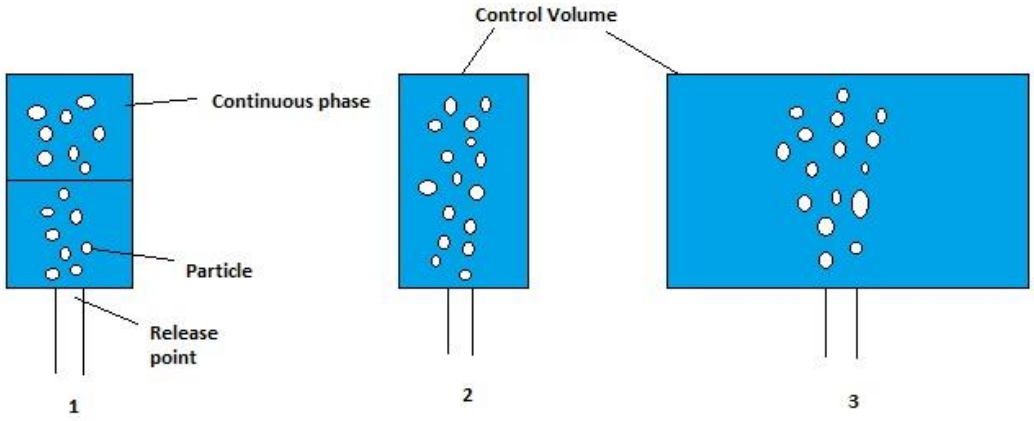

Figure 48: Three examples of control volumes containing water and bubbles.

For number 1 and 2, the total change in momentum for the continuous phase will be approximately the same, as the total number of particles and the total volume surrounding the particles are equal. However, the continuous phase in number 3 may experience minor change in momentum as the control volume is sufficiently large and the number of particles is the same as in number 1 and 2. Moreover, little change in momentum implies smaller velocity gradients. This effect is supported by figure 46 and figure 47. The motion of the oil is more in the west and north direction (ref. figure 47), suggesting a greater ocean velocity dragging more on the overall oil cloud. A faster moving continuous phase may also contribute to a lower rise time, as indicated by table 7.

II. Production of Turbulence

As discussed above, the grid size may affect the change in momentum and thus the degree of velocity gradients in the continuous phase. The production of the turbulent kinetic energy (k) is dependent of the mean velocity gradients in the ocean (ref. Equation 27). For simplicity, this equation is given below:

$$G_k = -\rho\overline{u_i' u_j'}\frac{\partial \overline{u_j}}{\partial x_i}$$

Smaller control volumes may contribute to larger change in momentum for the continuous phase, making the velocity gradients more significant and therefore the production of turbulence greater. Higher production of turbulent kinetic energy will increase velocity fluctuations (ref. Equation 36), which subsequently provides a more dispersed plume. The simulation results seem to support this reasoning, as figure 46 shows a much more dispersed discrete phase compared with figure 40. Moreover, Karacz & Kacperski (57) studied the effect of grid quality on the results of numerical simulations by applying the k-ε model. The conclusion was found to be: "Density of the computational grid significantly affects the distribution of the turbulence kinetic energy and its dissipation. More detailed features of the flow field can be captured using denser numerical grid."

In addition, the number of injected parcels should be sufficiently large so the dispersion accounts for all directions, as noted in section 3.6.2:II. A larger amount of injected parcels would most likely contribute to a greater horizontal extension of the discrete phase, however, at the expense of significantly longer computational time.

8.3.2 Additional Factors

Integral models are highly dependent of appropriate empirical data, mentioned in section 2.4. A maximum rise velocity of the oil droplets of 0.13 m/s was implemented in the DeepBlow model (34). As seen from table 7, the computed rise time is significantly larger than observed during the DeepSpill experiment. Johansen et al. (34), suggests therefore reconsideration of the rise velocity of oil droplets and their drop size formation (ref. section 4.2.4) through e.g. laboratory tests. Moreover, the DeepBlow model does not account for the presence of gas bubbles which probably will influence the rising velocity of nearby oil droplets. In ANSYS FLUENT, this effect is automatically accounted for when a two-way coupling is employed. As seen from figure 49, the oil droplets closest to the more buoyant bubble plume possesses higher rise velocity (z-velocity), than drops further away. This is a

possible cause of the discrepancy in computed rise time between the current simulations and the DeepBlow model, observed in table 7.

In the current application, a spherical drag law is applied to the oil droplets (ref. section 4.1.2). The DeepBlow model includes the Stoke law regime for Re < 1 and the constant drag regime for Re > 1000 (34). These differences in drag laws may be an additional factor for the observed discrepancy in rise time between the two computational models.

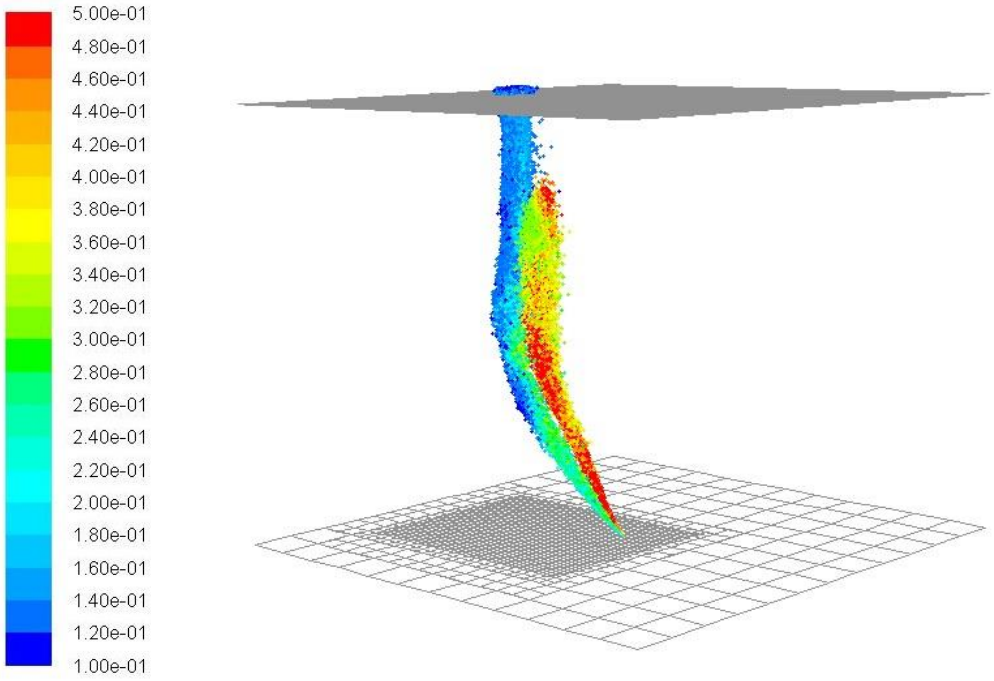

Figure 49: Parcel plot of oil droplets and gas bubbles colored by the z-velocity.

8.3.3 Reduction Factor Gas Dissolution

As noted in section 8.2.2, the solubility of methane in seawater is in principle based upon the solubility coefficients provided by Lekvam & Bishnoi (16) and the mass transfer coefficient is given by Zhang & Xu (74), presented in section 4.4. However, by inserting these values into the mass transfer equation did provide a significantly higher dissolution rate from the discrete particles than suggested by the echo-plot signals during the DeepSpill experiment. A possible source of error is that the solubility coefficients provided by Lekvam & Bishnoi (16) is measured for somewhat different water temperatures ($1℃ - 12℃$*), and that these values are rounded in the current application.*

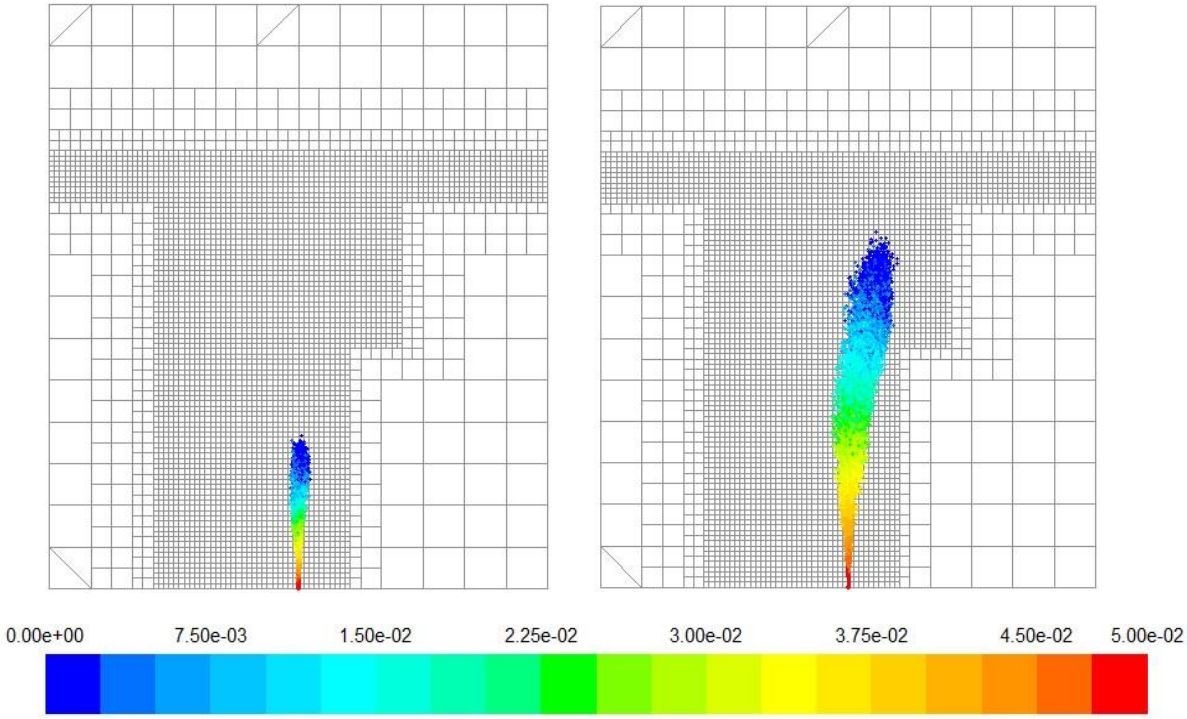

Figure 50: Parcel plot of methane gas colored by the parcel mass, with and without the reduction factor.

Moreover, saturation of the surrounding sea water is neglected in the present work, which may contribute to a slightly higher dissolution rate (ref. section 4.4 and 6.3.2). However, these assumptions are not expected to contribute to such high discrepancy as observed in figure 50. Left hand side of figure 50 shows the parcel mass without any reduction factor (tuned by author), while the right hand side of figure 50 is the same plot as presented in section 8.2.2.

A significant source of error is the assumption of ideal gas behavior. Ideal gas behavior may affect the point of complete gas dissolution substantially (ref. chapter 5 and 7). As non-ideal gas behavior (ref. figure 16) suggests a significantly higher gas density at great depths, the mass of each bubble is expected to be greater when the compressibility factor (Z-factor) is employed. By assuming ideal gas behavior the gas may therefore be trapped below the water surface at greater depths, as indicated by figure 50.

In fact, the DeepBlow model overestimated the dissolution rate as well. After a vertical rise of about 200 m, the gas bubbles were completely dissolved into the surrounding water. The calculations were based on known solubility of methane gas in sea water and mass transfer coefficients derived from laboratory experiments (34). In order to match the maximum vertical rise of about 720 m, the DeepBlow model had to include a reduction factor.

8.3.4 Modeling of Ocean Currents

As discussed in section 4.5.3:Method 3 and 6.3.4, the discrete phase determines the path of the entrained ocean plume in the current application. In a real blowouts (ref. section 2.3.2), the entrained water plume are affected by surrounding ocean current, making it bend over and possible separate from the discrete phase. However, as seen from figure 51, the water surrounding the entrained water plume is actually stationary and the discrete phases determines its path all the way towards the ocean surface.

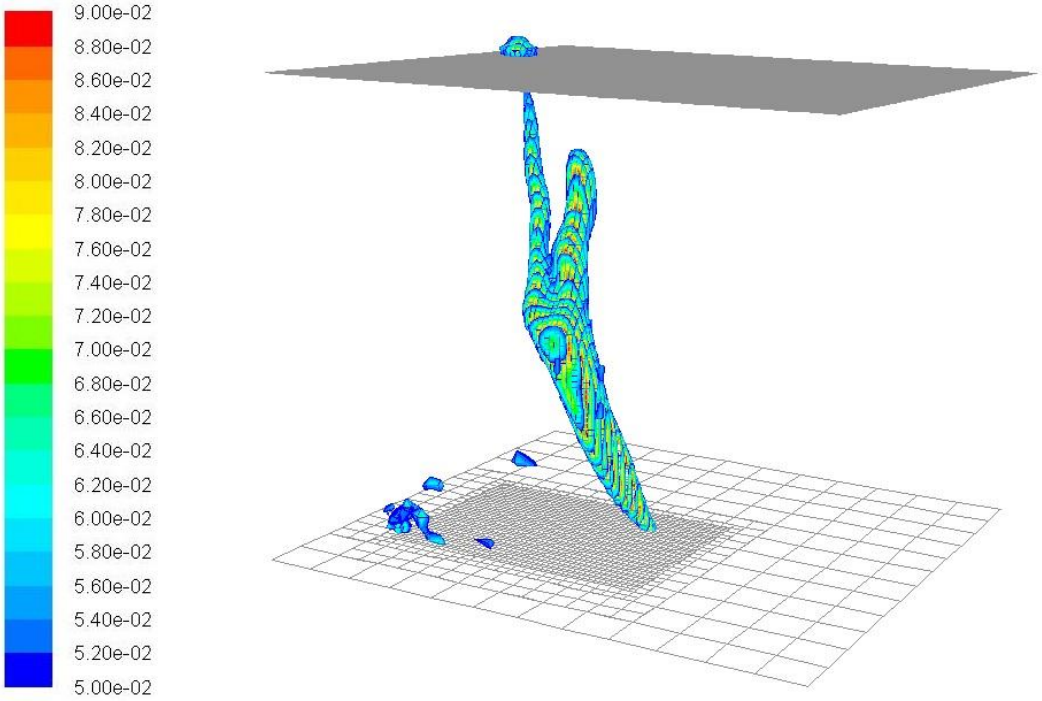

Figure 51: Water velocity in z-direction for the oil and gas experiment.

Thus, in real blowouts, the entrained water plume may drag more on the overall oil cloud, making the effect of ocean currents more significant before the oil droplets eventually starts to separate from it. However, it is hard to determine the exact impact of this assumption at this point.

8.3.5 Alternative Method

In the current application, no gas bubbles reached the ocean surface and the surface slick of oil developing is not considered. Therefore, the surface tracking technique (ref. section 3.6.3:A) implemented in the VOF model is more or less superfluous for the primary model. As tracking of a liquid-gas interfaces requires relatively fine and uniform grid distribution (ref. section 5.1.2), the computational time is increased. An alternative method may be to define the water surface boundary as a wall, allowing the surfacing oil droplets to escape. If desired, the position of all surfacing oil droplets can be written to a file using the user-defined function "DeepSpill.c" (ref. appendix B.1). However, as the model is supposed to be valid for shallower depths and/or higher gas flow rates, for which gas bubbles may surface, interface tracking is employed in the primary model.

8.4 Further Work

This section contains suggestions for further work related CFD simulation of deep water oil and gas blowouts.

1. *In the present work the size distribution of oil droplets is approximated by a Rosin-Rammler distribution, as discussed in section 4.2.3. However, Johansen et al. (34) recommends further evaluation of the droplet size formation. Laboratory tests is the cheapest and most efficient way of observe such formation. These suggestions are based upon the discrepancy in rise time modelled by the DeepBlow model, which is acquired both the modelled maximum rise velocity and the oil droplet size distribution (34). An evaluation of drop size may improve the rise time of oil simulated in the primary model.*

2. *Non-ideal gas behavior should definitely be accounted for in the present work, as discussed in section 5.2.1 and 7.4.1. This is, however, done in the work of Skjetne & Olsen (2) by use of the Peng-Robinson equation of state. As seen from figure 16, the non-ideal gas behavior may be implemented for depths below 250 m.*

   During the DeepSpill experiment no hydrate formation was observed, as discussed in section 2.1. However, multiple literature sources asserts that natural gas exiting from a subsea release point into high water pressure and low temperature quickly combines with the cold water and form the solid ice-like substance known as hydrate.

Moreover, Johansen (15) claims that: "For discharges in deep waters (700-1500 m depth), hydrate formation is found to be a dominating process in limiting plume rise." The lack of hydrate formation in the DeepSpill experiment is explained by Johansen et al. (34) by the absence of the dissolved gas saturation necessary for sustained hydrate formation. Therefore, equations related hydrate formation may be necessary to implement in forecasting of blowout incidents at other locations and depths.

3. *As discussed in section 2.3.1 and 6.3.3, the effect of stratification in the ambient sea water is neglected in the current application. Density stratification may affect the plume dynamics considerably as entrained water cannot be detrained from the water plume (72) and thus the process of water mixing is obstructed.*

4. *Oil and gas computer models should account for the presence of wind affecting oil slicks developing at the sea surface. In the present work, such slick formation is not evaluated and the effect of shear forces between wind and sea surface is neglected. Since the water surface is captured by the VOF model, the thickness of the developing oil slick could easily be measured and compared with experimental data presented in the work of Johansen et al. (34). However, as wind forces are neglected the results are expected to not compare very well with the measurements of oil slicks during the DeepSpill experiment. Wind forces may drag an oil slick in various directions, making the thickness of the oil slick thinner.*

5. *As seen from section 8.2.1 and section 8.3.1, the plume dynamics is highly dependent on the numerical grid size, especially in the lower regions where the largest flow gradients are located. A mesh independence study should be applied to determine an appropriate grid size for deep water oil and gas releases. The best solution may be to activate transient mesh adaptation in the ocean region, while the sea surface is given a fixed mesh to avoid numerical instabilities, as discussed in section 5.1.2. However, this requires major amounts of computational time available.*

Chapter 9

Concluding Remarks

The main objective of this thesis was to provide a general framework in CFD of deep water oil and gas releases. The fundamental theoretical framework and model set-up is based on work developed by Cloete (5) ,Cloete et al. (11), Skjetne & Olsen (2), and generalized to allow for presence of oil droplets and plume tracking, underwater currents and gas dissolution.

Results and discussion of the validation model is presented in chapter 7, followed by some concluding remarks.

In the present work, a user-defined drag force estimates the effect of ocean currents on each bubble/ droplet. An acceleration term is then returned to each particle, in which replaces the effect of a moving continuous phase. Simulation results of the mean cloud path of oil are found to correspond quite well with the overall shape of the echo-sound images taken during the DeepSpill experiment. Grid refinements of the release zone, where the highest gradients of flow variables are located, indicates an improved mean cloud path of oil in the lower ocean region and significantly greater plume dispersion is observed. Moreover, an improved rise time is computed when the first 30 m of the plume path consists of a denser grid distribution. This may indicate that a larger region of the plume path requires mesh refinement, which seems to contribute to greater change in momentum for the continuous phase and higher production of turbulent kinetic energy, as discussed in section 8.3. However, due to the major amount of computational time required, additional refinements are not possible at the present time.

Gas dissolution is based on solubility coefficients provided by Lekvam & Bishnoi (16) and the mass transfer coefficient expressed by Zhang & Xu (74). Gas bubbles are assumed to be contaminated by surfactants and non-ideal gas behavior is neglected. The mean cloud path of methane gas is observed to be centered well within the echo-sounder signals. However, a mass transfer reduction factor is applied to match the point of complete gas dissolution observed from the echo-signals, which may to a certain be extent be caused by the assumption of ideal gas behavior.

The DeepBlow model (15; 34; 37), which is an integral model highly dependent of appropriate empirical data (e.g. bubble and oil size distribution, entrainment coefficients, and maximum rise velocity), is shown to model quite similar mean cloud path of oil as the primary model. However, the DeepBlow model provided a longer rise time compared with both experimental data and simulation results, presented in chapter 9. This discrepancy may

be due the fact that the DeepBlow model does not include the effect of more buoyant gas bubbles which may increase the rise velocity of nearby oil droplets. This is, however, automatically accounted for in ANSYS FLUENT when a two-way coupling is employed. Moreover, the bubble size model, implemented in the present work, provides a more general distribution of bubble sizes as turbulence and ocean properties determines the frequency of bubble breakup and coalescence. Integral models are, however, dependent of appropriate empirical data.

As indicated above, the overall simulation results are found to yield quite good agreement with the DeepSpill experiment:

- *The mean cloud path of oil corresponds quite well with the overall shape of the echo-sound images taken during the oil and gas release.*
- *The mean cloud path of methane gas is observed to be well centered within the echo-sounder signals. Moreover, the point of complete gas dissolution matches the experimental data, when the mass transfer reduction factor is applied.*
- *The rise time of oil is about 15 min longer than observed during the DeepSpill experiment, which is more accurate than the rise time provided by the DeepBlow model (30 min longer).*

However, further work on the model is necessary before it may be applied for risk assessments of subsea oil and gas blowouts, such as:

- *Implementation of non-ideal gas behavior.*
- *Laboratory tests of oil droplet size distribution and measurements of drag forces related rising oil drops in sea water.*
- *Implementation of ocean stratification.*
- *For higher gas flow rates and/or shallower water depths surface damping effects should be employed.*
- *A complete mesh independence study is necessary to provide reliable simulation results.*

A mesh independency study of blowouts at such great depths, as investigated in the current application, requires great amounts of computational time available. This is a major drawback for CFD simulations of oil and gas releases. Integral models are, on the other hand, much cheaper and more efficient. For that reason, integral models, such as the DeepBlow model, may be preferred in forecasting of blowout incidents when appropriate empirical data is available.

Bibliography

1. *SL ROSS, "Oil Spill Fate and Behavior Modelling in Support of Corridor Resources Old Harry Exploratory Drilling Enviromental Assessment Updated Report," S.L Ross, Ottawa, 2012.*

2. *P. Skjetne and J. E. Olsen, "A parcel based modelling concept for studying subsea gas release and the effect of gas dissolution,"* Progress in Computational Fluid Dynamics, Vol. 12, Nos. 2/3, *pp. 187-195, 2012.*

3. *A. Sayma, Computational Fluid Dynamics, Abdulnaser Sayma & Ventus Publishing Aps, 2009.*

4. *J. D. J. Anderson, Computational Fluid Dynamics, The Basics with Applications, New York: McGraw-Hill Book Co., 1995.*

5. *S. W. P. Cloete, "A Mathematical Modelling Study of Fluid Flow and Mixing in Gas Stirred Ladles," Department of Process Enginerring at Stellenbosch University, Stellenbosch, 2008.*

6. *MIT, "www.mit.edu," 2011. [Online]. Available: http://www.mit.edu/course/1/1.061/www/dream/SEVEN/SEVENTHEORY.PDF (Obtained: 05.10.2014).*

7. *H. Oertel, Prandtl's Essentials of Fluid Mechanics, New York: Springer, 2003.*

8. *C. T. Crowe, Multiphase flow handbook, Washington: Taylor & Francis Online, 2006.*

9. *L. Mazzei, "Eulerian modelling and computational fluid dynamics simulation of mono and polydisperse fluidized suspensions," University College London, London, 2008.*

10. *J. Xia, T. Ahokainen and L. Holappa, "Analysus of flows in a ladle with gas-stirred melt,"* Scandinavian Journal of Metallurgy, *pp. 69-76, 2001.*

11. *S. Cloete, J. E. Olsen and P. Skjetne, "CFD modeling of plume and free surface behavior resulting from a sub-sea gas release,"* Applied Ocean Research 31, *pp. 220-225, 2009.*

12. *W. L. Shew, S. Poncet and J.-F. Pinton, "Force measurements on rising bubbles," Laboratoire de Physique, Lyon, 2006.*

13. *Y. Sheng and G. Irons, "Measurement and Modelling of Turbulence in the Gas/Liquid Two-Phase Zone during Gas Injection,"* Metallurgical and Materials Transactions Vol. 24B, *pp. 695-705, 1993.*

14. *F. M. White, Fluid Mechanics Sixth Edition, New York: McGraw-Hill, 2008.*

15. *Ø. Johansen, "DeepBlow - a Lagrangian Plume Model for Deep Water Blowouts,"* Spill Science & Technology, Vol 6, No. 2, *pp. 103-111, 2000.*

16. *K. Lekvam and P. R. Bishnoi, "Dissolution of methane in water at low temperatures and intermediate pressures,"* Fluid Phase Equlibria, *pp. 297-309, 1997.*

17. *F. Millero, "The activity coefficients of non-electrolytes in seawater,"* Marine Chemistry 70, *pp. 5-22, 2000.*

18. *R. Clift, J. R. Grace and M. E. Weber, Bubbles, Drops, and Particles, London: ACADEMIC PRESS, 1978.*

19. *F. P. Incropera, D. P. Dewitt, T. L. Bergman and A. S. Lavine, Fundamentals of Heat and Mass Transfer, Sixth Edition, Notre Dame: John Wiley & Sons, 2007.*

20. *L. Zheng and P. D. Yapa, "Modeling gas dissolution in deepwater oil/gas spills,"* MARINE SYSTEMS, *pp. 299-309, 2002.*

21. *L. Zeng and P. D. Yapa, "Buoyant velocity of spherical and non-spherical bubbles/droplets," Clarkson University, New York.*

22. *M. J. Friedel and T. Fanneløp, "Bubble plumes and their interaction with the water surface,"* Applied Ocean Research 22, *pp. 119-128, 2000.*

23. *K. E. Einardsrud, "Kinetic energy approach to dissolving multiphase plumes," NTNU, Trondheim, 2011.*

24. *J. Milgram, "Mean flow in round bubble plumes,"* J. Fluid Mech. vol 133, *pp. 345-376, 1983.*

25. *H. Laux and S. T. Johansen, "A CFD analysis of the air entrainment rate due to a plunging steel jet combining mathematical models for dispersed and separated multiphase flows," Fluid Flow Phanomena in Metals Processing, Trondheim, 1999.*

26. *B. Smith, "On the modelling of bubble plumes in a liquid pool,"* Applied Mathematical Modelling 22, *pp. 773-797, 1998.*

27. *O. J. Tveit, "Risiko knyttet til gassutslipp under vann," Petroleumstilsynet, 2006.*

28. *A. FLUENT, "ANSYS FLUENT Theory Guide," 2011.*

29. *M. Bettelini and T. Fanneløp, "Underwater plume from an instanteously started source,"* Ocean Research 15, *pp. 195-206, 1993.*

30. *S. Thorpe, An introduction to ocean turbulence, New York: Cambridge University Press, 2005.*

31. *J. Sodja, Turbulence models in CFD, Ljubljana: University of Ljubljana, 2007.*

32. *E. Furbo, "Evaluation of RANS turbulence models for flow problems with significant impact of boundary layers," Uppsala Universitet, Uppsala, 2010.*

33. *W. Reynolds, Computation of turbulent flows, Stanford: Annual Reviews, 1976.*

34. *Ø. Johansen, H. Rye, A. G. Melbye, H. V. Jensen, B. Serigstad and T. Knutsen, "Deep Spill JIP - Experimental Discharges of Gas and Oil at Helland Hansen," SINTEF, Trondheim, 2000.*

35. *P. D. Yapa and L. K. Dasanayaka, "Role of plume dynamics phase in a deepwater oil and gas release model,"* Hydro-enviroment Research, *pp. 243-253, 2009.*

36. *P. D. Yapa and F. Chen, "A model for simulating deepwater oil and gas blowouts - Part I: Theory and model formulation,"* Journal of Hydraulic Resaerch Vol.41, No. 4, *pp. 339-351, 2002.*

37. *Ø. Johansen, H. Rye and C. Cooper, "DeepSpill - Field Study of a Simulated Oil and Gas Blowout in Deep Water,"* Spill Science & Technology Bulletin, Vol. 8 Nos. 5-6, *pp. 433-443, 2003.*

38. *A. Adcroft, R. Hallberg, J. P. Dunne, B. L. Samuels, J. Galt, C. H. Barker and D. Payton, "Simulation of underwater plumes of dissolved oil in the Gulf of Mexico," GEOPHYSICAL RESEARCH LETTERS, Princeton, 2010.*

39. *B. B. Maini and P. R. Bishnoi, "Experimental invetigation of hydrate formation behaviour of a natural gas bubble in a simulated deep sea enviroment,"* Chemical Enginnering Science Vol. 36, *pp. 183-189, 1980.*

40. *S. A. Socolofsky, E. E. Adams and C. R. Sherwood, "Formation dynamics of subsurface hydrocarbon intrusions following the Deepwater Horizon blowout," GEOPHYSICAL*

RESEARCH LETTERS, VOL. 38, 2011.

41. *S. Ross, "Fate and Behavior of Deepwater Subsea Oil Well Blowouts in the Gulf of Mexico," S.L. Ross Enviromental Research Ltd., 1997.*

42. *J. Domgin, P. Gardin and M. Burnet, "Experimental and numerical investigation of gas stirred ladles," Second International Conference on CFD in the Minerals and Prosess Industries, 1999, 1999.*

43. *A. Massê , "Experiments and Numerical Simulations of the Flow Within a Model of a Hydraulic Turbine Surge Chamber," Deparment of Mechanical Engineering McGill University, Montreal, 2010.*

44. *M. R. Davidson, "Numerical calculations of two-phase flow in a liquid bath with bottom gas injection: The central plume," CSIRO Division of Mineral and Process Enginerring, Lucas Heights Research Laboratories, Lucas Heights, 1990.*

45. *M. Schmidtke and D. Lucas, "CFD Approaches for Modelling Bubble Entraiment by Impinging Jet,"* Hindawi Publishing Corporation Science and Technology Installations Volume 2009, *p. 12, 2008.*

46. *A. d. Souza, "Understand Computational Fluid Dynamics," NAFEMS Ltd, 2005.*

47. *Q. Pan, "Modelling of Turbulent Flows with Strong Dispersed Phase-Continuous Fluid Interactions," NTNU, Trondheim, 2014.*

48. *H. Laux and S. T. Johansen, "A CFD analysis of the air entrainment rate due to a plunging steel jet comining mathematical models for dispersed and separated multiphase flows," SINTEF, Trondheim.*

49. *Y. Sheng and G. Irons, "Measurements of the Internal Structure of Gas-Liquid Plumes," Academic Press, London, 1972.*

50. *E. W. North, E. E. Adams, Z. Schlag, C. R. Sherwood, R. He, K. H. Hyun and S. A. Socolofsky, "Simualting Oil Droplet Dispersal From the DeepWater Horizon Spill With a Lagrangian Approach," American Geophysical Union, 2011.*

51. *S. Johansen, T. Ytrehus and K. Einardsrud, Modeling of Multiphase Flows, Trondheim: NTNU.*

52. *J. E. Olsen and S. Cloete, "Coupled DPM and VOF Model for Analyses of Gas Strirred Ladles at Higher Gas Rates," SINTEF Materials & Chemistry, Trondheim, 2009.*

53. *M. R. Snyder, "Analysis of the Behavior of Bubbles and Droplets in Isotropic Turbulence," The Johns Hopkins University, Baltimore, Maryland, 2007.*

54. *T. Engebretsen, T. Northug, K. Sjøen and T. Fanneløp, "Surface Flow and Gas Dispersion from a Subsea Relase of Natural Gas," The International Society of Offshore and Polar Engineers, Honolulu, 1997.*

55. *H. K. Versteeg and W. Malalasekera, An Introduction to Computational Fluid Dynamics The Finite Volume Method second edition, Loughborough: Pearson Education Limited, 2007.*

56. *Ø. Johansen, "Development and verification of deep-water blowout models,"* Marine Pollution Bulletin 47, *pp. 360-368, 2003.*

57. *J. Karcz and L. Kacperski, "AN EFFECT OF GRID QUALITY ON THE RESULTS OF NUMERICAL SIMULATIONS OF THE FLUID FLOW FIELD IN AN AGITATED VESSEL," West Pomeranian University of Technology, Szczecin, Department of Chemical Engineering, Szczecin, 2012.*

58. *R. K. Dewey, J. N. Moum, C. A. Paulson, D. R. Caldwell and S. D. Pierce, "Structure and Dynamics of a Coastal Filament,"* Journal of Geophysical Research, Vol. 96 NO. C8, *pp. 14,885-14,907, 1991.*

59. *T. K. Fanneløp and M. Bettelini, "Very Large Deep-Set Bubble Plumes From Broken Gas Pipelines," Petroleumstilsynet, 2007.*

60. *P. D. Yapa, "people.clarkson.edu," Clarkson University, February 2011. [Online]. Available: http://people.clarkson.edu/~pyapa/cdog.htm.*

61. *T. Fanneløp and K. Sjøen, "Hydrodynamics of underwater blowouts," 18th Aerospace Science Meeting, American Inst. of Aeronautics and Astronautics, Pasadena, CA, 1980.*

62. *ASCE, "The state-of-the-art of modeling oil spills,"* Journal of Hydraulic Engineering, *vol. 122, no. 11, pp. 594-609, 1996.*

63. *T. McDougall, "Bubble plumes in stratified enviroments,"* Journal of Fluid Mechanics, *vol. 133, pp. 655-672, 1978.*

64. *M. Spaulding, "A state-of-the-art reveiw of oil spill trajectory and fate modelling,"* Oil and Chemical Pollution, *vol. 4, pp. 39-55, 1988.*

65. *T. Fannelop, S. Horschberg and J. Kuffer, "Surface current and recirculating cells generated by bubble curtains and jets,"* Journal of Fluid Mechanics, *vol. 229, pp. 629-657,*

1991.

66. *P. Yapa and L. Zheng, "Simulation of oil spills form underwater accidents I: Model development,"* Journal of Hydraulic Research, *pp. 673-688, 1997.*

67. *R. Bhaskaran and L. Collins, "Introduction to CFD Basics".*

68. *S. Johansen, D. Robertson, K. Woje and T. Engh, "Fluid Dynamics in Bubble Stirred Ladles: Part I. Experiments," SINTEF, Trondheim, 1986.*

69. *B. Launder and D. Spalding, Lectures in mathematical models of turbulence, London: Academic Press, 1972.*

70. *J. W. Han, S. H. Heo, D. H. Kam, B. D. You, J. J. Pak and H. S. Song, "Transient Fluid Flow Phenomena in a Gas Stirred Liquid Bath with Top Oil Layer - Appriach by Numerical Simulation and Water Model Experiments,"* ISIJ International, *vol. 41, pp. 1165-1172, 2001.*

71. *Z.-G. Feng and E. Michaelides, "Drag coefficients of viscous spheres at intermediate and high reynolds numbers,"* Transactions of the ASME: Journal of Fluids Engineering, *vol. 123, pp. 841-849, 2001.*

72. *P. Vollestad, "Bubble plume modelling, Including gas dissolution in an integral model," Norwegian University of Science and Technology (NTNU), Trondheim, 2013.*

73. *A. Karabelas, "Droplet size spectra generated in turbulent pipe flow of dilute liquid/liquid dispersion,"* AICHE Journal, *vol. 24, pp. 70-180, 1978.*

74. *Y. Zhang and Z. Xu, "Kinetics of convective crystal dissolution and melting, with applications to methane hydrate dissolution and dissociation in seawater,"* Earth and Planetary Science Letters , *vol. 213, pp. 133-148, 2003.*

75. *NASA, "Ocean Motion and Surface Currents," NASA, [Online]. Available: http://oceanmotion.org/html/background/ocean-in-motion.htm.*

76. *ANSYS, "ANSYS FLUENT UDF Manual," ANSYS, Inc., 2011.*

77. *ANSYS, "ANSYS FLUENT User's Guide," ANSYS Inc., 2011.*

78. *CFD-online, "cfd-online.com," [Online]. Available: http://www.cfd-online.com/Wiki/Mesh_adaptation (Obtained: 14.05.2014).*

79. *C. Soga and C. Rehmann, "Dissipation of turbulent kinetic energy near a bubble plume,"* Journal of hydraulic engineering, *vol. 130, pp. 441-449, 2004.*

80. *N. Bowditch, The American Practical Navigator Pub. No. 9, Bethesda, Maryland: National Imagery and Mapping Agency, 2002.*

Appendix A

User-defined Functions: Validation Model

The following sections contain user-defined functions (UDFs) used for the calculations done in chapter 7.

A.1 "Validation_Model.c"

The UDF "Validation_Model.c" accounts for size distribution (ref. section 4.2.4) and density changes (ref. section 4.2.1) of buoyant air-bubbles. The position (x- and y-position) of surfacing bubbles is first written to a file together with their residence time and parcel mass, before deleted due to computational efficiency (ref. 7.1.3). The UDF allows for parallel calculations, which implies the use of multiple CPUs. Senior Technical Consultant Love Håkansson at EDRMedeso provided very valuable help in order to achieve this.

```
#include "udf.h"


real depth = 7.0;
real injection_height = 0.33;
real Temp = 15.0;
real Pressure_REF = 101325.0;
real MolarWeight = 28.97; /* air */

/* Constants used in the bubble size model */
real SurfaceTension = 0.07199;
real C1 = 4.0;
real C2 = 100.0e-6;

real write_x_position;
real write_y_position;

static int position_var = 1;
static int position_var1 = 1;


#if !RP_NODE
FILE *ptr_file;
```

```
#endif /* !RP_NODE */

struct P_List_Item_ {
      real p_radius;
      real p_mass;
      real write_x_position;
      real write_y_position;
      struct P_List_Item_ *next;
};

typedef struct P_List_Item_ P_List_Item;

DEFINE_ON_DEMAND(Init_file)
{
#if !RP_NODE
      ptr_file = fopen("surfaceflux.txt", "w");
#endif /* !RP_NODE */
}

DEFINE_ON_DEMAND(Term_file)
{
#if !RP_NODE
      fclose(ptr_file);
#endif /* !RP_NODE */
}

#if !RP_HOST
int doRemove(Particle *p){

      if (P_POS(p)[2] > 0.05) return 1;
      else return 0;
}

void remove_p(Particle *p){
      p->flags = 4;
}

P_List_Item *add_to_list(P_List_Item *head, P_List_Item *tail, Particle
*p){
      if (head->next == NULL){
            head->next = tail;
      }
      else {
            tail->next = (P_List_Item *)malloc(sizeof(P_List_Item));
            tail = tail->next;
            tail->next = NULL;
      }
      /* distance from plume center line */
      tail->p_radius = sqrt(pow(P_POS(p)[0], 2) + pow(P_POS(p)[1], 2));
      /* parcel mass */
       tail->p_mass = P_N(p)*P_MASS(p);
      /* x-position */
      tail->write_x_position = P_POS(p)[0];
      /* y-position */
```

```
	tail->write_y_position = P_POS(p)[1];

	return tail;
}

void clear_list(P_List_Item *head){
	P_List_Item *temp, *item;
	item = head;
	while (item != NULL){
		temp = item->next;
		free(item);
		item = temp;
	}
	head = NULL;
	free(temp);
	temp = NULL;
	free(item);
	item = NULL;
}


void send_list(P_List_Item *head, int n){
	real *p_prop1;
	real *p_prop2;
	real *p_prop3;
	real *p_prop4;

	int i = 0, j;
	P_List_Item *item;
	item = head->next;

	if (n!=0) {
		p_prop1 = (real *) malloc(sizeof(real)*n);
		p_prop2 = (real *) malloc(sizeof(real)*n);
		p_prop3 = (real *)malloc(sizeof(real)*n);
		p_prop4 = (real *)malloc(sizeof(real)*n);

		for (i=0;i<n;i++){
			*(p_prop1+i) = item->p_radius;
			*(p_prop2+i) = item->p_mass;
			*(p_prop3+i) = item->write_x_position;
			*(p_prop4+i) = item->write_y_position;
			item = item->next;
		}

		if (!I_AM_NODE_ZERO_P){
			PRF_CSEND_INT(node_zero, &n, 1, myid);
			PRF_CSEND_REAL(node_zero, p_prop1, n, myid);
			PRF_CSEND_REAL(node_zero, p_prop2, n, myid);
			PRF_CSEND_REAL(node_zero, p_prop3, n, myid);
			PRF_CSEND_REAL(node_zero, p_prop4, n, myid);
		}
		else {
			PRF_CSEND_INT(node_host, &n, 1, myid);
```

```
                PRF_CSEND_REAL(node_host, p_prop1, n, myid);
                PRF_CSEND_REAL(node_host, p_prop2, n, myid);
                PRF_CSEND_REAL(node_host, p_prop3, n, myid);
                PRF_CSEND_REAL(node_host, p_prop4, n, myid);

          }
          free(p_prop1);
          free(p_prop2);
          free(p_prop3);
          free(p_prop4);

     }
     else {
          if (!I_AM_NODE_ZERO_P){
                PRF_CSEND_INT(node_zero, &n, 1, myid);
          }
          else {
                PRF_CSEND_INT(node_host, &n, 1, myid);
          }
     }

     if (I_AM_NODE_ZERO_P){
          compute_node_loop_not_zero(j)
          {
          PRF_CRECV_INT(j, &n, 1, j);
          if (n!=0){
                p_prop1 = (real *) malloc(sizeof(real)*n);
                p_prop2 = (real *) malloc(sizeof(real)*n);
                p_prop3 = (real *) malloc(sizeof(real)*n);
                p_prop4 = (real *) malloc(sizeof(real)*n);

                PRF_CRECV_REAL(j, p_prop1, n, j);
                PRF_CRECV_REAL(j, p_prop2, n, j);
                PRF_CRECV_REAL(j, p_prop3, n, j);
                PRF_CRECV_REAL(j, p_prop4, n, j);

                PRF_CSEND_INT(node_host, &n, 1, myid);
                PRF_CSEND_REAL(node_host, p_prop1, n, myid);
                PRF_CSEND_REAL(node_host, p_prop2, n, myid);
                PRF_CSEND_REAL(node_host, p_prop3, n, myid);
                PRF_CSEND_REAL(node_host, p_prop4, n, myid);

                free(p_prop1);
                free(p_prop2);
                free(p_prop3);
                free(p_prop4);

                }
                else {
                      PRF_CSEND_INT(node_host, &n, 1, myid);
                }
          }
     }
}
```

```
#endif /* !RP_HOST */

DEFINE_EXECUTE_AT_END(Remove_and_write_to_file)
{/* executed at end of every time step */
#if !RP_HOST
      Domain *d = Get_Domain(1);
      Injection *i, *i_all;
      Particle *p;
      cell_t c;
      Thread *t;
      real  parcelmass_pre, parcelmass, P, gas_density, solubility;
      real x[ND_ND];
      real dpm_frac, visc, visc_dpm, rho, d_eq, d_pre, tau, t_k, dt;
      real eps, k;

      P_List_Item *head, *tail, *item, *temp;
      int np_removed=0;

      i_all = Get_dpm_injections();
      head = (P_List_Item *)malloc(sizeof(P_List_Item));
      tail = (P_List_Item *)malloc(sizeof(P_List_Item));
      head->next = NULL;
      tail->next = NULL;

      thread_loop_c(t, d)
      {

            begin_c_loop(c, t)
            {
            /* calculate volume fraction of DPM-phase  */
            C_CENTROID(x, c, t);
            P = Pressure_REF + 9.81*C_R(c, t)*(depth - x[2]);
            P = MAX(Pressure_REF, P);
            gas_density = P * MolarWeight * 0.001 / (8.314 * (Temp +
            273.15));
            C_UDMI(c, t, 0) = C_DPMS_CONCENTRATION(c, t) / gas_density;
            C_UDMI(c, t, 1) = MIN(C_UDMI(c, t, 0), 0.9);
            }
            end_c_loop(c, t)


            /* loop through particles */
            loop(i, i_all){
            loop(p, i->p){

            cell_t c0 = P_CELL(p);
            Thread *t0 = P_CELL_THREAD(p);
            Thread *t0_ocean = THREAD_SUB_THREAD(t0, 1);
            Thread *t0_atmos = THREAD_SUB_THREAD(t0, 0);

            parcelmass_pre = P_MASS(p)*P_N(p);
            /* calcualtion of hydrostatic pressure for each bubble */
            P = Pressure_REF + 9.81*C_R(c0, t0)*(depth - P_POS(p)[2]);
            P = MAX(Pressure_REF, P);
```

```
            /* bubble density */
            P_RHO(p) = P * MolarWeight * 0.001 / (8.314 * (Temp + 273.15));

            /* bubble size model, Pan(47) */
            visc_dpm = 1.7849e-05;
            visc = C_MU_L(c0, t0_ocean);
            eps = MAX(C_D(c0, t0), 1.0e-6);
            k = C_K(c0, t0);
            rho = C_R(c0, t0);
            dpm_frac = C_UDMI(c0, t0, 1);
            dt = CURRENT_TIMESTEP;
            /* equlibrium diameter */
            d_eq = C1 * sqrt(dpm_frac) * (pow(SurfaceTension / rho, 0.6) /
            pow(eps, 0.4))*(pow(visc_dpm / visc, 0.25)) + C2;
            d_pre = P_DIAM(p);
            t_k = 6.0*sqrt(visc / (rho*eps));
            if (d_pre > d_eq) /* breakup */
                {
                  tau = pow(d_pre, 0.66667)*pow(eps, -0.33333333);
                  }
                  else /* coalescence */
                  {
                  tau = d_pre / (0.2*6.0*MAX(1.0e-06, sqrt(dpm_frac*k)));
                  }

                  tau = MAX(tau, t_k);
                  P_DIAM(p) = (d_pre + d_eq*dt / tau) / (1.0 + dt / tau);
                  P_DIAM(p) = MAX(P_DIAM(p), 0.0001); /* bubble diameter */
                  P_MASS(p) = P_RHO(p) * M_PI * pow(P_DIAM(p), 3.0) / 6.0;
                    /* number of particles in a parcel */
                  P_N(p) = parcelmass_pre / P_MASS(p);
                  /* delete surfacing bubbles */
                  if (C_VOF(c0, t0_ocean) < 0.1){
                        np_removed++;
                        tail = add_to_list(head, tail, p);
                        remove_p(p);
                        }
                  }
            }
      }
      send_list(head, np_removed);
      clear_list(head);
      head = NULL;
      tail = NULL;

#endif /* !RP_HOST */

#if !RP_NODE
      int j, *n, k;
      real *temp1, *temp2, *temp3, *temp4;
      n = (int *) malloc(sizeof(int));
      compute_node_loop(j)
      {
            PRF_CRECV_INT(node_zero, n, 1, node_zero);
```

```
            Message("Node %d: %d particles removed.\n", j, *n);
            if (*n!=0){
                  temp1 = (real *) malloc(sizeof(real)*(*n));
                  temp2 = (real *) malloc(sizeof(real)*(*n));
                  temp3 = (real *) malloc(sizeof(real)*(*n));
                  temp4 = (real *) malloc(sizeof(real)*(*n));

                  PRF_CRECV_REAL(node_zero, temp1, *n, node_zero);
                  PRF_CRECV_REAL(node_zero, temp2, *n, node_zero);
                  PRF_CRECV_REAL(node_zero, temp3, *n, node_zero);
                  PRF_CRECV_REAL(node_zero, temp4, *n, node_zero);


                  for (k=0;k<(*n);k++){
                  /* print distance, mass, position, and residence time */
                  fprintf(ptr_file, "%g %g %g %g %g\n", *(temp1+k),
                  *(temp2+k),              *(temp3+k), *(temp4+k),
                  CURRENT_TIME);
                  }
                  /* free variables */
                  free(temp1);
                  free(temp2);
                  free(temp3);
                  free(temp4);

            }
      }
      free(n);

#endif /* !RP_NODE */
}
```

The UDF is compiled and executed at the end of every time step.

A.2 "Bubble_drag.c"

The UDF "Bubble_drag.c" accounts for the drag force exercised on each rising air-bubble (ref. section 4.1.1). The same drag force is compiled for methane gas in the primary model, only the values of density and surface tension is replaced.

```
#include "udf.h"

DEFINE_DPM_DRAG(particle_drag_term, Re, p)
{
    real drag_term;
    real Eo;
    real Cd;
      real rho_water = 998.2; /* sea water: 1027.0 */
      real surf_tension = 0.07199; /* methane: 0.06180 */
      /* bubble shape - Xia et al. (10) */
      Eo = 9.81*(rho_Ocean - P_RHO(p))*pow(P_DIAM(p), 2) / surf_tension;
      /* drag coefficient */
    Cd = 2.0/3.0*pow((Eo/3),0.5);

      drag_term = 18.0*Re*Cd / 24.0;

    return (drag_term);
}
```

Appendix B

User-defined Functions:

Primary Model

The following sections contain user-defined functions (UDFs) used for the calculations done in Chapter 8. In section B.3, the UDF described in section 4.5.3:Method 1 is presented.

B.1 "Primary_Model.c"

The UDF "Primary_Model.c" accounts for the size distribution (ref. section 4.2.4) and density changes (ref. section 4.2.1) of buoyant methane bubbles. Gas phase is allowed to be dissolved into the surrounding sea water, as explained in section 4.4. The position (x, y and z) of oil droplets or methane bubbles is printed out for various elevations in order to track the plume of interest. Moreover, the residence time of the surfacing oil droplets is first written to a file. The UDF allows for parallel calculations, which implies the use of multiple CPUs. Senior Technical Consultant Love Håkansson at EDRMedeso provided very valuable help in order to achieve this.

```
#include "udf.h"
#include "dpm.h"
#include <math.h>
#include "surf.h"
#include "sg_vof.h"
#include "mem.h"


real depth = 844.0;
real injection_height = 0.33;
real Temp = 5.0;
real Pressure_REF = 101325.0;
real MolarWeight = 16.04; /* methane */
real SurfaceTension = 0.06180;
real C1 = 4.0;
real C2 = 100.0e-6;

static int position_var = 1;
static int position_var1 = 1;
static int position_var2 = 1;
```

```
static int position_var3 = 1;
static int position_var4 = 1;
static int position_var5 = 1;
static int position_var6 = 1;
static int position_var7 = 1;
static int position_var8 = 1;
static int position_var9 = 1;
static int position_var10 = 1;
static int position_var11 = 1;

#if !RP_NODE
FILE *ptr_file;
#endif /* !RP_NODE */

struct P_List_Item_ {
      real p_radius;
      real p_mass;
      real write_x_position;
      real write_y_position;
      real write_z_position;
      struct P_List_Item_ *next;
};

typedef struct P_List_Item_ P_List_Item;

DEFINE_ON_DEMAND(Init_file)
{
#if !RP_NODE
      ptr_file = fopen("surfaceflux.txt", "w");
#endif /* !RP_NODE */
}

DEFINE_ON_DEMAND(Term_file)
{
#if !RP_NODE
      fclose(ptr_file);
#endif /* !RP_NODE */
}

#if !RP_HOST
int doRemove(Particle *p){
      if (P_POS(p)[2] > 0.05) return 1;
      else return 0;
}

void remove_p(Particle *p){
      p->flags = 4;
}

P_List_Item *add_to_list(P_List_Item *head, P_List_Item *tail, Particle
*p){
      if (head->next == NULL){
            head->next = tail;
      }
```

```
      else {
            tail->next = (P_List_Item *)malloc(sizeof(P_List_Item));
            tail = tail->next;
            tail->next = NULL;
      }

      tail->p_radius = sqrt(pow(P_POS(p)[0],2)+pow(P_POS(p)[1],2));
      tail->p_mass = P_N(p)*P_MASS(p);
      tail->write_x_position = P_POS(p)[0];/* x-position */
      tail->write_y_position = P_POS(p)[1];/* y-position */
      tail->write_z_position = P_POS(p)[2];/* z-position */

      return tail;
}

void clear_list(P_List_Item *head){
      P_List_Item *temp, *item;
      item = head;
      while (item != NULL){
            temp = item->next;
            free(item);
            item = temp;
      }
      head = NULL;
      free(temp);
      temp = NULL;
      free(item);
      item = NULL;
}


void send_list(P_List_Item *head, int n){
      real *p_prop1;
      real *p_prop2;
      real *p_prop3;
      real *p_prop4;
      real *p_prop5;
      int i = 0, j;
      P_List_Item *item;
      item = head->next;

      if (n!=0) {
            p_prop1 = (real *) malloc(sizeof(real)*n);
            p_prop2 = (real *) malloc(sizeof(real)*n);
            p_prop3 = (real *) malloc(sizeof(real)*n);
            p_prop4 = (real *) malloc(sizeof(real)*n);
            p_prop5 = (real *) malloc(sizeof(real)*n);

            for (i=0;i<n;i++){
                  *(p_prop1+i) = item->p_radius;
                  *(p_prop2+i) = item->p_mass;
                  *(p_prop3+i) = item->write_x_position;
                  *(p_prop4+i) = item->write_y_position;
                  *(p_prop5+i) = item->write_z_position;
```

```
                item = item->next;
        }

        if (!I_AM_NODE_ZERO_P){
                PRF_CSEND_INT(node_zero, &n, 1, myid);
                PRF_CSEND_REAL(node_zero, p_prop1, n, myid);
                PRF_CSEND_REAL(node_zero, p_prop2, n, myid);
                PRF_CSEND_REAL(node_zero, p_prop3, n, myid);
                PRF_CSEND_REAL(node_zero, p_prop4, n, myid);
                PRF_CSEND_REAL(node_zero, p_prop5, n, myid);

        }
        else {
                PRF_CSEND_INT(node_host, &n, 1, myid);
                PRF_CSEND_REAL(node_host, p_prop1, n, myid);
                PRF_CSEND_REAL(node_host, p_prop2, n, myid);
                PRF_CSEND_REAL(node_host, p_prop3, n, myid);
                PRF_CSEND_REAL(node_host, p_prop4, n, myid);
                PRF_CSEND_REAL(node_host, p_prop5, n, myid);
        }
        free(p_prop1);
        free(p_prop2);
        free(p_prop3);
        free(p_prop4);
        free(p_prop5);
}
else {
        if (!I_AM_NODE_ZERO_P){
                PRF_CSEND_INT(node_zero, &n, 1, myid);
        }
        else {
                PRF_CSEND_INT(node_host, &n, 1, myid);
        }
}

if (I_AM_NODE_ZERO_P){
        compute_node_loop_not_zero(j)
        {
                PRF_CRECV_INT(j, &n, 1, j);
                if (n!=0){
                        p_prop1 = (real *) malloc(sizeof(real)*n);
                        p_prop2 = (real *) malloc(sizeof(real)*n);
                        p_prop3 = (real *) malloc(sizeof(real)*n);
                        p_prop4 = (real *) malloc(sizeof(real)*n);
                        p_prop5 = (real *) malloc(sizeof(real)*n);

                        PRF_CRECV_REAL(j, p_prop1, n, j);
                        PRF_CRECV_REAL(j, p_prop2, n, j);
                        PRF_CRECV_REAL(j, p_prop3, n, j);
                        PRF_CRECV_REAL(j, p_prop4, n, j);
                        PRF_CRECV_REAL(j, p_prop5, n, j);

                        PRF_CSEND_INT(node_host, &n, 1, myid);
```

```
                        PRF_CSEND_REAL(node_host, p_prop1, n, myid);
                        PRF_CSEND_REAL(node_host, p_prop2, n, myid);
                        PRF_CSEND_REAL(node_host, p_prop3, n, myid);
                        PRF_CSEND_REAL(node_host, p_prop4, n, myid);
                        PRF_CSEND_REAL(node_host, p_prop5, n, myid);

                        free(p_prop1);
                        free(p_prop2);
                        free(p_prop3);
                        free(p_prop4);
                        free(p_prop5);
                }
                else {
                        PRF_CSEND_INT(node_host, &n, 1, myid);
                }
        }
    }
}
#endif /* !RP_HOST */

DEFINE_EXECUTE_AT_END(Remove_and_write_to_file) /* executed at the end of
a time step in a transient run */
{
#if !RP_HOST
    Domain *d = Get_Domain(1);
    Injection *i, *i_all;
    Particle *p;
    cell_t c;
    Thread *t;
    real P, gas_density, solubility, parcelmass_pre, MDOT, parcelmass;
    real x[ND_ND];
    real dpm_frac, visc, visc_dpm, rho, d_eq, d_pre, tau, t_k, dt;
    real eps, k;
    real u_s, Re, Pe, k_CH4, n_CH4_sol, J, A, D_CH4;
    real M_H2O = 18.01528;
    real M_NaCl = 58.44277;
    real s = 35.0;
    real red_factor = 0.38;
    real Y_CH4_w = 0;

    P_List_Item *head, *tail, *item, *temp;
    int np_removed=0;

    i_all = Get_dpm_injections();
    head = (P_List_Item *)malloc(sizeof(P_List_Item));
    tail = (P_List_Item *)malloc(sizeof(P_List_Item));
    head->next = NULL;
    tail->next = NULL;

    thread_loop_c(t, d)
    {

    begin_c_loop(c, t)
    {
```

```
            /* calculate volume fraction of DPM-phase  */
            C_CENTROID(x, c, t);
            P = Pressure_REF + 9.81*C_R(c, t)*(depth - x[2]);
            P = MAX(Pressure_REF, P);
            gas_density = P * MolarWeight * 0.001 / (8.314 * (Temp +
273.15));
            C_UDMI(c, t, 0) = C_DPMS_CONCENTRATION(c, t) / gas_density;
            C_UDMI(c, t, 1) = MIN(C_UDMI(c, t, 0), 0.9);
      }
      end_c_loop(c, t)
            /* loop through particles */
            loop(i, i_all){
             loop(p, i->p){


            cell_t c0 = P_CELL(p);
            Thread *t0 = P_CELL_THREAD(p);
            Thread *t0_ocean = THREAD_SUB_THREAD(t0, 1);
            Thread *t0_atmos = THREAD_SUB_THREAD(t0, 0);

            /* gas phase */
            if (P_RHO(p) < 200) {

            parcelmass_pre = P_MASS(p)*P_N(p);
            /* calculation of hydrostatic pressure for each bubble */
            P = Pressure_REF + 9.81*C_R(c0, t0)*(depth - P_POS(p)[2]);
            P = MAX(Pressure_REF, P);
            /* bubble density */
            P_RHO(p) = P * MolarWeight * 0.001 / (8.314 * (Temp + 273.15));

            /* bubble size model, Pan(47) */
            visc_dpm = 1.7849e-05;
            visc = C_MU_L(c0, t0_ocean);
            eps = MAX(C_D(c0, t0), 1.0e-6);
            k = C_K(c0, t0);
            rho = C_R(c0, t0);
            dpm_frac = C_UDMI(c0, t0, 1);
            dt = CURRENT_TIMESTEP;
            /* equilibrium diameter */
            d_eq = C1 * sqrt(dpm_frac) * (pow(SurfaceTension / rho, 0.6) /
            pow(eps, 0.4))*(pow(visc_dpm / visc, 0.25)) + C2;
            d_pre = P_DIAM(p);
            t_k = 6.0*sqrt(visc / (rho*eps));
            if (d_pre > d_eq) /* breakup */
                  {
                        tau = pow(d_pre, 0.66667)*pow(eps, -0.33333333);
                  }

                  else /* coalscence */
                  {
                        tau = d_pre / (0.2*6.0*MAX(1.0e-06,
sqrt(dpm_frac*k)));
                  }
```

```
            tau = MAX(tau, t_k);
            P_DIAM(p) = (d_pre + d_eq*dt / tau) / (1.0 + dt / tau);
            P_DIAM(p) = MAX(P_DIAM(p), 0.0001); /* bubble diameter */
            P_MASS(p) = P_RHO(p) * M_PI * pow(P_DIAM(p), 3.0) / 6.0;

            /*Gas Dissolution*/
            D_CH4 = 1.3*pow(10, (-9));
            u_s = fabs(P_VEL(p)[2] - C_W(c, t)); /* slip velocity */
            Re = (P_DIAM(p) * u_s * rho) / visc;
            Pe = 2.0 * u_s * (P_DIAM(p) / 2.0) / D_CH4;

            /* mass transfer coefficient Zhang & Xu */
            k_CH4 = (1 + pow((1 + Pe), (1.0 / 3.0)) * (1 +
(0.096*(pow(Re, (1.0 / 3.0))) / (1.0 + 7.0 * pow(Re, (-2.0))))))) *
(D_CH4 / P_DIAM(p));

if (3000000.0 < pressure && pressure < 15000000.0)
{
n_CH4_sol0 = 0.002;
}
else if (1600000.0 < pressure && pressure < 3000000.0)
{
n_CH4_sol0 = 0.0005;
}
else
{
n_CH4_sol0 = 0.00008;
}

n_CH4_sol = n_CH4_sol0*exp(-0.319*s / M_NaCl);
/* assume Y_CH4_w = 0 */
J = k_CH4*rho*(n_CH4_sol*(MolarWeight / M_H2O) - Y_CH4_w);
A = M_PI*pow(P_DIAM(p), 2.0);

/* mass transfer with reduction factor */
MDOT = red_factor*A*J*P_N(p);

if ((parcelmass_pre - (CURRENT_TIMESTEP * MDOT)) >= 0.0)
{
parcelmass = parcelmass_pre - (CURRENT_TIMESTEP * MDOT);
}
else
{
parcelmass = 0.0;
}
/* update number of particles in parcel */
P_N(p) = parcelmass / P_MASS(p);

}
/* tracking oil droplets (or bubbles) */
if (P_RHO(p) > 700) {

if (40.0 < P_POS(p)[2] && P_POS(p)[2] < 90.0 && position_var != 0)
      {
```

```
        position_var = 0;
        np_removed;
        tail = add_to_list(head, tail, p);
        }
    if (100.0 < P_POS(p)[2] && P_POS(p)[2] < 150.0 && position_var1 != 0)
        {
        position_var1 = 0;
        np_removed++;
        tail = add_to_list(head, tail, p);
        }
    if (160.0 < P_POS(p)[2] && P_POS(p)[2] < 210.0 && position_var2 != 0)
        {
        position_var2 = 0;
        np_removed++;
        tail = add_to_list(head, tail, p);
        }
    if (220.0 < P_POS(p)[2] && P_POS(p)[2] < 270.0 && position_var3 != 0)
        {
        position_var3 = 0;
        np_removed++;
        tail = add_to_list(head, tail, p);
        }
    if (280.0 < P_POS(p)[2] && P_POS(p)[2] < 330.0 && position_var4 != 0)
        {
        position_var4 = 0;
        np_removed++;
        tail = add_to_list(head, tail, p);
        }
    if (340.0 < P_POS(p)[2] && P_POS(p)[2] < 400.0 && position_var5 != 0)
        {
        position_var5 = 0;
        np_removed++;
        tail = add_to_list(head, tail, p);
        }
    if (400.0 < P_POS(p)[2] && P_POS(p)[2] < 460.0 && position_var6 != 0)
        {
        position_var6 = 0;
        np_removed++;
        tail = add_to_list(head, tail, p);
        }
    if (470.0 < P_POS(p)[2] && P_POS(p)[2] < 530.0 && position_var7 != 0)
        {
        position_var7 = 0;
        np_removed++;
        tail = add_to_list(head, tail, p);
        }
    if (540.0 < P_POS(p)[2] && P_POS(p)[2] < 600.0 && position_var8 != 0)
        {
        position_var8 = 0;
        np_removed++;
        tail = add_to_list(head, tail, p);
        }
    if (610.0 < P_POS(p)[2] && P_POS(p)[2] < 670.0 && position_var9 != 0)
        {
```

```
            position_var9 = 0;
            np_removed++;
            tail = add_to_list(head, tail, p);
            }
if (680.0 < P_POS(p)[2] && P_POS(p)[2] < 740.0 && position_var10 != 0)
            {
            position_var10 = 0;
            np_removed++;
            tail = add_to_list(head, tail, p);
            }
if (750.0 < P_POS(p)[2] && P_POS(p)[2] < 810.0 && position_var11 != 0)
            {
            position_var11 = 0;
            np_removed++;
            tail = add_to_list(head, tail, p);
                                    }

            }
            /* delete surfacing bubbles */
            if (P_RHO(p) < 200)
            {
            if (C_VOF(c0, t0_ocean) < 0.1){
            np_removed++;
            tail = add_to_list(head, tail, p);
            remove_p(p);
                  }
            }
      }
}
send_list(head, np_removed);
clear_list(head);
head = NULL;
tail = NULL;

#endif /* !RP_HOST */

#if !RP_NODE
      int j, *n, k;
      real *temp1, *temp2, *temp3, *temp4, *temp5;
      n = (int *) malloc(sizeof(int));
      compute_node_loop(j)
      {
            PRF_CRECV_INT(node_zero, n, 1, node_zero);
            Message("Node %d: %d particles removed.\n", j, *n);
            if (*n!=0){
                  temp1 = (real *) malloc(sizeof(real)*(*n));
                  temp2 = (real *) malloc(sizeof(real)*(*n));
                  temp3 = (real *) malloc(sizeof(real)*(*n));
                  temp4 = (real *) malloc(sizeof(real)*(*n));
                  temp5 = (real *)malloc(sizeof(real)*(*n));

                  PRF_CRECV_REAL(node_zero, temp1, *n, node_zero);
                  PRF_CRECV_REAL(node_zero, temp2, *n, node_zero);
                  PRF_CRECV_REAL(node_zero, temp3, *n, node_zero);
```

```
                PRF_CRECV_REAL(node_zero, temp4, *n, node_zero);
                PRF_CRECV_REAL(node_zero, temp5, *n, node_zero);

                for (k=0;k<(*n);k++){/* write to file */
                      fprintf(ptr_file, "%g %g %g %g %g\n", *(temp1 + k),
*(temp2 + k), *(temp3 + k), *(temp4 + k), CU*(temp5 + k), RRENT_TIME);
                }
                free(temp1);
                free(temp2);
                free(temp3);
                free(temp4);
                free(temp5);
           }
      }
      free(n);

#endif /* !RP_NODE */
}
```

The UDF is compiled and executed at the end of every time step.

B.2 "Ocean_currents.c" – Method 3

This section contains the UDF used to simulate the effect of ambient ocean currents, described in section 4.5.3:Method 3. The exact current velocities are obtained from figure 19 and ScanIt, which is software for extracting data from scanned graphs.

```
#include "udf.h"
#include "dpm.h"

DEFINE_DPM_BODY_FORCE(crossflow_body_force, p, i)
{
      real Fd, Eo, Cd, Re, Uslip;
      real surf_tension = 0.07199;
      real crossflow_X = -0.0983;/* east/north direction */
      real crossflow_Y = 0.0811; /* north/south direction */
      real accel = 0.0;

      Thread *t = P_CELL_THREAD(p);
      cell_t c = P_CELL(p);

      Eo = 9.81*(C_R(c, t) - P_RHO(p))*pow(P_DIAM(p), 2.0) / surf_tension;
      Cd = 2.0 / 3.0*pow((Eo / 3.0), 0.5);

      if (0.0 < P_POS(p)[2] && P_POS(p)[2] < 73.5)
      {
            crossflow_X = -0.0983;
```

```
            crossflow_Y = 0.0811;

Uslip = sqrt(pow(P_VEL(p)[0] - C_U(c, t) - crossflow_X, 2.0) +
pow(P_VEL(p)[1] - C_U(c, t) - crossflow_Y, 2.0) + pow(P_VEL(p)[2] - C_W(c,
t), 2.0));
Re = C_R(c, t)*P_DIAM(p)*Uslip / C_MU_L(c, t);
Fd = (18.0 / 24.0)*Re*Cd*C_MU_L(c, t) / (P_RHO(p)*P_DIAM(p)*P_DIAM(p));


            if (i == 0) accel = Fd*crossflow_X;
            if (i == 1) accel = Fd*crossflow_Y;

            return accel;
      }

      else if (73.5 < P_POS(p)[2] && P_POS(p)[2] < 123.5)
      {
            crossflow_X = -0.0706;
            crossflow_Y = 0.0874;

Uslip = sqrt(pow(P_VEL(p)[0] - C_U(c, t) - crossflow_X, 2.0) +
pow(P_VEL(p)[1] - C_U(c, t) - crossflow_Y, 2.0) + pow(P_VEL(p)[2] - C_W(c,
t), 2.0));
Re = C_R(c, t)*P_DIAM(p)*Uslip / C_MU_L(c, t);
Fd = (18.0 / 24.0)*Re*Cd*C_MU_L(c, t) / (P_RHO(p)*P_DIAM(p)*P_DIAM(p));

            if (i == 0) accel = Fd*crossflow_X;
            if (i == 1) accel = Fd*crossflow_Y;

            return accel;
      }

      else if (123.5 < P_POS(p)[2] && P_POS(p)[2] < 173.5)
      {
            crossflow_X = -0.0813;
            crossflow_Y = 0.1;

Uslip = sqrt(pow(P_VEL(p)[0] - C_U(c, t) - crossflow_X, 2.0) +
pow(P_VEL(p)[1] - C_U(c, t) - crossflow_Y, 2.0) + pow(P_VEL(p)[2] - C_W(c,
t), 2.0));
Re = C_R(c, t)*P_DIAM(p)*Uslip / C_MU_L(c, t);
Fd = (18.0 / 24.0)*Re*Cd*C_MU_L(c, t) / (P_RHO(p)*P_DIAM(p)*P_DIAM(p));

            if (i == 0) accel = Fd*crossflow_X;
            if (i == 1) accel = Fd*crossflow_Y;

            return accel;
      }

      else if (173.5 < P_POS(p)[2] && P_POS(p)[2] < 223.5)
      {
            crossflow_X = -0.0643;
            crossflow_Y = 0.0937;
```

```
Uslip = sqrt(pow(P_VEL(p)[0] - C_U(c, t) - crossflow_X, 2.0) +
pow(P_VEL(p)[1] - C_U(c, t) - crossflow_Y, 2.0) + pow(P_VEL(p)[2] - C_W(c,
t), 2.0));
Re = C_R(c, t)*P_DIAM(p)*Uslip / C_MU_L(c, t);
Fd = (18.0 / 24.0)*Re*Cd*C_MU_L(c, t) / (P_RHO(p)*P_DIAM(p)*P_DIAM(p));

            if (i == 0) accel = Fd*crossflow_X;
            if (i == 1) accel = Fd*crossflow_Y;

            return accel;
      }

      else if (223.5 < P_POS(p)[2] && P_POS(p)[2] < 273.5)
      {
            crossflow_X = -0.0516;
            crossflow_Y = 0.126;

Uslip = sqrt(pow(P_VEL(p)[0] - C_U(c, t) - crossflow_X, 2.0) +
pow(P_VEL(p)[1] - C_U(c, t) - crossflow_Y, 2.0) + pow(P_VEL(p)[2] - C_W(c,
t), 2.0));
Re = C_R(c, t)*P_DIAM(p)*Uslip / C_MU_L(c, t);
Fd = (18.0 / 24.0)*Re*Cd*C_MU_L(c, t) / (P_RHO(p)*P_DIAM(p)*P_DIAM(p));

            if (i == 0) accel = Fd*crossflow_X;
            if (i == 1) accel = Fd*crossflow_Y;

            return accel;
      }

      else if (273.5 < P_POS(p)[2] && P_POS(p)[2] < 323.5)
      {
            crossflow_X = -0.0239;
            crossflow_Y = 0.119;

Uslip = sqrt(pow(P_VEL(p)[0] - C_U(c, t) - crossflow_X, 2.0) +
pow(P_VEL(p)[1] - C_U(c, t) - crossflow_Y, 2.0) + pow(P_VEL(p)[2] - C_W(c,
t), 2.0));
Re = C_R(c, t)*P_DIAM(p)*Uslip / C_MU_L(c, t);
Fd = (18.0 / 24.0)*Re*Cd*C_MU_L(c, t) / (P_RHO(p)*P_DIAM(p)*P_DIAM(p));

            if (i == 0) accel = Fd*crossflow_X;
            if (i == 1) accel = Fd*crossflow_Y;

            return accel;
      }

      else if (323.5 < P_POS(p)[2] && P_POS(p)[2] < 373.5)
      {
            crossflow_X = 0.0337;
            crossflow_Y = 0.0743;

Uslip = sqrt(pow(P_VEL(p)[0] - C_U(c, t) - crossflow_X, 2.0) +
pow(P_VEL(p)[1] - C_U(c, t) - crossflow_Y, 2.0) + pow(P_VEL(p)[2] - C_W(c,
t), 2.0));
```

```
Re = C_R(c, t)*P_DIAM(p)*Uslip / C_MU_L(c, t);
Fd = (18.0 / 24.0)*Re*Cd*C_MU_L(c, t) / (P_RHO(p)*P_DIAM(p)*P_DIAM(p));

            if (i == 0) accel = Fd*crossflow_X;
            if (i == 1) accel = Fd*crossflow_Y;

            return accel;
      }

      else if (373.5 < P_POS(p)[2] && P_POS(p)[2] < 423.5)
      {
            crossflow_X = 0.0742;
            crossflow_Y = 0.0849;

Uslip = sqrt(pow(P_VEL(p)[0] - C_U(c, t) - crossflow_X, 2.0) +
pow(P_VEL(p)[1] - C_U(c, t) - crossflow_Y, 2.0) + pow(P_VEL(p)[2] - C_W(c,
t), 2.0));
Re = C_R(c, t)*P_DIAM(p)*Uslip / C_MU_L(c, t);
Fd = (18.0 / 24.0)*Re*Cd*C_MU_L(c, t) / (P_RHO(p)*P_DIAM(p)*P_DIAM(p));

            if (i == 0) accel = Fd*crossflow_X;
            if (i == 1) accel = Fd*crossflow_Y;

            return accel;
      }

      else if (423.5 < P_POS(p)[2] && P_POS(p)[2] < 473.5)
      {
            crossflow_X = 0.0806;
            crossflow_Y = 0.0528;

Uslip = sqrt(pow(P_VEL(p)[0] - C_U(c, t) - crossflow_X, 2.0) +
pow(P_VEL(p)[1] - C_U(c, t) - crossflow_Y, 2.0) + pow(P_VEL(p)[2] - C_W(c,
t), 2.0));
Re = C_R(c, t)*P_DIAM(p)*Uslip / C_MU_L(c, t);
Fd = (18.0 / 24.0)*Re*Cd*C_MU_L(c, t) / (P_RHO(p)*P_DIAM(p)*P_DIAM(p));

            if (i == 0) accel = Fd*crossflow_X;
            if (i == 1) accel = Fd*crossflow_Y;

            return accel;
      }

      else if (473.5 < P_POS(p)[2] && P_POS(p)[2] < 523.5)
      {
            crossflow_X = 0.0528;
            crossflow_Y = 0.0442;

Uslip = sqrt(pow(P_VEL(p)[0] - C_U(c, t) - crossflow_X, 2.0) +
pow(P_VEL(p)[1] - C_U(c, t) - crossflow_Y, 2.0) + pow(P_VEL(p)[2] - C_W(c,
t), 2.0));
Re = C_R(c, t)*P_DIAM(p)*Uslip / C_MU_L(c, t);
Fd = (18.0 / 24.0)*Re*Cd*C_MU_L(c, t) / (P_RHO(p)*P_DIAM(p)*P_DIAM(p));
```

```
            if (i == 0) accel = Fd*crossflow_X;
            if (i == 1) accel = Fd*crossflow_Y;

            return accel;
      }

      else if (523.5 < P_POS(p)[2] && P_POS(p)[2] < 573.5)
      {
            crossflow_X = 0.0826;
            crossflow_Y = 0.0484;

Uslip = sqrt(pow(P_VEL(p)[0] - C_U(c, t) - crossflow_X, 2.0) +
pow(P_VEL(p)[1] - C_U(c, t) - crossflow_Y, 2.0) + pow(P_VEL(p)[2] - C_W(c,
t), 2.0));
Re = C_R(c, t)*P_DIAM(p)*Uslip / C_MU_L(c, t);
Fd = (18.0 / 24.0)*Re*Cd*C_MU_L(c, t) / (P_RHO(p)*P_DIAM(p)*P_DIAM(p));

            if (i == 0) accel = Fd*crossflow_X;
            if (i == 1) accel = Fd*crossflow_Y;

            return accel;
      }

      else if (573.5 < P_POS(p)[2] && P_POS(p)[2] < 623.5)
      {
            crossflow_X = 0.0633;
            crossflow_Y = 0.0889;

Uslip = sqrt(pow(P_VEL(p)[0] - C_U(c, t) - crossflow_X, 2.0) +
pow(P_VEL(p)[1] - C_U(c, t) - crossflow_Y, 2.0) + pow(P_VEL(p)[2] - C_W(c,
t), 2.0));
Re = C_R(c, t)*P_DIAM(p)*Uslip / C_MU_L(c, t);
Fd = (18.0 / 24.0)*Re*Cd*C_MU_L(c, t) / (P_RHO(p)*P_DIAM(p)*P_DIAM(p));

            if (i == 0) accel = Fd*crossflow_X;
            if (i == 1) accel = Fd*crossflow_Y;

            return accel;
      }

      else if (623.5 < P_POS(p)[2] && P_POS(p)[2] < 673.5)
      {
            crossflow_X = 0.0611;
            crossflow_Y = 0.0675;

Uslip = sqrt(pow(P_VEL(p)[0] - C_U(c, t) - crossflow_X, 2.0) +
pow(P_VEL(p)[1] - C_U(c, t) - crossflow_Y, 2.0) + pow(P_VEL(p)[2] - C_W(c,
t), 2.0));
Re = C_R(c, t)*P_DIAM(p)*Uslip / C_MU_L(c, t);
Fd = (18.0 / 24.0)*Re*Cd*C_MU_L(c, t) / (P_RHO(p)*P_DIAM(p)*P_DIAM(p));

            if (i == 0) accel = Fd*crossflow_X;
            if (i == 1) accel = Fd*crossflow_Y;
```

```
            return accel;
      }

      else if (673.5 < P_POS(p)[2] && P_POS(p)[2] < 723.5)
      {
            crossflow_X = 0.0632;
            crossflow_Y = 0.0568;

Uslip = sqrt(pow(P_VEL(p)[0] - C_U(c, t) - crossflow_X, 2.0) +
pow(P_VEL(p)[1] - C_U(c, t) - crossflow_Y, 2.0) + pow(P_VEL(p)[2] - C_W(c,
t), 2.0));
Re = C_R(c, t)*P_DIAM(p)*Uslip / C_MU_L(c, t);
Fd = (18.0 / 24.0)*Re*Cd*C_MU_L(c, t) / (P_RHO(p)*P_DIAM(p)*P_DIAM(p));

            if (i == 0) accel = Fd*crossflow_X;
            if (i == 1) accel = Fd*crossflow_Y;

            return accel;
      }

      else if (723.5 < P_POS(p)[2] && P_POS(p)[2] < 773.5)
      {
            crossflow_X = 0.0589;
            crossflow_Y = 0.0674;

Uslip = sqrt(pow(P_VEL(p)[0] - C_U(c, t) - crossflow_X, 2.0) +
pow(P_VEL(p)[1] - C_U(c, t) - crossflow_Y, 2.0) + pow(P_VEL(p)[2] - C_W(c,
t), 2.0));
Re = C_R(c, t)*P_DIAM(p)*Uslip / C_MU_L(c, t);
Fd = (18.0 / 24.0)*Re*Cd*C_MU_L(c, t) / (P_RHO(p)*P_DIAM(p)*P_DIAM(p));

            if (i == 0) accel = Fd*crossflow_X;
            if (i == 1) accel = Fd*crossflow_Y;

            return accel;
      }

      else if (773.5 < P_POS(p)[2] && P_POS(p)[2] < 823.5)
      {
            crossflow_X = 0.0374;
            crossflow_Y = 0.078;

Uslip = sqrt(pow(P_VEL(p)[0] - C_U(c, t) - crossflow_X, 2.0) +
pow(P_VEL(p)[1] - C_U(c, t) - crossflow_Y, 2.0) + pow(P_VEL(p)[2] - C_W(c,
t), 2.0));
Re = C_R(c, t)*P_DIAM(p)*Uslip / C_MU_L(c, t);
Fd = (18.0 / 24.0)*Re*Cd*C_MU_L(c, t) / (P_RHO(p)*P_DIAM(p)*P_DIAM(p));

            if (i == 0) accel = Fd*crossflow_X;
            if (i == 1) accel = Fd*crossflow_Y;

            return accel;
      }
```

```
      else if (823.5 < P_POS(p)[2] && P_POS(p)[2] < 844.0)
      {
            crossflow_X = 0.0246;
            crossflow_Y = 0.0801;

Uslip = sqrt(pow(P_VEL(p)[0] - C_U(c, t) - crossflow_X, 2.0) +
pow(P_VEL(p)[1] - C_U(c, t) - crossflow_Y, 2.0) + pow(P_VEL(p)[2] - C_W(c,
t), 2.0));
Re = C_R(c, t)*P_DIAM(p)*Uslip / C_MU_L(c, t);
Fd = (18.0 / 24.0)*Re*Cd*C_MU_L(c, t) / (P_RHO(p)*P_DIAM(p)*P_DIAM(p));

            if (i == 0) accel = Fd*crossflow_X;
            if (i == 1) accel = Fd*crossflow_Y;

            return accel;
      }

      return accel;
}
```

The UDF is hooked to the body force section in the DPM.

B.3 "Velocity_inlet.c" – Method 1

As described in section 4.5.3, the first attempt of recreating the ocean currents measured in start of each discharge during the DeepSpill experiment, was to define the vertical water boundaries as velocity inlets. The various ocean current velocities is given from two UDFs, in which represents the velocities in x-direction (east/west) and y-direction (north/south). As this method was tested out and quickly discarded, only a few of the measured current velocities are included.

```
#include "udf.h"
#include "mem.h"

/* east/west direction */
DEFINE_PROFILE(inletEastLeft_x_velocity, thread, position)
{
      real x[ND_ND];
      face_t f;
      real z;
      begin_f_loop(f, thread)
      {
            F_CENTROID(x, f, thread);
```

```
            z = x[2];

            if ( 0.0 < z && z < 98.0)
            {
                  F_PROFILE(f, thread, position) = -0.08445;
            }
            if (98.0 < z && z < 197.0)
            {
                  F_PROFILE(f, thread, position) = -0.0728;
            }
            if (197.0 < z && z < 297.0)
            {
                  F_PROFILE(f, thread, position) = -0.03755;
            }
            if (297.0 < z && z < 395.0)
            {
                  F_PROFILE(f, thread, position) = 0.05395;
            }
            if (395.0 < z && z < 496.0)
            {
                  F_PROFILE(f, thread, position) = 0.0677;
            }
            if (496.0 < z && z < 594.0)
            {
                  F_PROFILE(f, thread, position) = 0.07295;
            }
            if (594.0 < z && z < 691.0)
            {
                  F_PROFILE(f, thread, position) = 0.06215;
            }
            if (691.0 < z && z < 785.0)
            {
                  F_PROFILE(f, thread, position) = 0.06645;
            }
            if (785.0 < z && z < 844.0)
            {
                  F_PROFILE(f, thread, position) = 0.0246;
            }
      }
      end_f_loop(f,t)
}


#include "udf.h"
#include "mem.h"

/* north/south direction */
DEFINE_PROFILE(inletNorthBack_y_velocity, thread, position)
{
      real x[ND_ND];
      face_t f;
      real z;

      begin_f_loop(f, thread)
```

```
	{

		F_CENTROID(x, f, thread);
		z = x[2];


		if (0.0 < z && z < 98.0)
		{
			F_PROFILE(f, thread, position) = -0.08425;
		}
		if (98.0 < z && z < 197.0)
		{
			F_PROFILE(f, thread, position) = -0.09685;
		}
		if (197.0 < z && z < 297.0)
		{
			F_PROFILE(f, thread, position) = -0.1225;
		}
		if (297.0 < z && z < 395.0)
		{
			F_PROFILE(f, thread, position) = -0.07496;
		}
		if (395.0 < z && z < 496.0)
		{
			F_PROFILE(f, thread, position) = -0.0485;
		}
		if (496.0 < z && z < 594.0)
		{
			F_PROFILE(f, thread, position) = -0.06865;
		}
		if (594.0 < z && z < 691.0)
		{
			F_PROFILE(f, thread, position) = -0.06215;
		}
		if (691.0 < z && z < 785.0)
		{
			F_PROFILE(f, thread, position) = -0.0727;
		}
		if (785.0 < z && z < 844.0)
		{
			F_PROFILE(f, thread, position) = -0.0801;
		}
	}
	end_f_loop(f,t)

}
```