TECHNISCHE UNIVERSITÄT MÜNCHEN
Lehrstuhl für Computation in Engineering

Coupling of Models and Scales in Computational Fluid Dynamics

Nevena Perović-Rott

Vollständiger Abdruck der von der Ingenieurfakultät Bau Geo Umwelt der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs

genehmigten Dissertation.

Vorsitzender: *Prof. Dr.-Ing.habil. Michael Manhart*

Prüfer der Dissertation:

1. *Prof. Dr.rer.nat. Ernst Rank*
2. *Prof. Dr.rer.nat. Miriam Mehl*
3. *Priv.-Doz. Dr.rer.nat. Ralf-Peter Mundani*

Die Dissertation wurde am 09.09.2020 bei der Technischen Universität München eingereicht und durch die Ingenieurfakultät Bau Geo Umwelt am 13.04.2021 angenommen.

To my parents,
who trusted in me
and supported me unconditionally along this journey,
that only some dare to take!
Thank you!

If you can

"If you can keep your head when all about you
Are losing theirs and blaming it on you,
If you can trust yourself when all men doubt you,
But make allowance for their doubting too;
If you can wait and not be tired by waiting,
Or being lied about, don't deal in lies,
Or being hated, don't give way to hating,
And yet don't look too good, nor talk too wise:

If you can dream–and not make dreams your master;
If you can think–and not make thoughts your aim;
If you can meet with Triumph and Disaster
And treat those two impostors just the same;
If you can bear to hear the truth you've spoken
Twisted by knaves to make a trap for fools,
Or watch the things you gave your life to, broken,
And stoop and build 'em up with worn-out tools:

If you can make one heap of all your winnings
And risk it on one turn of pitch-and-toss,
And lose, and start again at your beginnings
And never breathe a word about your loss;
If you can force your heart and nerve and sinew
To serve your turn long after they are gone,
And so hold on when there is nothing in you
Except the Will which says to them: 'Hold on!'

If you can talk with crowds and keep your virtue,
Or walk with Kings–nor lose the common touch,
If neither foes nor loving friends can hurt you,
If all men count with you, but none too much;
If you can fill the unforgiving minute
With sixty seconds' worth of distance run,
Yours is the Earth and everything that's in it,
And which is more - you'll be a Man, my son!"

Joseph Rudyard Kipling (1865–1936)

Mojim roditeljima,
koji su mi ukazali veliko povjerenje
i pružili bezrezervnu podršku na ovom putu,
kojim se redje ide!
Hvala vam!

Ako možeš

"Ako možeš da sačuvaš prisebnu glavu,
kada svi oko tebe gube svoju, i okrivljuju te za to,
Ako možeš da veruješ sebi,
kada svi u tebe sumnjaju i sam dodaješ njihovim sumnjama,
Ako možeš da čekaš - a da ti ne dosadi čekanje,
ili ako si prevaren - da sam ne varaš,
ili ako si omrznut - da sam ne mrziš,
a da pritom ne izgledaš predobar ili premudar,

Ako možeš da sanjariš, a da snovi ne ovladaju tobom,
Ako možeš da maštaš, a da ti maštanje ne bude cilj,
Ako možeš da se suočiš sa uspehom i neuspehom
i smatraš te dve varke kao da su potpuno iste,
Ako možeš da podneseš da istinu koju si rekao
izvrnu nitkovi kako bi od nje napravili zamku za budale,
ili da posmatraš propast onoga čemu si posvetio sav život,
i da pogrbljen, sa dotrajalim alatom opet novo stvaraš,

Ako možeš da prisiliš svoje srce, nerve i tetive,
da te služe dugo, iako si ih nemilice trošio,
i da izdržiš kada nema ničega više u tebi sem volje koja ti dovikuje - Istraj!

Ako možeš da razgovaraš sa nižima od sebe
i ne istakneš svoju superiornost,
ili da u društvu sa višima od sebe sačuvaš svoje dostojanstvo,
Ako ni prijatelj, ni neprijatelj ne mogu da te uvrede,
Ako te svi cene, ali ne previše,
Ako možeš da ispuniš jedan minut sadržajem koji traje šezdeset sekundi,
tvoja je zemlja i sve što je na njoj,
i iznad svega
bićeš Čovek, sine moj!"

Joseph Rudyard Kipling (1865–1936)
Preveo: Ivo Andrić (1892–1975)

Zusammenfassung

Durch die schnelle Entwicklung bei Hochleistungs-Computerinfrastrukturen in den letzten zwei Jahrzehnten und mithilfe von Software, die von solch komplexen Systemen unterstützt werden, kann man heutzutage eine Vielzahl von früher kaum lösbaren Problemen aus dem Ingenieurwesen effizient berechnen. Grundsätzlich ist es sogar möglich, die Ergebnisse in Teilen des Berechnungsgebiets zu visualisieren, während noch die Gesamtsimulation durchgeführt wird. Dies war die besondere Motivation für die Entwicklung des Hochleistungssimulationsframeworks MPFluid, das in einem früheren Forschungsprojekt entwickelt und auf mehreren Hochleistungsarchitekturen getestet wurde, unter anderem am Leibniz-Rechenzentrums. Die vorliegende Arbeit beschäftigt sich mit der Erweiterung dieses HPC-Frameworks um eine völlig neue Simulationspipeline, die mehrphasige reale Strömungsphänomene analysiert. Dies beinhaltet die Entwicklung von drei separaten numerischen Modellen: 2D Flachwassermodell (2D-SWE), 3D Flachwassermodell (3D-SWE) und 3D Navier-Stokes Modell für inkompressible, nichtviskose Newtonische Strömungen (3D-NSE) sowie von drei gekoppelten numerischen Modelle: Darcy-Strömung (DG) mit 3D-NSE, 2D-SWE mit 3D-SWE und 2D-SWE mit 3D-NSE. Es ist das Ziel dieser Kopplungen, von den geringeren Kosten der einfacheren Modelle und der höheren Genauigkeit der komplexeren Modelle zu profitieren. Die Erzeugung des numerischen Modells basiert auf einem hierarchischen Oktalbaum-Ansatz, der zu einem blockstrukturierten, kartesischen Netz führt. Um den Speicherbedarf zu reduzieren, wird zusätzlich eine kollokierte Netzanordnung verwendet. In einigen Fällen kann dies zu einer Entkopplung zwischen den Druck- und Geschwindigkeitsfeldern führen. Zu Lösung dieses Problems wird, angelehnt an die Rhie-Chow-Interpolation, eine Rekonstruktion der Flüsse an den Zelloberflächen durchgeführt. Dies ist bei der Advektion der Grenzfläche zwischen zwei Flüssigkeiten besonders wichtig. Die Navier-Stokes Impuls- und Kontinuitätsgleichungen werden mit der Chorin-Projektionsmethode gelöst. Im ersten Schritt wird dazu ein vorläufiges Geschwindigkeitsfeld berechnet, gefolgt von der Erzeugung einer Poisson-Druckgleichung mit variablen Koeffizienten, die durch eine einfache mathematische Reformulierung erhalten werden kann. Die Lösung dieser Poisson-Gleichung wird nun genutzt, um das vorläufige Geschwindigkeitsfeld zu korrigieren. Um die Poisson-Druckgleichung effizient lösen zu können, wird ein Multigrid-artiger Löser basierend auf der vorhandenen Baumdatenstruktur implementiert. Dabei kommen zwei verschiedene Ansätze zum Einsatz: ein einfacher V-Zyklus und ein vollständiges Multigrid-Verfahren, dessen Leistung in zuvor durchgeführten Arbeiten ausgiebig getestet wurde. Mithilfe der Poisson-Gleichung mit variablen Koeffizienten ist die Behandlung von Mehrphasenströmungen mit großen Gradienten in den Material- und Transporteigenschaften möglich. Die Grenzfläche zwischen zwei verschiedenen Flüssigkeiten wird mit einer konservativen Level-Set-Methode erfasst. Außerdem wird eine detaillierte Analyse verschiedener Capturing-Methoden durchgeführt. Mit jeweils unterschiedlichen Distanzfunktionen wird versucht, eine konstante Weite des Grenzflächenprofils zwischen zwei Flüssigkeiten zu erreichen. Der wesentliche Beitrag dieser Arbeit liegt in der Möglichkeit, mehrere Methoden, die unterschiedliche Komplexität aufweisen und die auf verschiedenen Längenskalen angewendet werden können, effizient zu kombinieren. Einfachere 2D-Modelle können die Lösungszeit erheblich verkürzen, wenn sichergestellt werden kann, dass die Ergebnisse nicht unzulässig beeinträchtigt werden. Teure 3D-Modelle können in beschränkten, geometrisch sehr komplexen Regionen genutzt werden, was zu genauen Ergebnissen führt und einen besseren Einblick in die wesentlichen Phänomene bietet. Solche komplexen Modelle werden vorwiegend im Mikromaßstab angewendet, während Modelle wie

das Darcy-Gesetz nur für die Analysen im Makromaßstab geeignet sind. Alle unabhängigen und gekoppelten Modelle wurden anhand von Benchmark-Problemen validiert und/oder verifiziert, wie beispielsweise an reinen Advektionstests, den Transporte des Grenzflächenprofils unter quellenfreien Geschwindigkeitsvektorfeldern, der Umströmung eines prismatischen Hindernisses, der Überflutung einer idealisierten Stadt, dem kreisförmigen Dammbruch und am Problem des Dammbruchs von Kleefsman. Für alle Validierungsbeispiele konnte sehr gute Übereinstimmung mit Referenzergebnissen erzielt werden. Schließlich wurden mehrere realitätsnahe Szenarien ausgewählt, um die Flexibilität, Effizienz und Leistungsfähigkeit des gesamten Frameworks zu veranschaulichen.

Abstract

With the rapid development of high-performance computing infrastructures in the last two decades and computer applications that are supported by such complex systems, a wide variety of engineering problems, which were once considered to be unsolvable in a reasonable period of time, can nowadays be easily run using HPC infrastructure. Theoretically, these processes that are performed in parallel can enable the visualisation of the results of one particular region, while performing a simulation in the entire domain of interest. This was the particular motivation for the development of the high-performance simulation framework MPFluid, which has been developed in a preceding research project and tested on several supercomputing architectures, including one maintained at Leibniz Supercomputing Centre. The scope of this thesis is the extension of this HPC framework with an entirely new simulation pipeline, which analyses multi-phase real-world fluid flow phenomena. This includes the development of three separate numerical models: the 2D Shallow Water (2D–SWE), 3D Shallow Water (3D–SWE) and 3D Navier–Stokes (3D–NSE) models for incompressible, inviscid Newtonian flows, and three coupled numerical models: Darcy's Law (DL) with 3D–NSE, 2D–SWE with 3D–SWE, and 2D–SWE with 3D–NSE, whose goal is to benefit from the low costs of the simple models and the high accuracy of the more complex ones.

Generation of the numerical domain is based on a hierarchical octree approach that leads to a block-structured, orthogonal Cartesian mesh. Additionally, to achieve a better memory footprint, a collocated mesh arrangement is adopted. In some cases this may cause a decoupling between the pressure and velocity fields, thus a Rhie–Chow-like reconstruction of the fluxes at the cell faces is performed to solve this problem. The Navier–Stokes momentum and continuity equations are solved using Chorin's projection method, which solves for the intermediate velocity vector field in the first step and, by simple mathematical manipulation, a Poisson pressure equation with variable coefficients is generated and solved once at each time-step. An efficient multi-grid-like solver is implemented exploiting the existing tree data structure, allowing two different multi-grid-like approaches: namely, a simple V-cycle and full-multigrid, whose performance was thoroughly tested in the work undertaken prior to this thesis.

With the variable-coefficient Poisson equation implemented inside the pressure system, the treatment of multi-phase flows with large gradients in the material and transport properties became possible. The interface between two different fluids is resolved using the conservative level-set method, one of the most promising interface-capturing methods for this group of problems. Besides that, a detailed analysis is performed, taking into consideration different capturing methods and their respective re-distance functions, responsible for the constant thickness of the interface profile between two fluids.

The main benefit of this work is the capability of efficiently combining several methods which have different complexities and which can be applied at different length scales. Hence, simple 2D models can significantly reduce the time to solution in those areas where the results would not be compromised by the simplicity of the model applied. Expensive 3D models can be exploited in small, geometrically very complex regions, resulting in accurate results that offer a deeper insight into the phenomenon of interest. Such complex models are almost exclusively performed on the micro scale, whereas models such as Darcy's Law are only suitable for

macro-scale analysis. All independent and coupled models have been validated and/or verified using standard benchmark cases, such as pure advection tests, solenoidal velocity field interface transport, flow around a prismatic obstacle, circular dam break, flooding of an idealised city and Kleefsman's dam-breaking experiment, resulting in great overall concurrence. Finally, several real-world scenarios have been chosen carefully in order to illustrate the flexibility, efficiency and capability of the complete built environment.

Acknowledgments

Starting on this journey, little did I know that it was not going to be just another challenge to solve. Six years later, it appears to be a life-changing event that has transformed me in many personal and professional aspects. At this point I would like to express my deepest gratitude to several people who supported me along the way, and helped me conceptualise the ideas and create this work.

At the very beginning I owe special gratitude to my parents, who not only have supported me for the last six years, but also created such an atmosphere from my early childhood in which education and learning new things played a very important role. And although this path may have been difficult and full of expectations imposed by others and enforced by oneself, it gave me a chance to do the right things, explore new horizons and ask unconventional questions, in order to discover my own purpose and sense of fulfilment. I had great pleasure sharing my profession with them and working in the same environment, gaining new insights and knowledge each and every day. While following my current path, I have received unconditional love and support, which has given me the moral strength to cope with the new challenges coming my way. Needless to say, they supported me emotionally and financially through the tough times, allowing me to stay focused on doing the research and writing my thesis. Furthermore, my brother, sister and husband understood the importance of this whole journey for me and encouraged me to persist, while creating a comfortable family base to which I could return, no matter what.

My supervisor **Prof. Dr.rer.nat. Ernst Rank** and mentor **PD Dr.rer.nat. habil. Ralf-Peter Mundani** earned my special appreciation for the professional support and guidance offered during the entire process. I was granted the freedom to explore my own idea, shape my research and follow my interests, while learning valuable skills that I can use on a daily basis in my future life. Besides that, they were involved in intensive, frequent scientific discussions since I formally left the department, in order to help me finalise my thesis in the way it is published now. The patience with which they approached me was exactly what I needed most in order to create a clear vision of the final goal that should be achieved.
Furthermore, I'm very thankful to **Prof. Dr. Miriam Mehl** for accepting the role of the formal evaluator and examiner, while giving me valuable comments that finally led to the improved version of my thesis.
I would also like to thank **Prof. Dr.-Ing. habil. Michael Manhart** for acting as the committee chairman during the defence of my dissertation, as well as for several project-based scientific discussions, which brought a wider understanding of the topics of interest.

Having chance to spend three months abroad at the Heat and Mass Transfer Technological Centre (CTTC) at Universitat Politècnica de Catalunya in Barcelona helped me advance my research at the current level and understand yet another scientific perspective, thus I would like to thank **Prof. Dr. Assensi Oliva** for granting me that opportunity, as well as my CTTC colleagues **Néstor Balcazar**, **Jordi Muela** and **Oscar Antepara** for their valuable scientific ideas. All my other teammates accepted me immediately as a family member, introducing me to their daily habits and culture, which made me think that I could spend the rest of my life there.

Needless to say, I was extremely happy to have my colleagues **Jérôme Frisch**, **Christoph Ertl** and **Bobby Minola Ginting** to share the office with, to learn a lot from them and discuss important research topics. They also made me feel as if I have a second home, respecting my culture and experience that I brought, and sharing their life values with me. I highly appreciate the scientific inputs that shaped my overall thinking, and enjoyed our nerd conversations in the late-afternoon coffee breaks at the 'Bügelbrett'.

Last but not least, during the work on my doctoral thesis at the Chair for Computation in Engineering I was involved in two large research projects:

1. High-performance interactive flood simulations (iFlood), financed and coordinated by the International Graduate School of Science and Engineering (IGSSE), and

2. High Performance Computing of Coupling 2D and 3D Numerical Modelling of Flood Propagation, as a part of the cooperation between Munich Technical University, Germany and King Abdullah University of Science and Technology (KAUST), Saudi Arabia.

Hereby, I would like to thank IGSSE for its financial and scientific support, as well as my colleague **Hao Zeng** for the joint work on the iFlood project. I'm also thankful to **Prof. Dr.-Ing. Markus Disse** from the Chair of Hydrology and River Basin Management (TUM), who allowed internal usage of the data gathered for the FloodEvac Project in Kulmbach area, Germany. A formal permit was also acquired from the Bavarian State Office for Survey and Geoinformation ('Bayerische Vermessungsverwaltung').

Munich, 24.06.2020.

Nevena Perović-Rott
nevena.perovic@tum.de

Contents

Table of Symbols

CFD	Computational Fluid Dynamics
LS	Level Set
CLS	Conservative Level Set
NSE	Navier-Stokes Equations
VOF	Volume of Fluid
VOFCLS	Volume of Fluid and Conservative Level Set coupled
VOSET	Volume of Fluid and Standard Level Set coupled
2DSW	2-dimensional Shallow Water
3DSW	3-dimensional Shallow Water
LSM	Least Square Method
SLIC	Simple Line Interface Calculation
PLIC	Piecewise Linear Interface Calculation
WLIC	Weighted Linear Interface Calculation
THINC	Tangent of Hyperbola Interface Capturing
RK1, RK2, RK3	Runge-Kuta 1^{st}, 2^{nd} and 3^{rd} order temporal discretization scheme
MAC	Marker-And-Cell interface capturing method
FTM	Front-Tracking Method
FCM	Front-Capturing Method

Chapter 1

Introduction

1.1 CFD advance through history – Physical and numerical modeling

Fluid mechanics, as a branch of physics, is presumed to be one of only a few disciplines that has gone through continuous development all the way from ancient times up to the present day. It collects knowledge about the mechanics of fluid flows and explains a wide variety of behaviours of fluid entities loaded with internal stresses and exposed to external forces. Ancient peoples, developing their first civilisations around large water sources, attempted to decipher the laws nature obeys and to establish early flood protection measures in order to settle down, protect their homes, crops and human lives from unexpected incidences of flooding. Besides flood protection, the aerodynamics of arrows and spears, the buoyancy of floating objects, and irrigation and drainage were the focus of development, leading to the first known collection of hydraulic principles "On Floating Bodies" written by Archimedes in 250 BC. A hundred years later, the first hydraulic pump was invented by Ptolemy allowing the simple transportation of water from one place to another. The first breakthrough in the new era was achieved by the prominent Roman civil engineer, Sextus Julius Frontinus, who established several basic laws in the field of water dynamics that were applied in the operation of Rome's famous aqueducts. The first scientific experimental methodologies were developed on the Arabian Peninsula in the ninth century AD, and soon afterwards these were combined with mathematical principles, describing the behaviour of a fluid at rest (fluid statics) and laying down the fundamentals for fluid dynamics as it is known today. Further rapid advancement in this field happened from 15th to 18th centuries involving important names such as: Leonardo da Vinci, Evangelista Torricelli, Blaise Pascal, Isaac Newton and Daniel Bernoulli. In this period, the importance of physical experiments was emphasised, several practical devices for measurement were developed, including the barometer, and finally the basic principles of hydrodynamics were precisely clarified so that scientists could start looking at some more complex concepts (e.g. fluid viscosity and turbulence effects). All these discoveries gave rise to the modern mathematically justified theory of fluid dynamics, which manifested itself through the work of several famous mathematicians and engineers, such as: Osborne Reynolds, Andrey Kolmogorov, Joseph Louis Lagrange, Pierre-Simon Laplace, Siméon Denis Poisson, Claude-Louis Navier and George Gabriel Stokes.
The mathematical formulations introduced could be applied to a wide variety of engineering

problems, yet they were almost always very complex to solve, thus adopting particular assumptions, various simpler models were brought in. These allowed some phenomena to be solved much faster, sacrificing an accuracy of the solution to the extent that could be justified. Because of this, today we can distinguish several different forms, such as: potential flow, also called ideal flow, as the simplest form derived from the Navier–Stokes model; then Stokes flow, where viscous forces are dominant in comparison to the inertial forces; Euler flow, which can be considered as a Navier–Stokes model with zero viscosity and zero thermal conductivity, etc. In order to better classify all these diverse sub-models, several numerical quantities, such as the Reynolds, Froude, Weber and Mach numbers were established, describing ratios of pairs of particular terms of importance. These numbers are especially important if only non-dimensional numerical analysis and comparison to scaled physical models are performed. The Reynolds number (R_e) represents the ratio of the inertial and viscous forces, establishing an entirely new classification label: laminar, transient and turbulent flow types. The Froude number (F_r) was determined to be the ratio between inertial and gravitational forces, further divided into super-critical and sub-critical flows, essentially introducing the influence of the geometrical characteristics of the setup calculated. The Weber number (W_e) describes the influence of inertial forces compared to surface tension, particularly useful in multi-phase flows, where water droplets or the formation of thin films plays a significant role. Finally, the Mach number (M) represents the proportion of the fluid velocity to the speed of sound through the fluid medium, grouping the flows into the subsonic, supersonic and hypersonic ones. This number is usually used to determine whether the flow can be treated as an incompressible and isothermal flow (M<0.3), or whether the compressibility effects should be taken into consideration.
Despite those significant simplifications, many of the models formulated back in the 18th and 19th centuries were utilised practically only in the late 20th century, when the development of computers and improvements in computing power that continue up until the present day, enabled researchers to solve complex problems simultaneously. This branch of fluid dynamics, that exploits computational resources in order to solve fluid motion phenomena, is called Computational Fluid Dynamics (CFD), and with the development of high-performance computing systems, is becoming more and more important in scientific advancement.

This is particularly visible in the analysis of complex turbulent flows with an irregular, occasional fluid motion accompanied by strong perturbations in the velocity and pressure fields. A large effort has been made to describe this often called chaotic behaviour since the early 19th century, resulting in an extensive experimental work, for instance A. E. Bryson [2], G. Guderley and H. Yoshihara [53] and K.-S. Choi [75] among all others, and a general classification into six different models, according to J. H. Ferziger and M. Perić [65] – starting from the most simple ones correlated to the Reynolds, Nusselt or Prandtl numbers, through models with one or two additional equations, also called closures, up to models that solve large-scale phenomena and approximate small-scale effects (LES). Finally, the most complex model (DNS) assumes the solution of the entire spectrum of scales present in the flow. Due to the difficulty of creating such random behaviour in an experimental environment, the validation of such models is often limited; instead direct numerical simulation (DNS) is used in the verification process of all the other, less complex models. Due to already broad objectives of this work, turbulence effects are not considered in this thesis; however, an extensive research with regard to different models coupled and turbulence influence has been done by F. Mintgen and M. Manhart [46], H. Zeng et al. [57] and H. Zeng et al. [58] within an open source OpenFOAM software.

At the beginning of CFD development, only flows that involve purely one fluid medium were analysed and compared with experimental results, in order to eliminate, as much as possible, all sorts of errors originating from the approximation of mathematical concepts, discretisation methods that convert differential equations into a system of algebraic equations, as well as rounding errors and errors emerging from the iterative methods used. It should not be forgotten that the required accuracy can only be obtained within systems that describe one phenomenon accurately. Nonetheless many incidents even nowadays are not being formulated precisely, thus a large amount of the collected data must be assessed and justified with great care and a lot of scientific awareness. Along with single-phase flows, both compressible and incompressible ones, that are well elaborated in the wide range of standard literature (J. H. Ferziger and M. Perić [65], and C. Hirsch [22], for instance), a vast variety of engineering applications has driven the development of multi-phase flow simulations, often involving three or more fluids of different consistencies, that may even react chemically with each other, additionally complicating the already complex numerical representation. These reactive flows deserve to be discussed separately and will not be dealt with within this work. Further systematisation is done with regard to inviscid fluid flows.
The challenging task of combining two or more inviscid fluids is discontinuity in the transport properties (e.g. density and viscosity) at the interface boundary, as well as the difficulty of accurately determining the position of the interface itself, where the effects of surface tension must be accounted for. Should this not be possible, non-physical velocities and pressures will appear at the interface, influencing the numerical solution to such an extent that the results may even not match the same physical problem. To address those issues, many different methods have been developed, with the objective of tracking the interface advection as accurately as possible. The Lagrangian methods belong to the first group, within which the mesh follows the motion of the fluid and the interface itself (H. H. Hu et al. [56], A. A. Johnson and T. E. Tezduyar [1]). The second group consists of front-tracking methods, using a fixed grid to compute both fluids, and an additional grid of low resolution is used to capture the interface, projecting markers onto the fixed grid and correcting the results accordingly (J. Glimm et al. [64], G. Tryggvason et al. [54]). The third group, and probably the most accurate one, involves using two separate conforming meshes for two interacting fluids and a simultaneous exchange of the parameters at the connecting interface (S. Takagi et al. [128]). The last, and still the most popular, group encompasses Eulerian front-capturing methods that make use of regular stationary grids, where the flow characteristics are calculated after advection of the interface is performed and the effects of surface tension included. Among these methods, the Marker-and-Cell (MAC), Volume-of-fluid (VOF) and Level-Set(LS) methods are the ones most frequently utilised. Although front-tracking methods (FTM) aim for better accuracy in comparison to front-capturing methods (FCM), the former have reported quite a few difficulties when simulating the separation and merging of two or more fluid entities within the domain. On the other hand, front-capturing methods are very sensitive to mass conservation and, in the case of large pressure or density gradients, may require additional attention concerning all the various stability criteria. Among all the FC methods, two large groups are evident: 1) sharp interface methods that conserve mass quite accurately; nonetheless, due to the discrete discontinuous representation, the curvature of the interface calculated is prone to error; and 2) smooth interface methods that generate a smooth transition for all the transport properties over the interface, allowing precise computation of the interface curvature; however, as published in [102], this is prone to mass conservation problems. In recent years, a few different reconstruction

techniques within the VOF methods have been applied, aimed at better accuracy and less spurious currents, while conservative level-set methods have provided better mass conservation. These methods are further developed in this work, thus all further details will be given in the following sections as depicted in Sec. 1.3. To the author's best knowledge, the most successful implementations in the VOF methodology have been published by C. W. Hirt and B. D. Nichols [25], L. Jofre et al. [79], M. van S. Annaland et al. [95], M. Malik et al. [89], A. Pathak and M. Raessi [7], W. Aniszewski et al. [146] and M. Sussman et al. [93]. A good overview of various VOF methods is also given by O. Ubbink and R. I. Issa [107] and D. Greaves [29], while within the LS methodology some valuable ideas and concepts are published by O. Desjardins et al. [106], S. Osher and J. A. Sethian [125], E. Olsson and G. Kreiss [35], M. Sussman et al. [92], N. Balcázar et al. [101] and Antepara [10]. Also worthy of mention is an approach of coupling two different interface-capturing methods in order to benefit from their particular advantages, eliminating the major, known disadvantages on both sides. One such example is the coupling of the Level-Set and Ghost-Fluid methods reported by B. Lalanne et al. [12], followed by Z. Wang et al. [156] and N. Balcázar et al. [102], where the volume-of-fluid and level-set (LS/CLS) methods have been coupled.

1.2 Objectives of this work

The goal of this work is the development of high-performance computer software, capable of simulating and analysing large urban flooding events. In order to tackle this complex problem in an efficient way, the following sub-tasks should be implemented:

- Implementation and validation of the 2D Shallow Water model (2D-SWE) that can be used for the rough assessment of the water depths and hazards in large inundation areas, without a complex terrain setting.
- Implementation and testing of the 3D Shallow Water model (3D-SWE) that uses the 2D-SWE as a foundation, and offers some enriched information about the pressure and velocity distribution over the z-axis. This additional data is unavailable in 2D-SWE, due to theoretical simplifications and assumptions. The terrain description is as complex as in the case of the 2D-SWE model.
- Development of the 3D Navier–Stokes model (3D-NSE) for multi-phase flows, capable of dealing with large pressure and density gradients. The interface reconstruction can be dealt with in a number of ways already known in the scientific community. The complexity of the geometrical representation within this model is unlimited.
- In order to tackle the efficiency of the simulation process, coupling strategies between the 2D-SWE and 3D-SWE models should be developed, allowing the vast majority of the domain to be dealt with using the cheaper 2D model; thus only a small part of the domain should undergo expensive 3D computation and reconstruction procedures.
- For the same reason that is stated in the previous task, coupling between the 2D-SWE and 3D-NSE models should be established, allowing for fast, effective simulation in the parts of the domain where a simple terrain description is found, and a resolution of the complex flow phenomena in regions where the 3D effects are crucial for the explanation of different flow behaviours. This includes very complex geometry descriptions, which are not possible to deal with within the coupled 2D–SWE/3D–SWE approach (e.g. flow under a bridge).

The entire work is split into chapters and sections as given in section 1.3, giving a necessary theoretical background and implementation details, followed by the validation/verification setups and application scenarios.

1.3 Outline of the thesis

Chapter 1 summarises the history of the development of computational fluid dynamics (CFD), starting from the early steps in ancient times, continuing through the rich Renaissance period that offered significant innovation in the field of fluid mechanics, finally focusing on the late 19th and 20th centuries, within which the breakthrough happened that allows us today to use numerical simulations instead of physical models. This chapter further includes the objectives of the thesis and the structure of the work.

In **Chapter 2**, a brief overview of geometric modelling techniques is given, focusing on the difference between surface and volumetric representation of geometry. The entire high-performance software implementation is based on the generation of a hierarchical space-tree structure and its usage in numerical iterative procedures (e.g. geometrical multigrid). For that reason, more details about the standard application routines, some specificities in this particular application branch, as well as the state-of-art work in this field are given. In Sec. 2.2, the three different geometrical input types used in this work are illustrated, followed by their conversion into the format required. The algorithmic approach in each section is included.

The focus of **Chapter 3** is the mathematical background of the models implemented: the 3D Navier–Stokes model, the 2D & 3D Shallow Water models, the Darcy hydraulic model for flow through porous media, and the three most frequently used models for free-surface tracking. Within every section, the advantages and disadvantages of each approach are given, as well as a justification for the chosen strategy in each particular case. As the conservative level-set approach is finally chosen as the interface-capturing method, Sec. 3.6.3 contains the set of different convection tests and verification examples, in order to prove the accuracy and efficiency of the method within this HPC package. Further considerations about mass, momentum and energy conservation can be also found in Sec. 3.1 of this chapter.

Keeping the mathematical formulation presented in Chapter 3 to hand, **Chapter 4** addresses all the numerical procedures used throughout the mesh discretisation process, calculation of the gradients of the transport properties, which appears to play a crucial part in the interface reconstruction process, followed by the treatment of the discontinuities in multi-phase flows and the resolution of the Riemann problem. Secs. 4.1.6 and 4.1.7 give an overview of the high-order temporal and spatial schemes used within this work and highlight the importance of the flux and gradient limiter functions, combined with the Total-Variation-Diminishing temporal settings that are again critical for the stability of the Runge–Kutta schemes.

In **Chapter 5**, a detailed summary of the high-performance computing framework MPFluid, which was used extensively and enhanced with a set of different multi-phase-flow models, is given. This includes a description of the data structure required, combined with the parallelisation strategies and iterative methods used. As both single- and multi-phase flows are integral components of this work, the underlying variable- and constant-coefficient Poisson equations used in the solution are explained thoroughly here in Secs. 5.2 and 5.3.

Following the objectives of this thesis depicted in Sec. 1.2, in **Chapter 6** the obtained results are presented, focusing primarily on the validation/verification process, followed by

real-world application scenarios. Sec. 6.1 is reserved for single-phase flow through porous media, Secs. 6.2 and 6.3 deal with the 2D and 3D shallow water models, while Sec. 6.4 depicts the 3D Navier–Stokes model combined with the level-set interface-capturing method. The last two chapters contain the coupled models' results for 2D-SWE & 3D-SWE and 2D-SWE & 3D-NSE, respectively.

Chapter 7 contains a short conclusion on what has been done within the scope of this work, suggesting several further topics that are considered to be a logical continuation in the development of both multi-dimensional model coupling and numerical simulations of multi-phase flows.

Chapter 2

Geometrical Modelling

2.1 Geometric models – Theoretical foundations

Development of the algorithms necessary for the mathematical description of geometric shapes is the most general definition of geometric modelling. This branch of applied mathematics is frequently used in the field of computer graphics and the gaming industry; nevertheless, such algorithms also play a significant role in the definition of complex geometries for fluid flows, which is an important part of any simulation process. Depending on the objectives of the simulation, geometrical objects, such as the complex terrain representation depicted in Fig. 2.9, a floating object (see Fig. 2.15) around which the river flow and resistance is calculated, or even a bio-mechanical simulation of a vertebra with pedicle screws, published by M. Elhaddad et al. [86], can be represented using either direct or indirect schemes. The latter focuses on the exact representation of the surface of an object, while the volume of that entity is only indirectly defined. This method is known as the boundary representation (B-rep or BREP method), within which a complex set of operations, such as extrusion, sweeping and drafting, is enabled by simple manipulation of shapes' essential elements: vertices, edges and faces (one such model is shown in Fig. 2.2). They are well incorporated into the Computer Aided Design (CAD) sector, supporting all kinds of different visual representations (Wireframe, Edge-face-rendering or Ray-tracing, for instance). The former group puts an emphasis on exact volume definition, forming another group of volume-based models (one such example is given in Fig. 2.3), within which the surface of that object is dealt with indirectly. Such models offer obvious advantages, if the volume-based simulation is to be done with access required to various parameters throughout the volume. A numerical simulation in the field of computational fluid dynamics, where the values of pressure, velocity, vorticity etc. must be inspected throughout the entire domain, is a very good example of such requirements. However, precise definition of the surface of volume-based models also plays an important role, especially when it comes to the simulation of multi-phase flows, where the deviations in the definition of the interface between two fluids leads to spurious currents and irregularity in the volume-based parameters. For that reason, it is important to establish a good connection between the surface and volume representation of the domain, making sure that the communication, exchange and update between both of them is done properly. The generation of numerical domains, from either surface- or volume-based geometrical representation, can be done in two distinct ways, resulting in either a non-hierarchical or a hierarchical model. Non-hierarchical models

are based on the equidistant discretisation of the entire domain, leading to the complexity $\mathcal{O}(n^3)$ in case of a three-dimensional model, where n represents the amount of voxels in one coordinate direction. Hierarchical models, on the other hand, employ a recursive sub-division of a domain that surrounds the geometry of interest into smaller primitives (triangles and quadrilaterals in 2D, and tetrahedrals and cuboids in 3D space), until the required precision is achieved. The internal mapping of the smaller primitives to their larger, parental primitives, concerning the memory requirements, is done using a hierarchical tree structure.

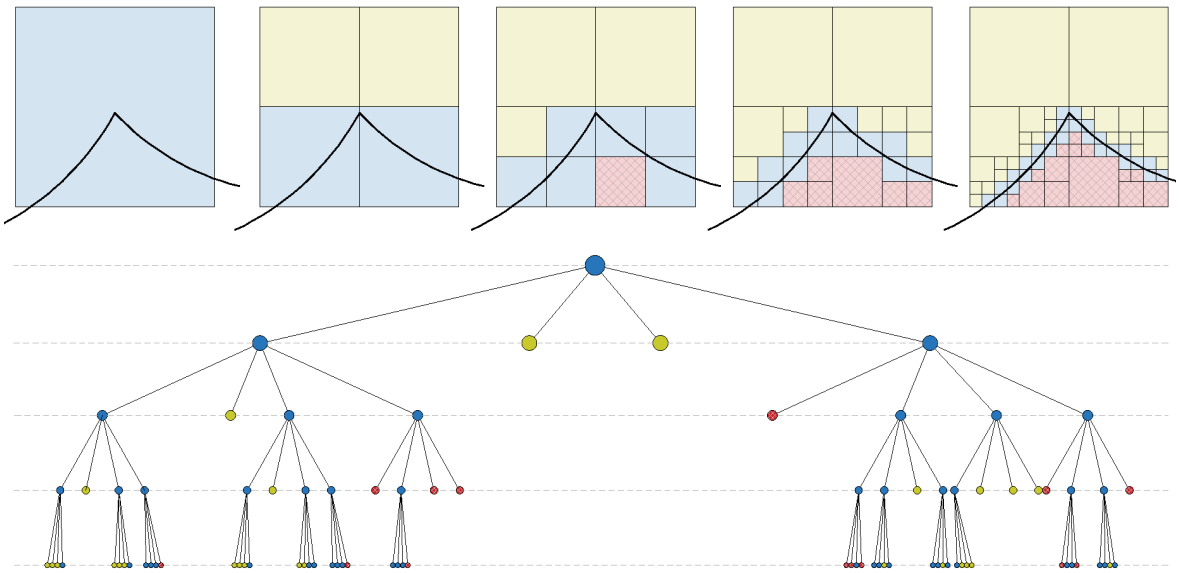


Figure 2.1: Generation of hierarchical models using a recursive subdivision technique within four subsequent levels of refinement. A hierarchical tree structure is used to represent an internal mapping between coarser and finer nodes (parents and children). Three classes of voxels are created in this process: those inside the geometry (red-coloured) , outside of the geometry (yellow-coloured) and, those laying on the border (blue-coloured) voxels.

A particular case of such three-dimensional tree-structure that is obtained using cubic primitives, with a specific limitation that the parent primitive must be bisected in every direction, resulting in eight equal child primitives, is called octree. For the sake of simplicity, an octree generation principle is explained using its two-dimensional equivalent, quadtree (see Fig. 2.1). As can be seen, the quadratic domain that undergoes a recursive subdivision surrounds an arbitrary geometry of interest (Fig. 2.1, upper-left corner). In four subsequent subdivisions, from left to right, only the cells that intersect the geometry are further divided, leading to the classification of all generated cells into three categories: inside (red-coloured), outside (yellow-coloured) and border (blue-coloured) cells. A common name for such created cells is 'voxel' (engl. volume [vox] and element [el]), thus this nomenclature is also adopted throughout the work. In Fig. 2.1, bottom sub-figure a tree-structure used to define the mapping between the parents and children is illustrated, showing also the division onto outside, inside and boundary voxels. The example depicted utilizes only four levels of division, for the sake of clear illustration; nonetheless, this recursive process is continued until the required precision is achieved. The size of voxels obtained in this process on the finest level can be calculated as $h = m/2^{d_{max}}$, where m represents the edge size of the initial domain on the coarsest level

(root level) and d_{max} is the maximum level of refinement (i.e in this depicted case $d_{max} = 4$). Comparing to non-hierarchical models ($\mathcal{O}(n^3)$), where n represents the amount of voxels in one coordinate direction, the complexity of a hierarchical model with the same discretisation quality can be reduced to $\mathcal{O}(n^2)$, which is especially important for large-scale simulations. Non-hierarchical models, on the other side, preserve a memory efficiency, due to the fact that the neighbouring voxels within the model are also located close to each other in the memory, thus their fetching is less expensive than the fetching in the case of hierarchical models. The latter, however, can be partially refined and coarsened without necessity to regenerate a complete mesh, offering a significant advantage to simulation scenarios based on moving geometry. Further discussions on multi-dimensional space-trees, as well as their complexity analysis are published by Frank [50] and M. Bader et al. [84].

Figure 2.2: Boundary representation (STL format, 856 242 triangular elements) of a complex geometry, available free of charge at [138]. The inside of the model is hollow.

A fast octree generator, developed by R. P. Mundani et al. [119], is used throughout this work. It is based on inexpensive geometric collision tests, when converting BREP models (as depicted in Fig. 2.2) into octree volume-based models (illustrated in Fig. 2.3). It can be seen (Fig. 2.3) that rather complex geometry representation can be taken into consideration, letting the user define the required depth of the tree. This influences the amount of voxels generated, as shown in tabular representation below the subfigures, therefore the quality of the discretisation obtained. This octree generator is adapted and embedded into the high-performance fluid framework MPFluid; thus for the sake of consistency, a brief summary of this modification is given in Sec. 5.1. For more details on implementation and efficiency analysis, the user is referred to [51].

DEPTH	1	2	3	4	5	6	7	8	9
Amount of points	64	512	2'360	7'456	24'648	89'328	356'896	1'533'848	7'037'640
Amount of voxels	8	64	295	932	3'081	11'166	44'612	191'731	879'705
Runtime [s]	3.039	3.235	3.869	4.056	4.173	7.091	10.065	12.025	17.857

Figure 2.3: Volume-based representation (octree format) of the complex geometry depicted in Fig. 2.2. The conversion from the BREP model to the volume-based model is done with an octree generator (see [100]) up to the maximum depth of tree $d_{max} = 10$, resulting in 3.64 mil. voxels and more than 29 mil. vertices in the finest case tested. All other cases are listed in the table at the bottom of the figure. The generation of all the cases represented is performed on a local Intel Core i5 Acer Aspire R13 machine with a processor clock speed of 2.3 GHz, sequentially.

2.2 Generation of input files

In the scope of this work, different coupling strategies have been investigated, using various models, such as: 2D and 3D Shallow Water, 3D Navier–Stokes and Darcy's Law through porous media, and in order to run some of these models, a proper numerical domain including complex physical obstacles must be generated. For the sake of clarity, all solid-type obstacles (rocky terrain configuration, for instance, or an urban area with the city infrastructure (see Fig. 2.4)) are created within the voxelisation process (numerical domain generation), whereas definition of the water and air sub-domains can be done at a later stage of the simulation pipeline. A brief overview of the voxelisation process is given in Secs. 2.1 and 5.1, whereas the generation of the input files for those three models in use is thoroughly explained in this chapter.
Unless a CT (computer tomography) scan of a real soil sample has been done, the generation of a synthetic sample that resembles a laboratory specimen must be performed. Bearing in mind that CT scanning might become extremely expensive in the large physical domain of interest, the focus has been put on the generation of a synthetic specimen and this procedure is illustrated in Sec. 2.2.1. In the case of the 2D Shallow Water model, instead of reconstructing solid-type obstacles, the terrain representation is included within the slope reconstruction process; thus, given a real terrain file, an elevation extraction procedure must be performed (see Sec. 2.2.2). Finally, water or any other fluid can enter the numerical domain through the pre-set boundary conditions, or, as is often required, it must be located in a particular part of the domain that has an arbitrary shape at the beginning of the simulation process. This complex insertion is explained in detail in Sec. 2.2.3.

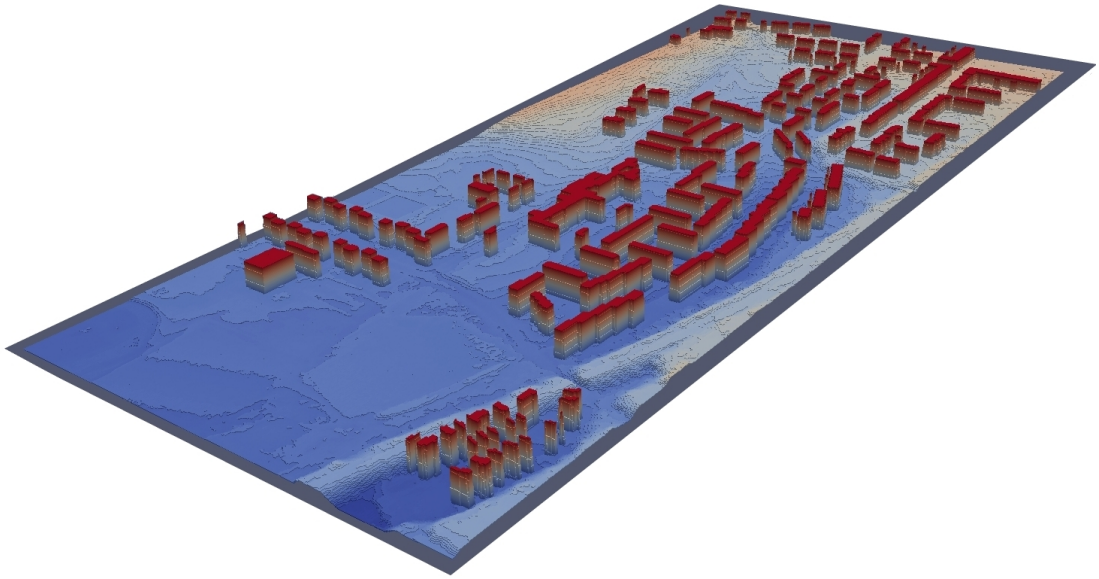


Figure 2.4: A small part of an urban area of the city of Glasgow in Scotland with the building infrastructure extruded. In the geometry generation process, the uneven terrain configuration, including buildings and roads, must be taken into consideration. The digital terrain model is provided from the British Environmental Agency as a part of the 2D flood inundation model benchmarking [59].

2.2.1 Sphere-packing generation

To simulate underground flow through a porous medium on a micro-scale, two essential steps are necessary: 1) to generate a realistic porous-media sample and 2) to couple the mass and momentum conservation laws on the micro-scale with Darcy's Law on the macro-scale. The latter strategy will be thoroughly explained in Secs. 3.3 and 4.2.1, while some further findings and additional sensitivity analysis can be found in [105].
In this chapter, the focus will be put on the generation of a representative porous-media sample, emphasising several important limits and requirements. This problem has been tackled by several research groups and some good results can be found in [88], [150], [151], [152], [71], [70] and [155]. While the emphasis of other research work has been put on simulation of the shear stresses between two adjacent sand grains, within this work the focus is set on the following goals:

# Configuration File for the Random Sphere Generator 2.0			
	d [mm]	%	∑Vsolid [%]
Diameter of sand grain and its volumetric ratio in the total solid volume [mm - %]	200	3	3
	180	8	11
	150	18	29
	125	23.8	52.8
	100	15.3	68.1
	85	14	82.1
	70	6.6	88.7
	60	4.4	93.1
	50	3.1	96.2
	40	2	98.2
	30	1.2	99.4
	20	0.5	99.9
	10	0.1	100
Dimension of the porous sample	Ls = 1 m	Bs = 1 m	Hs = 1 m
Dimension of the bounding box	Lb = 2 m	Bb = 1 m	Hb = 1 m
Porosity value	p [/]	0.5	

Table 2.1: An input file for the Random Sphere Generator, where different diameter classes of sand grains can be defined as well as their volumetric ratio, resulting in the specimen which might contain hundreds of thousands of spheres within predefined boundaries.

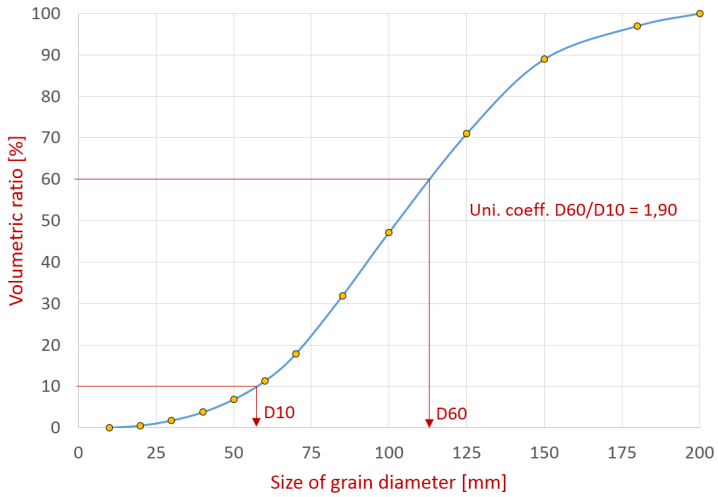


Figure 2.5: A depicted granulometric curve that corresponds to the tabular data representation in Tab. 2.1.

(A) Random insertion of predefined amount of sand grains, belonging to different fractions (see Tab. 2.1) into the quadratic domain with preset boundaries.

(B) Imitation of real granulometric distribution, where the size of the grain is not limited to one discrete value, but rather can take any value between two adjacent sieve diameters, influencing the final porosity parameter.

(C) Horizontal perturbation due to manoeuvring the sieve and vertical settlement due to gravity, resulting in a very compact, randomly distributed specimen with the value of permeability varying across a wide range.

The aim of requirement (A) is to provide as many unique porous-media samples as necessary with the same physical characteristics (i.e. porosity and distribution fractions (see Tab. 2.1)). Nowhere in nature are the sand particles sorted solely into discrete groups of one specific size (e.g. 1 mm or 2 mm); rather, if granulometric analysis of one soil sample were to be carried out, the result would be a sorted group of sand grains smaller than or equal to one specific size ($1\ \text{mm} < D_i \leq 2\ \text{mm}$). In order to mimic this behaviour, aiming at a physically more correct sample representation (requirement (B)), a particular insertion procedure is implemented (see Algs. 1 and 2), which is graphically depicted in Fig. 2.7. For more practical details, the reader is referred to the publication [105].
Due to the manoeuvring of the sieve in the horizontal plane, grains of sand are frequently displaced in all four horizontal directions, moving constantly towards each other and creating more space for those spheres still to be inserted. Additionally, under gravity the sand grains tend to settle down, stacking on top of each other, without the possibility of moving those grains of sand that have already settled (requirement (C)). Horizontal and vertical displacements can be triggered after an arbitrarily chosen number of positioned spheres (see *shakeFactor* Alg. 2), which affects the speed of the sample's generation, but also introduces a segregation process, if the chosen *shakeFactor* is too low.

Algorithm 1 Settlement of the spheres inside the domain

procedure SETTLE DOWN ALL INSERTED SPHERES($SphereList$)
  $dropDistance = Z_{current} - Z_{min}$ ▷ Z_{min} - bottom of the domain
  while $dropDistance > 0$ **do**
    $Z_{current} \leftarrow Z_{current} - \varepsilon$
    function COLLISION TEST($Sphere(Z_{current})$, $SphereList$)
    end function
    if (no-collision) **then**
      $dropDistance \leftarrow dropDistance - \varepsilon$
    end if
  end while
  $SphereList_{settled} \leftarrow Sphere(Z_{current})$
end procedure

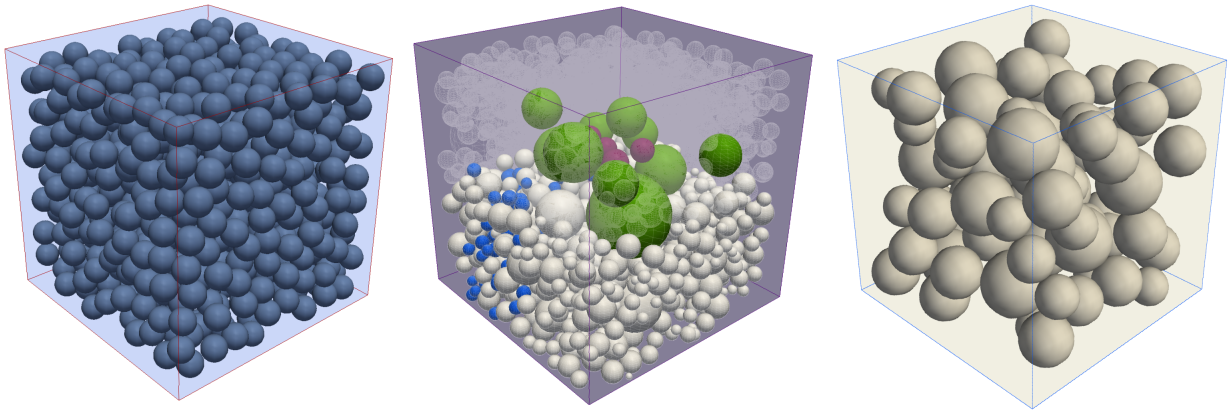


Figure 2.6: Three different porous media settings created within a 1m × 1m × 1m bounding domain. The left-most sample is built out of equally sized sand grains, the right-most sample contains four different coarse diameter classes, packed as densely as possible, and the middle specimen includes all 12 diameter classes, listed in Tab. 2.1 and graphically distributed as shown in Fig. 2.5.

Algorithm 2 Sphere-Package Distribution

```
1:  procedure DISTRIBUTE SAND PARTICLES(Sphere Array)
2:      for i ← 0, N_sp do                          ▷ Total number of spheres inserted N_sp
3:          function PICK RANDOM INSERTION POINT(x, y, z)
4:          end function
5:          for radius ← 0, R_sp do                 ▷ Available sphere sizes R_sp
6:              function COLLISION TEST(Sphere, SphereList)
7:              end function
8:              if (no-collision) then
9:                  SphereList ← Sphere
10:                 i ← (i + 1)
11:                 ShakeFactor ← (ShakeFactor + 1)
12:
13:                 function SETTLE DOWN ALL INSERTED SPHERES(SphereList)
14:                 end function      ▷ This can be done after each or dozen spheres inserted
15:             end if
16:             if (ShakeFactor == ShakeLimit) then    ▷ ShakeLimit is user defined limit
17:                 function SHAKE SPHERES HORIZONTALLY(SphereList)
18:                 end function
19:             end if
20:         end for
21:     end for
22: end procedure
```

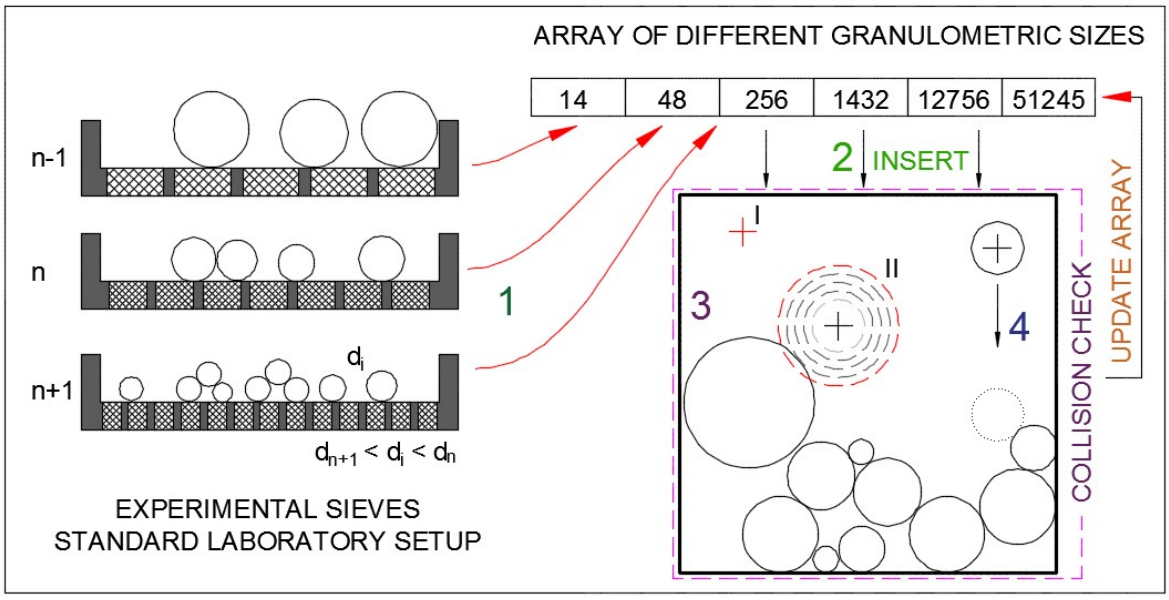


Figure 2.7: Sphere-package generation process, illustrated in four steps: laboratory analysis, grain-size determination, iterative insertion and check-and-update. On the left-hand side, a set of standard laboratory sieves of different grid sizes is depicted. Once the soil sample is put through the sieves (step No.1), a specific amount of material will remain in every sieve, according to the grain size. This amount can be measured in mass or volumetric units (as displayed in Tab. 2.1), and as soon as the volumetric part of one sieve class is known the amount of spherical sand grains within that part can be calculated (step No.2)(see upper, right corner of the figure 'Array of different sizes': the largest fraction has 14 grains, one class smaller has 48, the two classes smaller has 256 grains, etc.). The iterative insertion (step No.3) includes a random positioning of the grain centre point within the boundaries, followed by an iterative increase of the grain size and a simultaneous check whether the collision between the current sphere and all previously inserted ones occurs. The final check-and-update step implies an update of the array, if a sphere has been inserted. The horizontal and vertical displacement of the spheres inside the domain (step No.4) can be done at any point and as frequent as necessary. (see Alg. 2.)

2.2.2 2D terrain elevation extraction

In order to incorporate an uneven terrain configuration within a 3D simulation scenario after the voxelisation process, all the voxels located on the border or inside the geometry (see Sec. 2.1) are set to a solid-cell type and a boundary condition of interest is applied.
On the other hand, within a 2D simulation scenario only one layer of cells in the z-direction is available, being located at $z = 0$ level. The size of the computational cells in the 2D domain is identical to the uniform cell size in the 3D domain. For that reason, an arbitrary B-rep model that represents a terrain configuration can cut this layer of cells, as well as be located above or under the two-dimensional plane (at $z = 0$), leading to the terrain elevation data being not directly available in the 2D computational domain. In order to obtain this information, an elevation extraction of an arbitrarily positioned terrain surface is implemented. It is based on the inverse distance weighting (IDW) method (see Sec. 4.1.4.2). Once available, the terrain elevation can be used directly as a source term within the respective equations. This procedure is used in the vast majority of modern, terrain-modelling commercial software (e.g. ArcGIS [45]). As shown in Fig. 2.8, an arbitrarily shaped STL surface representation can be imported, with a triangle or quadrilateral as the basic element. For the sake of simplicity, in the further text, only triangles will be mentioned as basic elements; nonetheless, the procedure remains the same no matter what type of underlying element is chosen. The vertices V_i of all elements contain the elevation information that should be taken into consideration when creating a smooth terrain surface file. As mathematically defined in Eq. 2.1, the elevation of the vertices (z_i, Fig. 2.8), which are closer to the cell-centre z_c will play a more important role than the elevation of those located far away. The cell-centre z_c represents the centre of the voxel that is a part of the 2D domain.

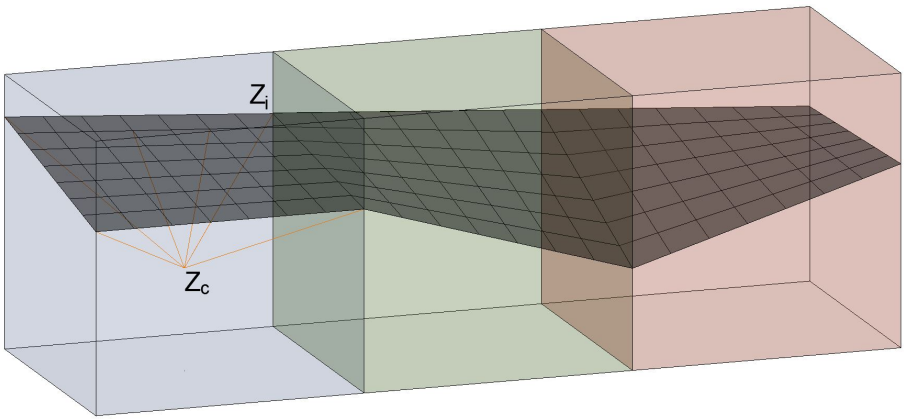


Figure 2.8: Arbitrary STL surface representation, with a quadrilateral underlying basic element, shown within three cubic voxels. The distance between central point z_c and every single vertex z_i, related to the STL geometry, must be computed and used as a pondering factor for their individual elevations.

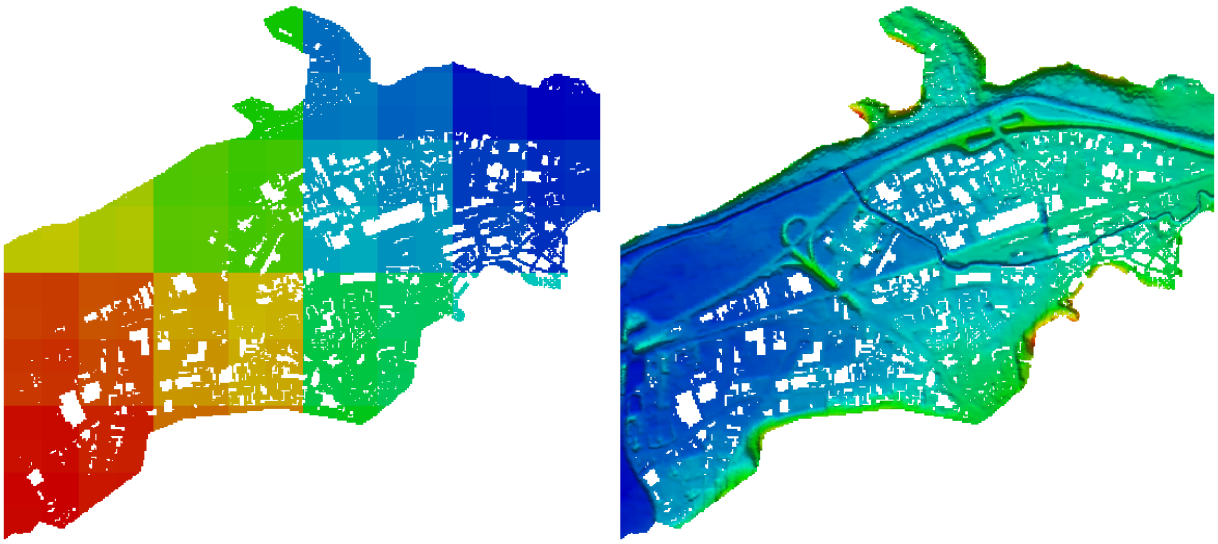


Figure 2.9: Extracted surface elevation (right-hand side) and related subsets of all the vertices, that increase the efficiency of the entire extraction process (left-hand side). The geometry represents the basin of the river White Main, in the Bavarian district of Kulmbach, and it is obtained as a part of the joint project FloodEvac, initiated and funded by the German Federal Ministry of Education and Research [108].

$$\text{Elevation}(x,y) = \begin{cases} \dfrac{1}{\omega_s}\displaystyle\sum_{i=0}^{N}\omega_i z_i, & d(z_c,z_i) > 0, \\ z_c, & d(z_c,z_i) = 0, \end{cases} \tag{2.1}$$

$$\omega_i = \frac{1}{[d(z_c,z_i)]^p}, \qquad \omega_s = \sum_{i=0}^{N}\omega_i \tag{2.2}$$

where z_c represents the elevation of the computing cell-centre, z_i the elevation of the i-th vertex V_i, $d(z_c, z_i)$ the distance between the cell-centre z_c and the elevation z_i of the vertex V_i, and finally the factor p determines the importance of peripheral points in comparison to the central ones. If p = 1, all points are equally involved, while for high values ($p > 16$) only the immediate points will significantly influence the results, thus the smoothness of the terrain elevation largely depends on the choice of this value. In this work the value $p = 2.7$ is used, as numerous tests conducted have confirmed a good concordance between the original STL file and the extracted elevation. Furthermore, in order to speed up the entire process, without significantly decreasing the quality of the results obtained, one can define a subset of all vertices that are relevant for one voxel, eliminating those vertices whose contribution is insignificant, due to the nature of the weighting factors ω_i. How large those subsets should be depends on the voxelisation process in this particular case; still, a rough estimation of an area with a radius $r_i = 10\Delta x$ has shown satisfactory results. In Fig. 2.9, these subsets are depicted on the left-hand side, resulting in the elevation extracted and depicted on the right-hand side.

2.2.3 Insertion of 3D multiple fluid regions

Depending on the problem simulated and the interface recognition method used to represent the interface between two inviscid fluids, the complexity of the fluid phase's insertion into the void or air-filled phase can vary significantly. In this section, the most complex procedure is considered and described in detail, followed by an algorithmic depiction of all the functions and necessary steps conducted. A interface recognition method, such as the marker-and-cell, volume-of-fluid or level-set method, dictates the parameters that must be calculated, as well as the fact that the fluid phase inserted does not have to consist of a single fluid entity; moreover, its distribution in the three-dimensional space is often rather complex. For the sake of clarity, a level-set interface recognition method will be considered here, and the fluid phase that has to be inserted will be defined using a standard STL triangulated surface representation. For the different geometrical formats and various underlying elements, such as quadrilaterals, pentagons, hexagons, etc., the insertion procedure should be the same; nonetheless, the underlying functions must be adapted to the particular basic element.

A more theoretical background of the level-set method and how it is implemented in the current work will be shown in Sec. 3.6. At this point it is important to remark that for both the standard and conservative level-set (SLS, CLS) definitions, the minimum distance function $d(x, y, z)$ from the fluid surface must be calculated (see Eqs. 2.3, 2.4).

$$\textbf{SLS:} \qquad \Phi(x) = d^{\perp}(x, y, z) = \min(|\vec{\mathbf{x}} - \vec{\mathbf{x}_\mathbf{I}}|), \tag{2.3}$$

$$\textbf{CLS:} \qquad \Phi(x, t) = \frac{1}{2}\left[\tanh\left(\frac{d(x, y, z)}{2\varepsilon}\right) + 1\right] \tag{2.4}$$

where $\Phi(x)$ depicts the Level-Set regularized function, ε the thickness of the interface profile between two arbitrary fluids, $\vec{\mathbf{x}}$ is the current position for which the minimum distance function must be calculated and, $\vec{\mathbf{x}_\mathbf{I}}$ represents the position of the vertex of a triangular element within the interface, regarding to which the minimum distance function is calculated, for I$\in [1, 2, 3]$.

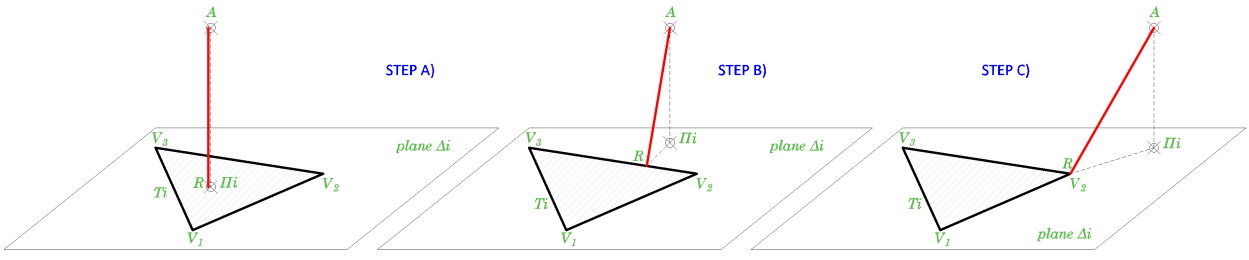


Figure 2.10: Computation of the minimum distance function $d(A, R)$ in three steps A, B, C, each of which, if successful, excludes all remaining steps - A) a normal distance function from the point to the plane, B) a shortest distance function from the point to the edge of the triangle and C) a shortest distance function from the point to the vertex.

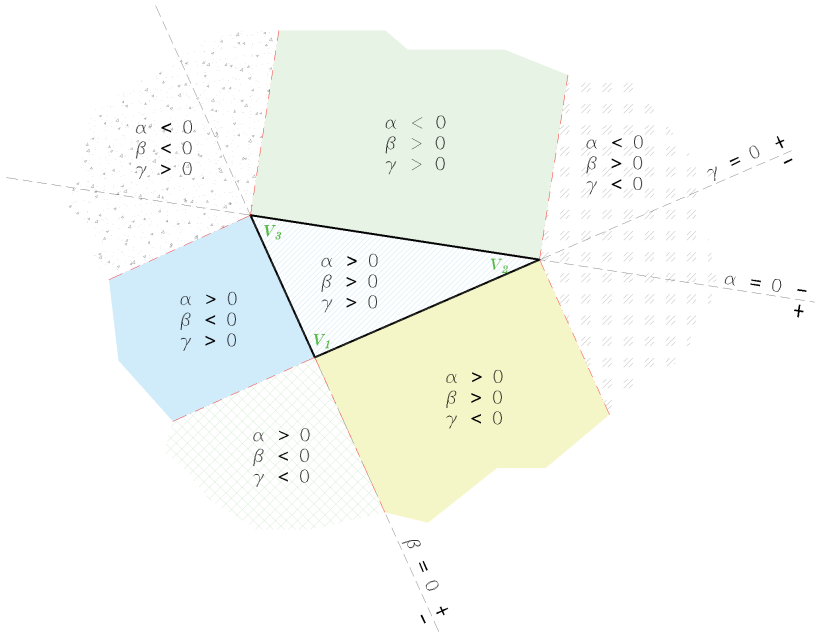


Figure 2.11: Calculation of the barycentric coordinates of an STL surface triangular element. Based on the sign of all three values, it is decided, whether it is necessary to calculate distance to the plane (STEP A), distance to the edge (STEP B) or distance to the vertex (STEP C). All three steps are graphically depicted in Fig. 2.10.

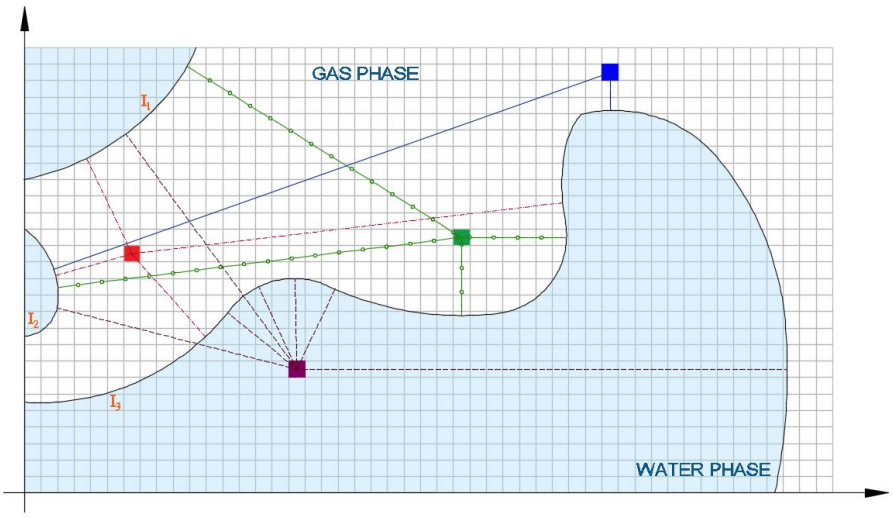


Figure 2.12: Calculation of the signed distance function Φ must be performed for every computational cell with respect to every interface I_i. In case of a complex shape of the interface, multiple intersections are possible, whereas only the shortest normal distance is relevant for the global function Φ.

The minimum distance function for a single fluid entity, whose shape can easily be defined mathematically (e.g. sphere or cube), can be calculated straightforwardly, avoiding several steps in the procedure further described. Adding additional arbitrarily positioned fluid entities, as illustrated in Fig. 2.12, complicates the entire process massively, as every mesh point in the

Algorithm 3 Minimum-Distance Value Calculation

```
1: procedure CALCULATE MINIMUM DISTANCE FROM INTERFACE(Point A)
2:     STL container consists of N_e elements
3:     for i ← 1, N_e do
4:         function CALC PROJECTED POINT AND BARYCENTRIC COORDINATES(α, β, γ, Π)
5:         end function
6:
7:         if (α ≥ 0 & β ≥ 0 & γ ≥ 0) then                    ▷ Execute STEP A
8:             function MINIMUM NORMAL DISTANCE(α, β, γ, norm_dist)
9:             end function
10:        end if
11:        if (norm_distance_exists) then
12:            function COMPARE(norm_dist, old_value)
13:            end function
14:        else                                                ▷ Execute STEP B
15:            function MINIMUM EDGE DISTANCE(α, β, γ, edge_dist)
16:            end function
17:        end if
18:        if (edge_distance_exists) then
19:            function COMPARE(edge_dist, old_value)
20:            end function
21:        else                                                ▷ Execute STEP C
22:            function MINIMUM VERTEX DISTANCE(α, β, γ, vertex_dist)
23:            end function
24:            function COMPARE(vertex_dist, old_value)
25:            end function
26:        end if
27:    end for
28: end procedure
```

domain discretized must be checked against each and every triangular element used to describe those entities (i.e. fluid entities are commonly imported as an STL surface representation file). For the sake of clarification, in Fig. 2.12, four different mesh points are chosen and they have to be compared against every water surface. Only the global minimum distance for one particular point is relevant for the further calculation of the level-set value.

In general, the signed distance function from the surface does not always represent the shortest orthogonal distance, which is why it is necessary to perform the three consecutive steps that include a calculation of the minimum distance value between: A) the chosen point A and the plane Δ_i that contains one surface triangle T_i; B) the chosen point A and the nearest edge E_{ij} of the triangle T_i; and C) the chosen point A and the nearest vertex V_{ik} of the previously introduced geometric entity (triangle T_i) (see Fig. 2.10 for clarification). The successful calculation of one of the distance values makes all the remaining steps unnecessary. This procedure should be repeated for each element in the STL geometry file, which is the reason why the minimum distance calculation is considered to be very expensive and is done only once at the very beginning of the simulation scenario. A brief explanation of all three

steps – A, B and C – is given below, including the algorithmic representation of the entire procedure.

Algorithm 4 Barycentric-Coordinates Calculation

procedure CALC PROJECTED POINT AND BARYCENTRIC COORDINATES($\alpha, \beta, \gamma, \Pi$)
    Normal vector of a surface element:
    for $i \leftarrow 0, 3$ **do**
        VEC1 = V1[i] - V3[i]
        VEC2 = V2[i] - V1[i]
        function CALC VECTOR PRODUCT(VEC1, VEC2, norm)
        end function
    end for
    Project the point A onto the plane:
     Projected distance $\leftarrow \vec{AV} \cdot \vec{\mathbf{norm}}$
    for $i \leftarrow 0, 3$ **do**
        Projected point $\Pi[i]$ = A[i] - nnorm[i]*Projected distance ▷ nnorm - normalized normal vector
    end for
    Barycentric coordinates:
    VEC1 = V2[i] - V1[i]
    VEC2 = V3[i] - V1[i]
    VEC3 = Π[i] - V1[i]

    DP11 = VEC1 · VEC1
    DP12 = VEC1 · VEC2
    DP22 = VEC2 · VEC2
    DP31 = VEC3 · VEC1
    DP32 = VEC3 · VEC2 ▷ dot product <·>

    denominator = DP11 * DP22 - DP12 * DP12;
    β = (DP22 * DP31 - DP12 * DP32) / denominator;
    γ = (DP11 * DP32 - DP12 * DP31) / denominator;
    α = 1.0 - β - γ;
end procedure

In order to speed up the process and immediately choose an appropriate step out of the three named above, the barycentric coordinates of a triangle are calculated as published by Ericson [41] and shown in Alg. 4. Once the three coordinates α, β, γ are available, based on their sign (see Fig. 2.11), it can be easily defined whether the minimum distance function equals the normal orthogonal direction ($\alpha+, \beta+, \gamma+$), or whether the distance from the edge and vertex must be calculated. The calculation of the three steps must be executed in this exact order (A, B and C) (see Alg. 3), and if step A returns a meaningful value, both steps B and C will be discarded. Likewise, if step B returns the minimum distance value, there is no need for the execution of the third step C.

Algorithm 5 Old-New Value Comparison

procedure COMPARE(new, current)
  if (current ≤ 0 & new ≤ 0) **then**
    current $\leftarrow max$(new, current)
  else if (current ≥ 0 & new ≥ 0) **then**
    current $\leftarrow min$(new, current)
  else if (current $*$ new ≤ 0) **then**
    if (|new| $\leq$ |current|) **then** current $\leftarrow$ new
    end if
  end if
end procedure

Algorithm 6 Minimum-Edge-Distance Calculation

procedure MINIMUM EDGE DISTANCE(α, β, γ, edge_dist)
  if ($\alpha < 0$ & $\beta \geq 0$ & $\gamma \geq 0$) **then**
    function CALC PROJECTION POINT ONTO FACE EDGE(V2, V3, A, R, distance2edge)
    end function
  else if ($\alpha \geq 0$ & $\beta < 0$ & $\gamma \geq 0$) **then**
    function CALC PROJECTION POINT ONTO FACE EDGE(V1, V3, A, R, distance2edge)
    end function
  else if ($\alpha \geq 0$ & $\beta \geq 0$ & $\gamma < 0$) **then**
    function CALC PROJECTION POINT ONTO FACE EDGE(V1, V2, A, R, distance2edge)
    end function
  end if
end procedure

Algorithm 7 Projection-Onto-Edge Calculation

procedure CALC PROJECTION POINT ONTO FACE EDGE(Vs, Vt, A, R, distance2edge)
  for $i \leftarrow 0, 3$ **do**
    $\overrightarrow{EDGE}$[i] = Vs[i] - Vt[i]
    $\overrightarrow{AV_s}$[i] = A[i] - Vs[i]
  end for
  projection_A_ontoEDGE = $\frac{(\overrightarrow{AV_s} \cdot \overrightarrow{EDGE})}{||\overrightarrow{EDGE}||}$ ▷ dot product <·>
  if (0 $\leq$projection_A_ontoEDGE $\leq ||\overrightarrow{EDGE}||$) **then**
    ▷ Projection Point R exists (see Fig. 2.10), Step B
    distance2edge = $\sqrt{||\overrightarrow{AV_s}||^2 - (\text{projection\_A\_ontoEDGE})^2}$
  end if
end procedure

Algorithm 8 Minimum-Vertex-Distance Calculation

```
1: procedure MINIMUM VERTEX DISTANCE(α, β, γ, vertex_dist)
2:     if (α ≥ 0 & β ≤ 0 & γ ≤ 0) then
3:         function CALC PROJECTION POINT ONTO FACE EDGE(V1, V2, A, R, distance)
4:         end function
5:         if (no distance calculated) then
6:             function CALC PROJECTION POINT ONTO FACE EDGE(V1, V3, A, R, distance)
7:             end function
8:             if (no distance calculated) then distance ← dist(Point A, Point V1)
9:             end if
10:        end if
11:    else if (α ≤ 0 & β ≥ 0 & γ ≤ 0) then
12:        function CALC PROJECTION POINT ONTO FACE EDGE(V1, V2, A, R, distance)
13:        end function
14:        if (no distance calculated) then
15:            function CALC PROJECTION POINT ONTO FACE EDGE(V2, V3, A, R, distance)
16:            end function
17:            if (no distance calculated) then distance ← dist(Point A, Point V2)
18:            end if
19:        end if
20:    else if (α ≤ 0 & β ≤ 0 & γ ≥ 0) then
21:        function CALC PROJECTION POINT ONTO FACE EDGE(V1, V3, A, R, distance)
22:        end function
23:        if (no distance calculated) then
24:            function CALC PROJECTION POINT ONTO FACE EDGE(V2, V3, A, R, distance)
25:            end function
26:            if (no distance calculated) then distance ← dist(Point A, Point V3)
27:            end if
28:        end if
29:    end if
30: end procedure
```

Algorithm 9 Minimum-Normal-Distance Calculation

```
1: procedure MINIMUM NORMAL DISTANCE(α, β, γ, norm_dist)
2:     Normal distance ← A⃗V · nor⃗m          ▷ V - Vertex of the triangular element
3: end procedure
```

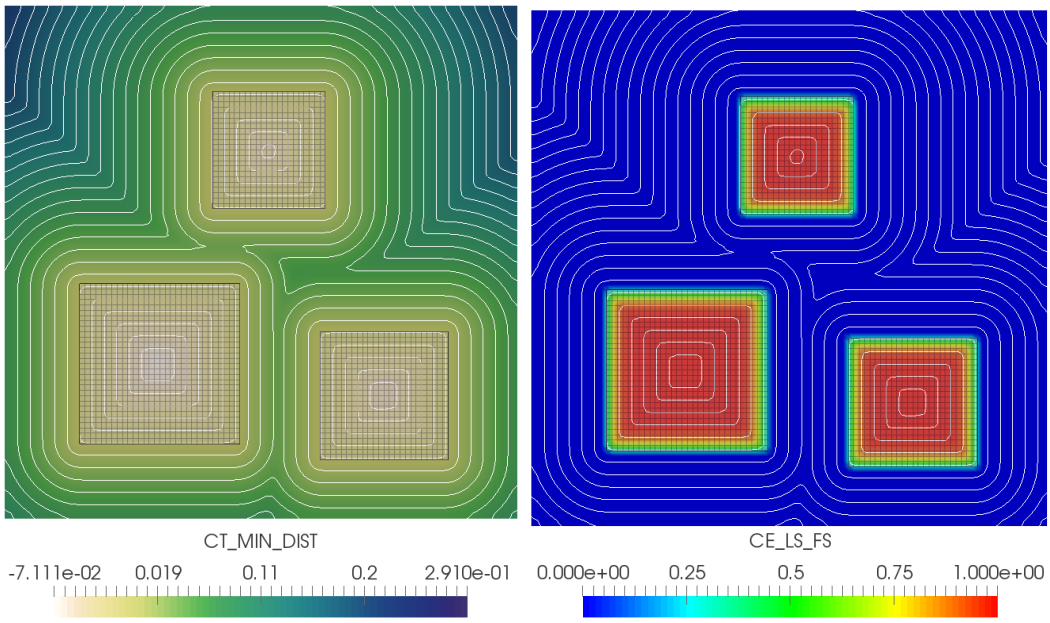


Figure 2.13: Insertion of multiple cubic fluid entities, arbitrarily positioned in space. On the left-hand side a cumulative minimum distance function is depicted using contour plot, whereas on the right-hand side the level-set function is represented, taking the value 1 in the fluid phase and the value 0 in the gaseous phase.

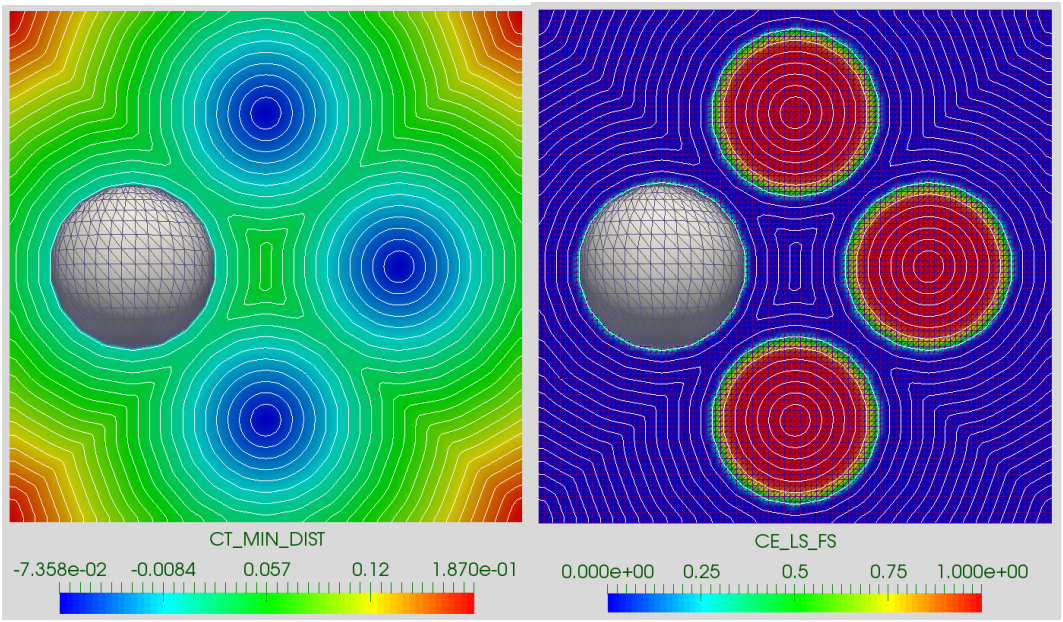


Figure 2.14: Insertion of multiple spherical fluid entities, arbitrarily positioned in space. On the left-hand side a cumulative minimum distance function is depicted using contour plot, whereas on the right-hand side the level-set function is represented, taking the value 1 in the fluid phase and the value 0 in the gaseous phase.

As it can been seen in Figs. 2.13 and 2.14, the indicator function Φ defined as in Eq. 3.60,

takes the value 1 in the fluid region and the value 0 in the gaseous phase, whereas the exact position of the interface **I** correlates to the value of $\Phi = 0.5$. This initialised indicator function is advected using the interface velocity **u** as a part of the advection process formulated in Eq. 3.55.

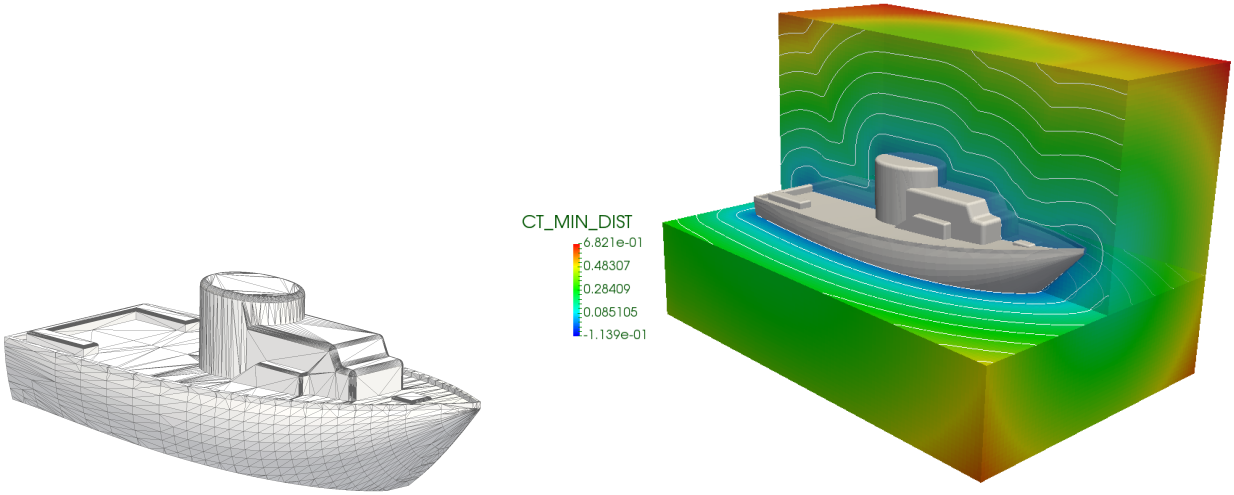


Figure 2.15: Insertion of a complex geometry representation (STL Boat Data-file contains 5862 triangular elements, [76]). The cumulative minimum distance function is calculated as explained in Algs. 3 - 5.

Chapter 3

Mathematical models

The field of Computational Fluid Dynamics, which covers a number of different flow phenomena, is very broad, as can be seen in the brief overview in section 1.1 that deals with the state-of-the-art and recent advances. Describing the main physical laws of fluid flow, in a general sense only, their mathematical characterisation and corresponding numerical representation is tedious work that requires, besides a lot of fundamental research, an immense amount of time and space. The intention of this work is not to try to cover all aspects of different fluid-flow occurrences, but rather to focus on the detailed mathematical and numerical representation, as well as useful practical implementation of two different fluid-flow families, namely single-phase Stokes flow through an underground porous medium and two-phase inviscid Newtonian flow in different hydraulic regimes, including resolution of shockwaves and rarefactions at the contact surfaces. Both fluid-flow families are mathematically formulated using the general form of Navier–Stokes equations

$$\frac{\partial}{\partial t}(\rho \mathbf{u}) = \nabla \cdot (\rho \mathbf{u}\mathbf{u}) + \mathbb{S} = 0, \tag{3.1}$$

with ρ being the density of respective fluid, $\mathbf{u}$ representing the velocity vector field with its components $\mathbf{u} = \{u, v, w\}$ in three main Cartesian directions, and $\mathbb{S}$ being the sum of all external forces applied onto a mass unit. This general form is obtained by application of conservation principles to the momentum $\rho\mathbf{u}$ and is further constrained depending on the particular phenomena that are of interest to the research, thus several different additional models will be introduced, starting with Darcy's Law describing flow through a porous medium, then a simplification of Stokes flow, depth-averaged shallow water formulation and an interface tracking approach necessary for multi-phase flow conjunction.

In order to apply the conservation of mass, momentum and energy principles (detailed description is given in the following Sec. 3.1), it is necessary to consider the fluid as a continuum, rather than observing separate particles on a molecular level, thus all the fluid properties describing fluid motion are at least weakly differentiable. Deployment of those principles and derivation of a conservative formulation of the Navier–Stokes equations, which will be extensively used in this work, will be shown in a concise fashion. Nevertheless, for more details and different development directions that have been taken throughout its history, the interested reader is referred to the standard literature (J. H. Ferziger and M. Perić [65], C. Hirsch [22] and Bear [18], for instance).

3.1 Basic conservation principles

Within this section a brief overview of the derivation of basic conservation principles will be provided, following the structure formulated in [65]. In most of the engineering phenomena considered, for a given quantity of matter, the rate of change of the extensive properties is equal to zero, as mass cannot be generated or destroyed, or is equal to some well-defined parameter that accounts for the influence of external forces on the system. Extensive properties, such as mass, momentum or energy, are measured within a unit control mass (CM) and they depend on the amount of matter being considered. Intensive properties, on the other hand, are derived from their extensive counterparts and they are independent of the amount of matter being examined (for example, ρ). All further analysis will utilise the intensive property formulation. For that reason it is necessary to define a unit control volume (CV) over which the necessary integration will be performed.
Two well-known approaches in the conservation analysis are the Lagrangian approach, considering a moving package of matter, and the Eulerian approach, based on the measurement of the rate of matter moving through a spatially fixed unit. Both approaches are used in the field of computational fluid dynamics, although the Eulerian approach has certain advantages, as the unit of matter being considered is not always easy to identify.
Both laws for the given quantity of matter are mathematically formulated as:

$$\frac{\mathrm{d}m}{\mathrm{d}t} = 0 \tag{3.2}$$

$$\frac{\mathrm{d}}{\mathrm{d}t}(m\mathbf{u}) = \mathbb{S}, \tag{3.3}$$

where m is the quantity of matter examined, and $m\mathbf{u}$ stands for the momentum of given mass, balanced by the sum of external forces applied onto the unit CM. A transformation from control mass (CM) to control volume (CV) defines the relation between extensive Ψ and intensive ψ fluid properties, formally expressed as in Eq. 3.4:

$$\Psi = \int_{\Omega_{CM}} \rho\psi \mathbf{d}\Omega_{CM}, \tag{3.4}$$

with ρ being mass per unit volume and Ω_{CM} the volume of predefined control mass unit.
Substituting Eq. 3.4 into the conservation law in Eq. 3.2 and applying the Reynolds transport theorem, the following form is obtained:

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\int_{\Omega_{CM}} \rho\psi \mathbf{d}\Omega_{CM}\right) = \int_{\Omega_{CV}} \frac{\partial}{\partial t}(\rho\psi)\mathbf{d}\Omega_{CV} + \oint_{F_c} (\mathbf{u}\cdot\mathbf{n}_f)\rho\psi \ \mathbf{d}F_c, \tag{3.5}$$

where F_c represents the surface of chosen control volume CV and $\mathbf{n}_f$ the unit vector perpendicular to the surface CV pointing outwards. The last term on the right-hand side of Eq. 3.5 is transformed into the volume integral, using Gauss' divergence theorem. The final, general formulation, arising from Eqs. 3.2 and 3.5, has the following composition:

$$\int_{\Omega_{CV}} \frac{\partial}{\partial t}(\rho\psi)\mathbf{d}\Omega_{CV} + \int_{\Omega_{CV}} \nabla\cdot(\rho\psi\mathbf{u}) \ \mathbf{d}\Omega_{CV} = 0 \tag{3.6}$$

Eq. 3.6 will be actively exploited in the further formulation of the mass, momentum and energy conservation principles.

3.1.1 Conservation of mass

Global mass conservation is satisfied inherently within Eq. 3.6, if no additional sources or sinks are introduced. For the constant value $\psi = 1$ and under assumption of an infinitesimally small control volume over which the integration is performed, this equation is transformed into the well-known form:

$$\int_{\Omega_{CV}} \left[\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) \right] \mathbf{d}\Omega_{CV} = 0 \tag{3.7}$$

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0 \tag{3.8}$$

The latter form is independent of the coordinate system used for the discretisation, under the hypothesis that the definition of the divergence operator in that system is known. As orthogonal structured grids in the Cartesian coordinate system are extensively used in this work, the two common forms of Eq. 3.8, which are used in the finite difference and finite volume formulation, respectively, are adopted here:

$$\frac{\partial \rho}{\partial t} + \frac{\partial \rho u}{\partial x} + \frac{\partial \rho v}{\partial y} + \frac{\partial \rho \omega}{\partial z} = 0 \tag{3.9}$$

$$\frac{\partial \rho}{\partial t} + \frac{1}{V_c} \sum_{f \in F(c)} \rho_f \mathbf{u} \cdot \mathbf{n_f} A_f = 0, \tag{3.10}$$

where (u, v, ω) are the Cartesian components of the velocity vector $\mathbf{u}$, A_f is the area of the control volume's face and V_c represents the volume over which the integration has been performed.

3.1.2 Conservation of momentum

Taking into account the momentum conservation law, expressed in Eq. 3.3, a procedure, similar to that which is done in the derivation of the mass-conservation expression, has to be performed, with the difference being that $\psi = \mathbf{u}$ instead of $\psi = 1$. The common form obtained, which will be further analysed, is depicted as:

$$\int_{\Omega_{CV}} \frac{\partial}{\partial t}(\rho \mathbf{u}) \mathbf{d}\Omega_{CV} + \int_{\Omega_{CV}} \nabla \cdot (\rho \mathbf{u}\mathbf{u}) \, \mathbf{d}\Omega_{CV} = \int_{\Omega_{CV}} \mathbb{S} d\Omega_{CV} \tag{3.11}$$

Assuming an infinitesimal size of the integration domain Ω_{CV}, a differential form is acquired

$$\frac{\partial}{\partial t}(\rho \mathbf{u}) + \nabla \cdot (\rho \mathbf{u}\mathbf{u}) = \mathbb{S} \tag{3.12}$$

where $\mathbb{S}$ comprises all the surface and volumetric forces applied to the system. Taking into account the Cauchy momentum equation that guarantees the conservation of momentum for any continuum that is mass-conservative, the forces on the system can be summarised as follows:

$$\int_{\Omega_{CV}} \mathbb{S} d\Omega_{CV} = \oint_{F_c} \sigma \mathrm{d}F_c + \int_{\Omega_{CV}} f \mathrm{d}\Omega_{CV} \tag{3.13}$$

$$\mathbb{S} = \nabla \cdot \sigma + f, \tag{3.14}$$

in which the term f accounts for the influence of volumetric forces (e.g. gravity, Coriolis or centrifugal forces, electromagnetic forces, etc.), while the stress tensor σ represents normal and tangential surface forces (for instance, pressure, surface tension, shear stresses, etc.) as shown below:

$$\sigma = \begin{bmatrix} \sigma_{11} & \tau_{12} & \tau_{13} \\ \tau_{21} & \sigma_{22} & \tau_{23} \\ \tau_{31} & \tau_{32} & \sigma_{33} \end{bmatrix} = - \underbrace{\begin{bmatrix} p & 0 & 0 \\ 0 & p & 0 \\ 0 & 0 & p \end{bmatrix}}_{\mathbb{P}} + \underbrace{\begin{bmatrix} \sigma_{11} + p & \tau_{12} & \tau_{13} \\ \tau_{21} & \tau_{22} + p & \tau_{23} \\ \tau_{31} & \tau_{32} & \sigma_{33} + p \end{bmatrix}}_{\mathbb{T}} \tag{3.15}$$

Here the terms $\mathbb{P}$ and $\mathbb{T}$ are the normal and deviatoric parts of the stress tensor, respectively, with p being evaluated as a mean value of three normal stresses $p = -\frac{1}{3}(\sigma_{11} + \sigma_{22} + \sigma_{33})$. This separation is motivated by an easier evaluation of both terms individually; moreover, the influence of normal stresses is of immense importance for the CFD field and must be accurately determined, while the term $\mathbb{T}$ is often approximated differently, depending on the class of fluid problems solved.

As the main focus of this work is the analysis of Newtonian fluids, within which linear correlation between non-isotropic components of the stress τ_{ij} and the rate of the strain $\frac{\mathrm{d}\epsilon}{\mathrm{d}t}$ is assumed, only those correlations will be presented next, whereas the general behaviour of some different classes such as dilatants, pseudoplastic or Bingham fluids can be revised in the standard literature.

$$\tau_{ij} = \mu \left[\nabla \mathbf{u} + (\nabla \mathbf{u})^T \right] + \delta \lambda \nabla \cdot \mathbf{u} \tag{3.16}$$

The linear dependence of the strain rate and particular components of stress can be represented in its vector form as depicted in Eq. 3.16. The second part of the equation accounts for viscous effects associated with the volumetric change; nevertheless, in the majority of simulated engineering cases, even for compressible fluids, this part is negligible, thus it will not be discussed any further. The first part has the property of being symmetric and positive-definite, and, as such, contributes to dissipation effects in the energy conservation process.

3.1.3 Conservation of energy

Unlike the mass and momentum conservation laws, which are derived for the direct unknowns of the system, and therefore are rather straightforward to satisfy, the conservation of kinetic energy

or any other indirectly derived quantity (e.g. vorticity or entropy) cannot be imposed directly in the process of construction of a numerical method (taking into account the discretisation method and mesh arrangement setting). There is a well-elaborated discussion about conservation properties with respect to the collocated and staggered mesh arrangement published in L. Jofre et al. [80], emphasising the importance of the preservation of kinetic energy, especially within the simulation of turbulent flows, where the rate of total kinetic energy in the absence of external sources or sinks must be equal to the numerical dissipation, formulated in Eq. 3.16. In other words, the kinetic energy in a turbulent flow is, due to convection, only carried from eddies at the larger scale to eddies at the smaller scale in a recursive manner, until it reaches a molecular level where, due to the influence of diffusion, it is completely dissipated.

In order to shed light on the conservation of kinetic energy, the properties of mathematical operators in the momentum equation must meet several requirements. Starting from the formulation of the momentum equation given in Eq. 3.12 and taking into account the fact that the source term $\mathbb{S}$ combines the volumetric, normal forces and surface, tangential forces as depicted in Eqs. 3.14–3.16, a more general formulation of the momentum equation (Eq. 3.17), where the volumetric forces are kept in the already introduced form, while the surface forces are split into the pressure and diffusion components, is used to clarify those requirements:

$$\Omega_{cv}\frac{\partial}{\partial t}(\rho\mathbf{u}) + \mathcal{C}(\rho\mathbf{u})\mathbf{u} = -\mathcal{G}p + \mathcal{D}(\mu)\mathbf{u} + \Omega_{cv}f \tag{3.17}$$

where $\mathcal{C}$, $\mathcal{D}$ and $\mathcal{G}$ represent the convective, diffusive and gradient operator respectively, and $\mathbb{S}$, p, $\mathbf{u}$ and Ω_{cv} stand for the external source term, pressure, velocity vector and integration control volume. As reported by Verstappen et al. [145], the convective operator $\mathcal{C}$ must be skew-symmetric, diffusive operator $\mathcal{D}$ symmetric, and the dot product anti-symmetric.

anti-symmetry	$\nabla \cdot \vec{a} \cdot \theta = -\vec{a} \cdot \nabla\theta$	dot product $\langle\vec{a}\vert\theta\rangle$
skew-symmetry	$A = -A^*$	convective operator
symmetry	$A = A^T$	diffusive operator ∇^2

Starting from Eq. 3.17, the energy conservation equation (see Eq. 3.18) is obtained. If previous requirements are applied to the operators and dot products named, as shown in Eqs. 3.18, 3.19 and 3.20, the influence of particular terms can be estimated.

$$\Omega_{cv}\frac{\partial}{\partial t}(\rho\mathbf{u})\cdot\mathbf{u} + \mathcal{C}(\rho\mathbf{u})\mathbf{u}\cdot\mathbf{u} = -\mathcal{G}p\cdot\mathbf{u} + \mathcal{D}(\mu)\mathbf{u}\cdot\mathbf{u} + \Omega_{cv}f\cdot\mathbf{u} \tag{3.18}$$

$$\langle\mathcal{C}(\mathbf{u},\rho\mathbf{u})\vert\mathbf{u}\rangle = -\langle\mathcal{C}(\mathbf{u},\mathbf{u})\vert\rho\mathbf{u}\rangle = 0, \tag{3.19}$$

$$\langle\mathcal{G}p\vert\mathbf{u}\rangle = \langle\, p\,\vert\nabla\cdot\mathbf{u}\rangle = 0, \tag{3.20}$$

$$\Omega_{cv}f\cdot\mathbf{u} = 0, \tag{3.21}$$

$$\langle\mathcal{D}(\mu)\mathbf{u}\vert\mathbf{u}\rangle = \langle\mu\nabla^2\mathbf{u}\vert\mathbf{u}\rangle = -\mu\langle\nabla\mathbf{u}\vert\nabla\mathbf{u}\rangle = -\mu\|\nabla\mathbf{u}\|_{L2}^2 \leq 0 \tag{3.22}$$

Obviously, under the assumption of non-existent external sources (Eq. 3.21), the convective (Eq. 3.19) and pressure (Eq. 3.20) terms do not contribute to the rate of kinetic energy, whereas the diffusion term (Eq. 3.22) is always negative, leading to the dissipation of energy at the molecular level, as stated above.

$$\frac{\partial}{\partial t}E_k = \Omega_{cv}\frac{\partial}{\partial t}(\rho\mathbf{u})\cdot\mathbf{u} = -\mu\|\nabla\mathbf{u}\|_{L2}^2 \tag{3.23}$$

The choice of numerical schemes for the convective and pressure terms particularly can lead to introduction of additional 'non-physical' dissipation, in which case the solution might still conserve energy, but the physics of the problem solved may significantly change.

3.2 Derivation of incompressible Navier–Stokes equations

Considering conservation laws, previously introduced in Secs. 3.1.1, 3.1.2 and 3.1.3, a straightforward substitution of total stresses (see Eq. 3.14) into the conservation of momentum (Eq. 3.12) leads to the well-known form of Navier–Stokes set of equations:

$$\frac{\partial}{\partial t}(\rho\mathbf{u}) + \nabla\cdot(\rho\mathbf{u}\mathbf{u}) = -\nabla p + \nabla\cdot\left[\mu\left(\nabla\mathbf{u} + \nabla^T\mathbf{u}\right)\right] + \rho\mathbf{g} \tag{3.24}$$

Further generalisations that arise from the incompressibility constraint, may simplify this expression even more, leading to the disappearance of the cross-diffusion term $\nabla^T\mathbf{u}$. Moreover, in the case of single-phase flows, the transport properties μ and ρ are considered to be constant, which finally removes much of the complexity from Eq. 3.24, resulting in the following equation:

$$\frac{\partial\mathbf{u}}{\partial t} + \nabla\cdot(\mathbf{u}\mathbf{u}) = -\frac{1}{\rho}\nabla p + \nu\nabla^2\mathbf{u} + \mathbf{g} \tag{3.25}$$

Although the expression in Eq. 3.25 is very common in scientific literature, describing incom-

pressible Navier–Stokes equations, it should be kept in mind that this form can be applied

$$\frac{\partial \rho u}{\partial t} + \frac{\partial \rho u u}{\partial x} + \frac{\partial \rho u v}{\partial y} + \frac{\partial \rho u w}{\partial z} = -\frac{\partial p}{\partial x} + \\ + \frac{\partial}{\partial x}\left[\mu\left(\frac{\partial u}{\partial x} + \frac{\partial u}{\partial x}\right)\right] + \frac{\partial}{\partial y}\left[\mu\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right)\right] + \frac{\partial}{\partial z}\left[\mu\left(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x}\right)\right] + \rho g_x \quad (3.26)$$

$$\frac{\partial \rho v}{\partial t} + \frac{\partial \rho v u}{\partial x} + \frac{\partial \rho v v}{\partial y} + \frac{\partial \rho v w}{\partial z} = -\frac{\partial p}{\partial y} + \\ + \frac{\partial}{\partial x}\left[\mu\left(\frac{\partial v}{\partial x} + \frac{\partial u}{\partial y}\right)\right] + \frac{\partial}{\partial y}\left[\mu\left(\frac{\partial v}{\partial y} + \frac{\partial v}{\partial y}\right)\right] + \frac{\partial}{\partial z}\left[\mu\left(\frac{\partial v}{\partial z} + \frac{\partial w}{\partial y}\right)\right] + \rho g_y \quad (3.27)$$

$$\frac{\partial \rho w}{\partial t} + \frac{\partial \rho w u}{\partial x} + \frac{\partial \rho w v}{\partial y} + \frac{\partial \rho w w}{\partial z} = -\frac{\partial p}{\partial z} + \\ + \frac{\partial}{\partial x}\left[\mu\left(\frac{\partial w}{\partial x} + \frac{\partial u}{\partial z}\right)\right] + \frac{\partial}{\partial y}\left[\mu\left(\frac{\partial w}{\partial y} + \frac{\partial v}{\partial z}\right)\right] + \frac{\partial}{\partial z}\left[\mu\left(\frac{\partial w}{\partial z} + \frac{\partial w}{\partial z}\right)\right] + \rho g_z \quad (3.28)$$

straightforwardly only on single-phase flows, whereas multi-phase incompressible flows must be treated with more care in the process of discretisation of their equations, mainly because of the fact that the density and viscosity cannot be pulled out in front of the divergence operators in both convective and diffusive terms. For that reason, the discretized equations here will be kept in the conservative form (Eqs. 3.26–3.28) and further simplification will be performed on demand in every particular simulation scenario.

3.3 Simulation of flow through porous media

Investigation of fluid flows through porous media began in the mid-19$^{\text{th}}$ century with the experimental work of French engineer Henry Darcy. A series of experiments of flow through sand columns was conducted (see Fig. 3.1), in order to establish general fluid behaviour in the filtering process and the sand bed's conductive properties. This work resulted in the well-known Darcy's Law that describes the discharge rate Q for a fluid exposed to the hydraulic head $\Delta H = H_2 - H_1$, with respect to the transport fluid properties ρ and μ, and the conductive properties: permeability κ and conductivity K of a porous medium.

$$Q = KA\frac{H_1 - H_2}{L} \;\Rightarrow\; q = -K\frac{\Delta H}{L} \quad (3.29)$$

Q	$[m^3/s]$	discharge rate	H	$[m]$	hydraulic head
p	$[Pa]$	pressure	A	$[m^2]$	cross-section of the sand column
ρ	$[kg/m^3]$	fluid density	μ	$[Pa \cdot s]$	dynamic viscosity
κ	$[m^2]$	permeability	K	$[m/s]$	hydraulic conductivity
q	$[m/s]$	specific discharge			

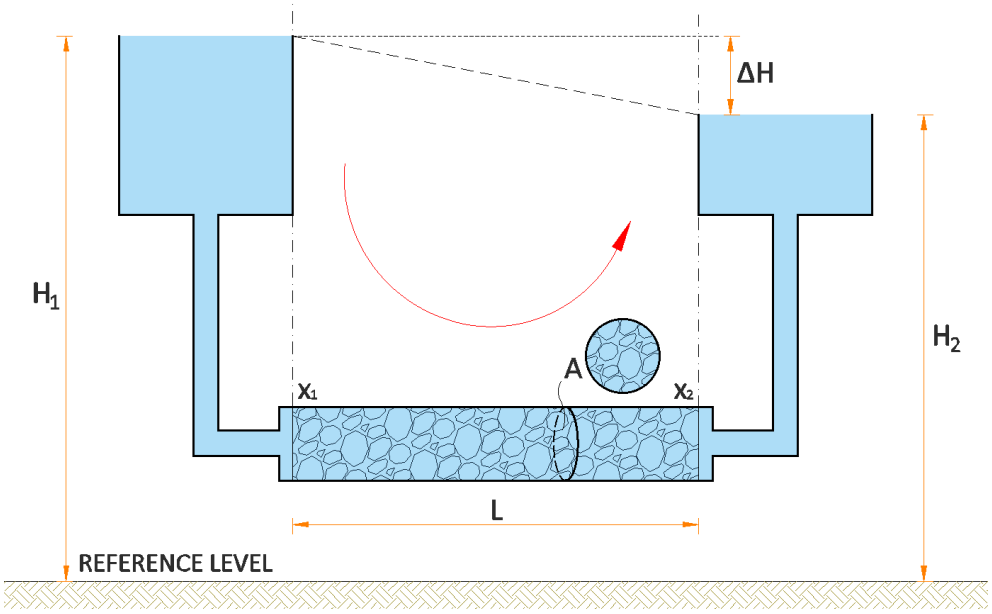


Figure 3.1: Experiment setup performed by H. Darcy in order to define a dependency between hydraulic head H, section length L, cross-section A and material properties of fluid medium, resulting in a Darcy law given in Eq. 3.29.

As illustrated in Fig. 3.1, the total hydraulic head on both sides of an experimental setup is kept constant, creating a constant, negative hydraulic head difference along the porous sample investigated. With an increase in the total head difference, the flow through the porous medium gets larger under assumption of the constant length of a sample. If the latter is shortened, the resistance of the porous medium is decreased (i.e. the water needs less effort to find the way through the pores), thus the total discharge increases accordingly. The same strategy can be applied onto the size of the cross-section area, leading to the proportional increase in the total flux with an increase in the area perpendicular to the flow direction. Out of the observed behaviours the first relation between the length, cross-section, hydraulic head difference and the total flux can be established and it reads:

$$Q\ [m^3/s] \approx A\frac{H_1 - H_2}{x_2 - x_1}\ [m^2] \tag{3.30}$$

It can be easily seen that this formulation is not dimensionally correct, thus some kind of a correlation factor with the dimension of a velocity $[m/s]$ must be introduced. This parameter is called a hydraulic conductivity K and it holds the information about the physical properties of the fluid (e.g. fluids with the larger viscosity have reduced discharge through the same porous medium), including the influence of the realistic pore cross-section area in comparison to the superficial large cross-section A, used in Eq. 3.30. In order to make this formulation more general, the hydraulic head is often expressed through the pressure difference as follows:

$$p_1 = \rho g H_1 \quad \& \quad p_2 = \rho g H_2 \tag{3.31}$$

$$Q = \frac{KA}{\rho g}\frac{p_1 - p_2}{L} = -\frac{KA}{\rho g}\nabla p \tag{3.32}$$

Finally, the hydraulic conductivity can be formulated as in Eq. 3.33 and the regrouping of the parameters can be performed, as shown in Eq. 3.35:

$$K = \frac{\kappa \rho g}{\mu} \tag{3.33}$$

$$q = Q/A \tag{3.34}$$

$$q = -\frac{\kappa}{\mu} \nabla p \tag{3.35}$$

where q $[m/s]$ is the specific discharge, also called the superficial velocity, as referred in [136], and κ, as depicted in Tab. 3.29, represents the hydraulic permeability. It can be understood as an influence of the resistance of solid porous medium exerted onto the fluid flow – an increase in the resistance is linked to the decrease in the permeability value. The unit of the permeability is $[m^2]$ or often expressed as 1 $Darcy = 10^{-12}$ m^2, honoring the scientific work of H. Darcy. Both formulations, given in Eqs. 3.29 and 3.34, describing macro-scale phenomena, are used in the coupling process with Navier–Stokes equations that relate to micro-scale occurrences (see section 4.2.1). Nevertheless, this is just one form of many possible approved formulations, and if a transient flow is to be simulated, a diffusion equation can be derived from the mass conservation law, as shown in Bear [18].
As Darcy's Law has become important in the petroleum industry, numerous extensions and reformulations have been done in order to account for the effects of two or more fluids (e.g. water and gas) impacting each other. Important work in this field has been done by M. Muskat and M. W. Meres [90] and Washburn [149]. In the case of an increased Reynolds number $R_e > 10$, the original formulation turns out to be no longer valid, and some additional terms must be included to account for non-linear inertial effects. Two widely known expressions are the Darcy–Forchheimer law and the Bosanquet equation, for the single- and multiple-phase flows, respectively.
It has been shown by S. Whitaker [130] and G. A. Narsilio et al. [52] that Darcy's Law can be derived from an incompressible Navier–Stokes formulation by means of volume-averaging (which mirrors the more rigorous process of asymptotic homogenisation [9]), if several assumptions are fulfilled: the fluid in a porous medium has a constant density and temperature; the inertial forces are negligible; and all the stresses are carried out by the porous medium itself. Instead of deriving one common model that describes both micro and macro phenomena, a coupling between multiple different scales has been performed, conducting the non-averaged NSE on the fine scale, after which Darcy's principles are applied to the results obtained, producing Darcy average quantities at the meso- and macro-scale. Further derivation details and more information on the coupling strategy applied can be found in section 4.2.1, whereas some acquired results are presented in section 6.1.

3.4 Depth-averaged 2D shallow water equations

The 2D shallow water equations (SWE), initially derived from the 3D Navier–Stokes equations (Eq. 3.26 and 3.28) by integration over the depth, are of great importance in many engineering fields where simulated waves propagate with a much larger horizontal than vertical wave length.

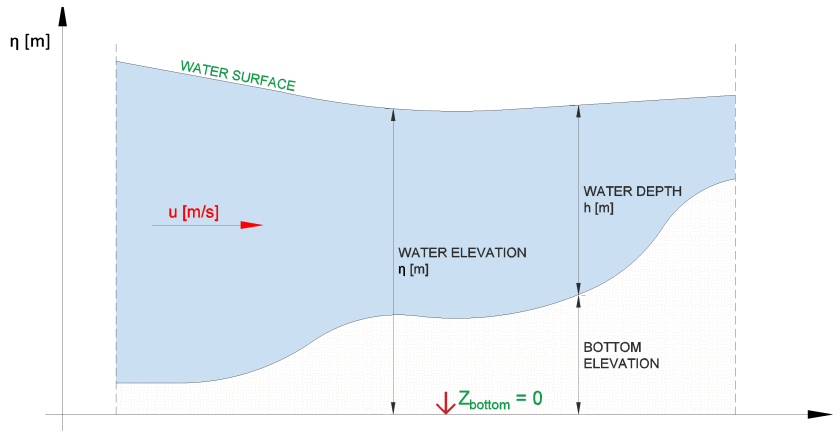


Figure 3.2: General explanation of several important parameters, frequently used in the simulation of shallow-water phenomena. Influence of surface roughness and gravity acceleration is incorporated into the model as a volume-averaged body force, as can be seen in Eq. 3.37

Such a phenomenon can occur both close to the coastline, where the waves break, and on the open sea, in so-called 'deep water'. Because of this, a simplified formulation of the 2D-SWE has been used with great success in the simulation of tsunamis, flooding events, and even atmospheric currents to some extent. As reported by Zsolt Horváth et al. [157], J. G. Zhou et al. [63], A. Kurganov and D. Levy [6] the SW method is proven to be accurate in smooth regions of flow, whereas in the vicinity of a discontinuity it introduces somewhat larger diffusion. It is also stable for both steady-state and transient flows, and if care is taken in the process of modelling the influence of the bed slope and surface friction, the solution is well balanced (i.e. the steady-state condition 'lake at rest' remains constant, if no additional impulse is added to the system).
The set of equations obtained after the integration over the depth is performed is:

$$\begin{bmatrix} h \\ hu \\ hv \end{bmatrix}_t + \begin{bmatrix} hu \\ hu^2 + \frac{1}{2}gh^2 \\ hvu \end{bmatrix}_x + \begin{bmatrix} hv \\ huv \\ hv^2 + \frac{1}{2}gh^2 \end{bmatrix}_y = \begin{bmatrix} 0 \\ -gh(S_{bx} - S_{fx}) \\ -gh(S_{by} - S_{fy}) \end{bmatrix} \tag{3.36}$$

This expression may be written concisely in its vector form as:

$$\frac{\partial \mathbf{U}}{\partial t} + F(\mathbf{U})_x + G(\mathbf{U})_y = S(\mathbf{U}) \tag{3.37}$$

where $\mathbf{U} = [h, hu, hv]^T$ represents the vector of conserved variables, F and G are flux values at homologous faces of control volume in the x and y directions, respectively, S encompasses all volumetric forces, such as those from gravitational acceleration, surface roughness and the shape of the bed. In the derived form, u and v are components of the velocity vector $\mathbf{u}$ in the orthogonal Cartesian coordinates system, h is the water depth (see Fig. 3.2), S_{bx} depicts the influence of the change of topology (Fig. 3.2, bottom elevation) and S_{fx} accounts for the resistance from friction on the bottom of the river bed. The parameter R in Eq. 3.38 is called the hydraulic radius and it contains the information about the geometrical properties of the river bed.

$$S_{bx} = \frac{\partial z_b}{\partial x}; \qquad S_{fx} = \frac{\tau}{\rho g R}; \qquad R = \frac{\text{Area of wet cross-section A}}{\text{Wet perimeter O}} \tag{3.38}$$

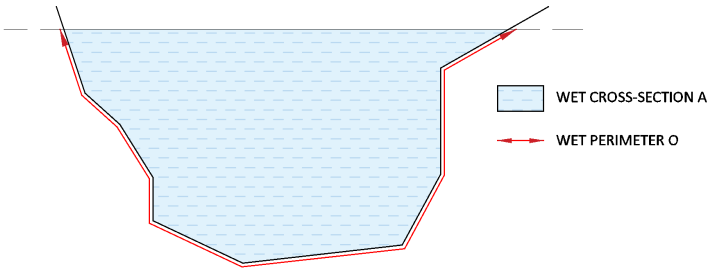


Figure 3.3: River bed cross section with two important parameters depicted: the wet cross-section area A and the wet perimeter O.

The conservative formulation depicted in Eq. 3.36 is also suitable for complex flow occurrence such as wave breaking or hydraulic jump energy dissipation, as the conservation of mass and momentum is enforced. In order to ensure all conservation principles and account for more than two different fluids (stratified flows), while respecting the limitations that arise from the initial assumptions, the Eqs.3.39–3.41 should be used.

$$\frac{\partial \rho\eta}{\partial t} + \frac{\partial \rho\eta u}{\partial x} + \frac{\partial \rho\eta v}{\partial y} = 0 \tag{3.39}$$

$$\frac{\partial \rho\eta u}{\partial t} + \frac{\partial}{\partial x}(\rho\eta u^2 + \frac{1}{2}\rho g\eta^2) + \frac{\partial \rho\eta uv}{\partial y} = 0 \tag{3.40}$$

$$\frac{\partial \rho\eta v}{\partial t} + \frac{\partial \rho\eta vu}{\partial x} + \frac{\partial}{\partial y}(\rho\eta v^2 + \frac{1}{2}\rho g\eta^2) = 0 \tag{3.41}$$

where $\eta = z_b + h$ represent the total water elevation (see Fig. 3.2). It should not be forgotten that, due to the much larger horizontal length scale in comparison to the vertical one, the pressure gradients in the vertical direction correspond to the hydrostatic pressure distribution, the velocity ω in the z-direction is considered to be small, whereas the components of the velocity vector $\mathbf{u}$ in the x and y directions are constant over the depth of fluid. Since integration over depth is performed in order to obtain the well-known form of equations (Eq. 3.36 or 3.39), the velocity component ω vanishes and cannot be computed directly. Nevertheless, if required, it can be recovered from the continuity equation. More information on multi-layer shallow water models, as well as some further elaboration on additional assumptions and restrictions that arise from the increased model complexity, followed by a set of validation data is published in Chiapolino and Saurel [27].

3.5 Non-hydrostatic 3D shallow water equations

A wide range of fluid problems can be solved effectively without significant loss of accuracy using the simplest 2D shallow water model (see previous Sec. 3.4), which is derived under the premise that the vertical component of the acceleration is rather small in comparison to the acceleration in the two horizontal directions, thus the total pressure in the flow is assumed to be hydrostatically distributed. Due to this assumption, the pressure component can often be computed explicitly, causing no problems in either the numerical stability or conservation of all the transport properties. Nevertheless, the assumption regarding the hydrostatic pressure distribution does not hold within flows where a strong gradient of the transport properties (e.g. density gradient), a large vertical variation in the viscosity and abrupt changes in the terrain topology over which the flow occurs are observed. A natural solution to resolve such flows is to use a more complex model that includes the hydrodynamic component of pressure. The full 3D Navier–Stokes model, in combination with a conservative interface reconstruction or tracking method, is proven to give reliable results, although it leads to large computational costs due to the sufficiently fine numerical discretisation required. In the absence of high-performance computing power, the latter method is limited to small simulation events, leaving no scope for a general insight into larger simulation areas, such as moving currents over the Atlantic Ocean or simulation of the influence of both the Atlantic Ocean and the Red Sea on water mixing and ecosystems in the Mediterranean Sea.

Numerous models with complexities greater than the 2D shallow water model and decreased computational cost in comparison to the 3D Navier–Stokes model have been developed to balance the accuracy and domain-size requirements. Extensive work is published by P. K. Stansby and J. G. Zhou [111] and [62] in the field of two-dimensional models, whereas the large scope of three-dimensional non-hydrostatic models, also called 3D Shallow Water (3DSW), has been investigated and published by C. Ai et al. [20], M. B. Kocyigit et al. [83], X. Liu et al. [153] and V. Casulli [139]. All the mentioned models, although sharing the same approach, differentiate themselves concerning the inclusion strategy of the hydrostatic and hydrodynamic components (e.g. explicit and implicit calculation, semi-implicit correction, implicit correction based on predicted elevation, etc.). Among all the 3DSW models, only M. B. Kocyigit et al. [83] adopted the finite difference method on a structured orthogonal mesh, whereas all others introduced a free-surface-fitting σ-coordinate system. In the MPFluid framework (see Sec. 5.1 for more details), a combination of the finite volume method and a regular orthogonal mesh has been implemented and validated using the available analytical solutions.

The governing equations of the 3D Shallow Water model are derived from the original NSE formulation depicted in Eqs. 3.26–3.28, incorporating the basic concept of separation of the total pressure influence p_t into its hydrodynamic p_{dyn} and hydrostatic p_{st} components.

$$p_t = p_{st} + p_{dyn} = \rho g_z \eta + p_{dyn} \tag{3.42}$$

$$\frac{\partial \rho u}{\partial t} + \frac{\partial \rho u u}{\partial x} + \frac{\partial \rho u v}{\partial y} + \frac{\partial \rho u w}{\partial z} = -\frac{\partial p_t}{\partial x} + \frac{\partial \tau_{xx}}{\partial x} + \frac{\partial \tau_{xy}}{\partial y} + \frac{\partial \tau_{xz}}{\partial z} + \rho g_x \tag{3.43}$$

$$\frac{\partial \rho v}{\partial t} + \frac{\partial \rho v u}{\partial x} + \frac{\partial \rho v v}{\partial y} + \frac{\partial \rho v w}{\partial z} = -\frac{\partial p_t}{\partial y} + \frac{\partial \tau_{yx}}{\partial x} + \frac{\partial \tau_{yy}}{\partial y} + \frac{\partial \tau_{yz}}{\partial z} + \rho g_y \tag{3.44}$$

$$\frac{\partial \rho w}{\partial t} + \frac{\partial \rho w u}{\partial x} + \frac{\partial \rho w v}{\partial y} + \frac{\partial \rho w w}{\partial z} = -\frac{\partial p_t}{\partial z} + \frac{\partial \tau_{zx}}{\partial x} + \frac{\partial \tau_{zy}}{\partial y} + \frac{\partial \tau_{zz}}{\partial z} + \rho g_z \tag{3.45}$$

$$\frac{\partial \eta}{\partial t} + \frac{\partial}{\partial x}\int_0^\eta u dz + \frac{\partial}{\partial y}\int_0^\eta v dz = 0 \tag{3.46}$$

where τ_{ij} represents the deviatoric part of the total stress and η the total water elevation measured from the topographic bottom to the water free-surface.
The position of the free-surface interface in the following time step $n+1$ is obtained by discretising Eq. 3.46, where a simple explicit formulation is reported to produce neither convergence, nor stability issues. In case of more complex topographic terrain, such difficulties might eventually occur generating non-physical or negative elevation values, thus an implicit formulation must be used, as suggested by M. B. Kocyigit et al. [83]. Plugging Eq. 3.42 into Eqs. 3.43–3.45 and rearranging the pressure and gravity terms, the final form implemented is obtained:

$$\frac{\partial \rho u}{\partial t} + \frac{\partial \rho u u}{\partial x} + \frac{\partial \rho u v}{\partial y} + \frac{\partial \rho u w}{\partial z} = -g_z \frac{\partial \rho \eta}{\partial x} - \frac{\partial p_{dyn}}{\partial x} + \frac{\partial \tau_{xx}}{\partial x} + \frac{\partial \tau_{xy}}{\partial y} + \frac{\partial \tau_{xz}}{\partial z} + \rho g_x \tag{3.47}$$

$$\frac{\partial \rho v}{\partial t} + \frac{\partial \rho v u}{\partial x} + \frac{\partial \rho v v}{\partial y} + \frac{\partial \rho v w}{\partial z} = -g_z \frac{\partial \rho \eta}{\partial y} - \frac{\partial p_{dyn}}{\partial y} + \frac{\partial \tau_{yx}}{\partial x} + \frac{\partial \tau_{yy}}{\partial y} + \frac{\partial \tau_{yz}}{\partial z} + \rho g_y \tag{3.48}$$

$$\frac{\partial \rho w}{\partial t} + \frac{\partial \rho w u}{\partial x} + \frac{\partial \rho w v}{\partial y} + \frac{\partial \rho w w}{\partial z} = -\frac{\partial p_{dyn}}{\partial z} + \frac{\partial \tau_{zx}}{\partial x} + \frac{\partial \tau_{zy}}{\partial y} + \frac{\partial \tau_{zz}}{\partial z} \tag{3.49}$$

As shown in Eqs. 3.47 and 3.48, the total elevation η, calculated in Eq. 3.46, is incorporated in the Navier–Stokes momentum equations, which are solved using the fractional step method, as explained in detail in Sec. 4.1.2. In that way, the solution should preserve the divergence-free velocity field, hence satisfy the mass and momentum conservation principles. The time to solution is reduced to 60% compared to the time to solution for the full 3D Navier–Stokes model applied to the identical simulation setup, due to the initial assumption that the horizontal length scale is much larger than the vertical one, thus a dominant hydrostatic component is computed and immediately included in the prediction step. The less pronounced hydrodynamic component is then calculated using fewer multigrid steps while solving the pressure Poisson equation (see Sec. 5.2 for more details) and included implicitly within the correction step. Nevertheless, this assumption reduces the complexity of the setups that can be handled by this model, thus more complicated scenarios with the fluid separation or merging cannot be

accurately dealt with using 3D shallow water models. For less complex simulation setups these models are very well suited. An exact how-to implementation procedure is given in Sec. 4.2.2. For interested user, some further possibilities are suggested also by C. Ai et al. [20] and V. Casulli [139].

3.6 Interface recognition methods within two-phase flows

The numerical simulation of two-phase flows has been frequently utilised in many engineering fields, such as chemical mixing processes, motor combustion, heat-transfer-driven flows, large flooding events and various environmental and atmospheric analysis, to name but a few. In almost all application setups a sudden change in the material properties (e.g. density and viscosity) occurs at the interface between the two examined fluids, leading to an abrupt jump in the pressure and strong but smooth gradients in the velocity field. Non-satisfactory interface resolution or an incorrect interface representation leads to a strong local non-physical deviation of both the pressure and velocity fields, altering the global simulation result often very severely, even if overall stability has been achieved. For this reason, a precise interface representation has been focused on ever since early simulation attempts and different approaches have been developed to control such local anomalies. Among all the various methods, two major branches have been established, dividing the scientific community into two parties: namely proponents of 'interface-tracking' and of 'interface reconstruction' methods. Interface-tracking methods, also called front-tracking methods (FTM) take advantage of the Lagrangian approach, conforming the mesh to the interface, while at the same time flow simulation is done utilising the Eulerian approach. This method leads to minimal errors in the interface representation and to promising simulation results; nevertheless, proper matching between the static and moving mesh causes major problems as the description of the simulation domain gains complexity. Significant work in this field has been carried out by G. Tryggvason et al. [55] and N. G. Deen et al. [103]. Interface reconstruction methods (IRM), in contrast to FTM, are less accurate due to taking a purely Eulerian approach, where the dynamically changing interface must be reconstructed at every time advancement, using the same mesh description as the one used for the flow simulation itself. Due to its implementation simplicity, many different subroutines have been developed to reduce these reconstruction errors, thus nowadays a successful integration has been reported in the volume-of-fluid approach by C. W. Hirt and B. D. Nichols [25], O. Ubbink and R. I. Issa [107], M. Malik et al. [89], W. Aniszewski et al. [146], V. Coralic and T. Colonius [140] and A. Pathak and M. Raessi [7]; in the level-set (LS) approach by S. Osher and J. A. Sethian [125] and M. Sussman et al. [92]; in the VOSET method by D. L. Sun and W. Q. Tao [31], Z. Wang et al. [156] and M. Sussman [91], and finally in the combined volume-of-fluid and conservative level-set approach (VOFCLS) by E. Olsson et al. [36], M. Sussman et al. [94] and N. Balcázar et al. [101].
For the sake of completeness, a short overview of the classical volume-of-fluid (VOF) and level-set (LS) methods will be given in the next sections, while full attention will be given to the conservative level-set (CLS) method. Although all three methods have been implemented in this work, the most promising results are obtained using CLS.

3.6.1 Volume-of-fluid (VOF) method

The volume-of-fluid (VOF) approach is the best-known sharp-interface reconstruction method, within which an interface between two fluids is defined in terms of the fraction of the cell's volume that belongs to one fluid phase $\forall_{ijk}$ (see Eq. 3.50). This implicit representation is done through the color function χ that is advected in every time step, solving an advection partial differential equation (Eq. 3.51).

$$\forall_{ijk} = \frac{1}{V_\Omega}\int_0^{\Delta z}\int_0^{\Delta y}\int_0^{\Delta x}\chi(x,y,z)\mathbf{d}V_\Omega = \begin{cases} 1, & \text{cell filled with fluid 100\%} \\ (0,1), & \text{partially filled with the fluid phase} \\ 0, & \text{empty cell} \end{cases} \tag{3.50}$$

where χ is the colour function and V_Ω the volume of the cell, over which integration of the colour function has been performed.

$$\frac{\partial\chi}{\partial t} + \frac{\partial\chi u}{\partial x} + \frac{\partial\chi v}{\partial y} + \frac{\partial\chi w}{\partial z} = 0 \tag{3.51}$$

In order to express Eq. 3.51 in terms of the cell-volume fraction, Eq. 3.50 is used and integration of the volume is performed using Gauss' theorem. This procedure results in the final formulation, which will be used in the further discretisation process (see Eq. 3.52).

$$\forall_{ijk}^{n+1} = \forall_{ijk}^{n} + \frac{\Delta t}{V_\Omega}\sum_0^{F_c} F_{ijk}A_c, \tag{3.52}$$

with F_{ijk} being the flux at the corresponding cell face F_c, and A_c the normal projected area of that particular face.

Once the value of the volume fraction $\forall$ is available, a geometric reconstruction (i.e. WLIC, SLIC, PLIC, THINC, LSM, etc.) must be done (see Fig. 3.4), in order to address the mass and momentum fluxes at the cell faces. A good overview of several commonly used reconstruction methods is given by W. Aniszewski et al. [146], including a comparison of the total mass loss

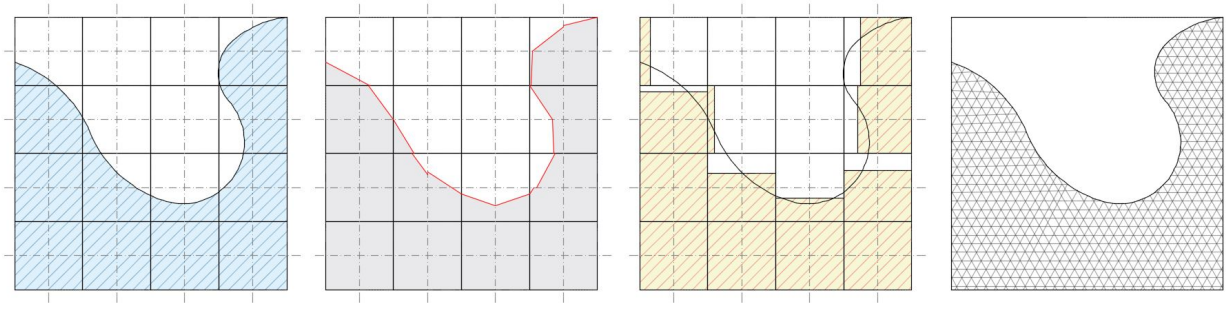


Figure 3.4: Three different interface reconstruction methods applied at the original smooth solution (the leftmost). The second subplot represents the piecewise linear reconstruction (PLIC), followed by the simple linear interface calculation (SLIC) and the front-tracking semi-Lagrangian method (FRM).

(using L_1 norm) in several of the best-known validation setups, such as '3D passive advection', 'static droplet test' and 'oscillating droplet', among others.

VOF methods generally have good mass conservation and are suitable for HPC parallel implementation, yet the reconstruction process is not unique, producing somewhat algorithm-dependent simulation results.

3.6.2 Level-set (LS) method

Unlike volume-of-fluid methods, where a sharp interface is obtained by a simple mapping of the colour function to the percentage of the cell-volume occupied by one of the fluid phases, the standard level-set method (SLS) uses a signed distance function Φ, which contains the information about the distance from the arbitrary point x under consideration to the fluid interface **I**:

$$\Phi(x) = d^{\perp}(x) = \mathbf{sign}\left[\min(\vec{\mathbf{x}} - \vec{\mathbf{x_I}})\right] \cdot \min(|\vec{\mathbf{x}} - \vec{\mathbf{x_I}}|), \tag{3.53}$$

with $\vec{\mathbf{x}}$ being the position vector of the corresponding cell and $\vec{\mathbf{x_I}}$ the position vector of the interface between two fluids. In case of more complex interface definition, multiple perpendicular vectors $|\vec{\mathbf{x}} - \vec{\mathbf{x_I}}|$ can be calculated, thus it is important to calculate a global minimum and assign only this value to the signed distance function. A detailed procedure on how to do that is given in Sec. 2.2.3. Having defined that, the function Φ will be positive $\Phi > 0$ on one side of the interface (fluid phase 1) and negative $\Phi < 0$ on the other side (fluid phase 2). The value $\Phi = 0$ represents the position of the interface **I**. In order to follow a transformation of the interface through time, the signed distance function Φ must be advected by solving the advection:

$$\frac{\partial \Phi}{\partial t} + \mathbf{u}\frac{\partial \Phi}{\partial x} = 0 \tag{3.54}$$

An incompressibility constraint in the advection process is imposed by the introduction of a divergence-free velocity field, which transforms Eq. 3.54 into its conservative form:

$$\frac{\partial \Phi}{\partial t} + \nabla \cdot (\mathbf{u}\Phi) = 0 \tag{3.55}$$

In order to address a sharp change in the transport and material properties at the interface, the Heaviside function H is introduced as follows:

$$H(\Phi) = \begin{cases} 1, & \Phi \geq 0 \\ 0, & \Phi < 0 \end{cases} \tag{3.56}$$

leading to the straightforward definition of the values transported (see Eqs. 3.57 and 3.58)

$$\rho = H(\Phi)\rho_f + (1 - H(\Phi))\rho_g \tag{3.57}$$

$$\mu = H(\Phi)\mu_f + (1 - H(\Phi))\mu_g \tag{3.58}$$

Due to discontinuous nature of the Heaviside function H, density and viscosity remain sharp and discontinuous all the time, which, while physically correct, can introduce some numerical stability issues. To maintain numerical robustness, E. Olsson and G. Kreiss [35] proposed an approximation to the Heaviside function, which preserves the initial density and viscosity values, away from the interface, and generate a smooth transition of those properties close to the material border.

$$H(\Phi) = \begin{cases} 1, & \Phi < -\varepsilon \\ \frac{1}{2} + \frac{\Phi}{2\varepsilon} + \frac{1}{2\pi} sin(\frac{\pi\Phi}{\varepsilon}), & -\varepsilon \leq \Phi \geq \varepsilon \\ 0, & \Phi > \varepsilon \end{cases} \tag{3.59}$$

where ε represents the thickness of the interface region over which the smooth transition is enforced. This value ε should be defined as a function of the mesh size, resulting in a reduced smeared area in the better-resolved mesh regions.

Despite the application of a conservative formulation (see Eq. 3.55), the property of the signed distance function is lost with time (the advected value Φ does not represent the normal shortest distance to the interface); thus to achieve global mass conservation, the function Φ must be reinitialised from time to time. If nothing is done, the interface region deforms due to the local addition and subtraction of mass, causing spurious pressure and velocity. This issue is a well-known disadvantage of the standard LS method, which has been addressed already in the early period of the development, and since then many different techniques have been introduced to overcome this problem. Successful implementations of level-set reinitialisation are reported by E. Olsson and G. Kreiss [35], M. Sussman et al. [92] and N. Balcázar et al. [101]. Moreover, the level-set reinitialisation combined with a reinitialisation of the density function, as reported by R. K. Shukla et al. [118], decreases the size of the transition region, leading to a better accuracy, while the numerical robustness has been kept unchanged.
In this work, the idea of E. Olsson and G. Kreiss [35] has been followed and implemented with several changes published by N. Balcázar et al. [101]. All the implementation details of this conservative level-set method are given in the next section.

3.6.3 Conservative Level-Set (CLS) method

As already elaborated in Sec. 3.6.2, the most important requirements imposed in the advection process of any level-set function are: mass conservation, avoidance of spurious currents and the constant thickness ε of the interface profile. The conservative level-set method (CLS) is developed as a natural extension of the standard LS method, in order to tackle these important constraints. Unlike S. Osher and J. A. Sethian [125], who defined the level-set function Φ as a signed distance function (see Eq. 3.53), the inventors and proponents of the CLS method define Φ as a regularised indicator function in the following form:

$$\Phi(x,t) = \frac{1}{2}\left[\tanh\left(\frac{d(x,t)}{2\varepsilon}\right) + 1\right] \tag{3.60}$$

where $d(x,t)$ represents the signed distance function used in the standard LS method and ε the thickness of the interface profile. As schematically shown in Fig. 2.12, the signed distance

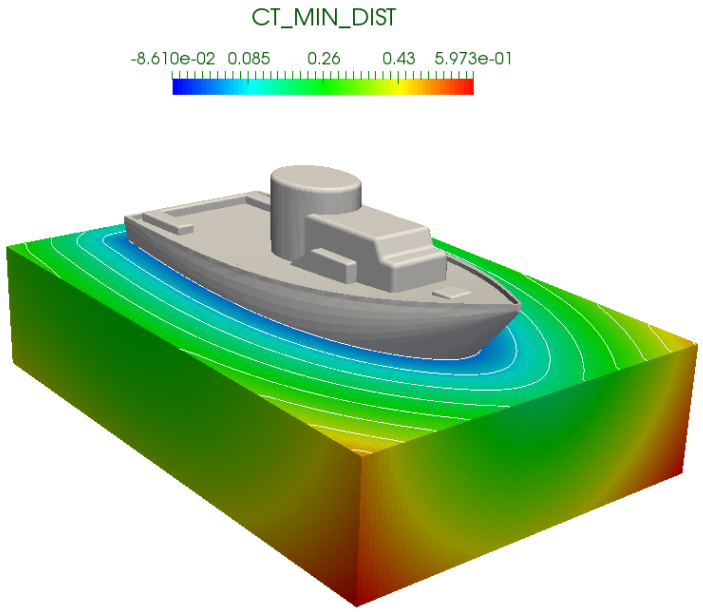


Figure 3.5: Insertion of a complex geometry representation (STL Boat Datafile contains 5862 triangular elements). The cumulative minimum distance function is calculated as explained in Sec. 2.2.3.

function $d(x,t)$ must be calculated taking into consideration every fluid surface existing within the numerical domain. While it is very simple to identify the projection of a point onto a single surface, the entire process is rather intricate, as additional arbitrarily positioned surfaces are involved. For the sake of simplicity, the standard STL surface representation has been used throughout this work, and the whole insertion procedure that includes multiple water bodies is described in detail in section 2.2.3.

As can been seen in Fig. 3.5, within that procedure an arbitrarily complex minimum distance function can be calculated, which is substituted into Eq. 3.60, resulting in an indicator function Φ that takes the value 1 in the fluid region and the value 0 in the gaseous region. The exact position of the interface $\mathbf{I}$ correlates to a value of $\Phi = 0.5$. Such an initialised indicator function is advected using the interface velocity $\mathbf{u}$ as part of the advection process given in Eq. 3.55.

In order to meet the three previously stated requirements, several different spatial and temporal schemes are employed and combined in the advection process. The temporal term is discretised using one of the following schemes: first-order accurate Euler scheme, second-order accurate Adams–Bashforth scheme or Total-Variation-Diminishing second- and third-order accurate Runge–Kutta schemes (TVD-RK2 and TVD-RK3) (see Tab. 3.1). For the sake of completeness, theoretical background and practical considerations of all spatial and temporal schemes are given in Secs. 4.1.6 and 4.1.7.

The convective term in Eq. 3.55 is discretised using the first-order central-difference non-TVD scheme, first-order upwind scheme, second-order upwind scheme with the Minmod, Van-Leer and Superbee limiters. As is reported by E. Olsson and G. Kreiss [35], the upwind scheme with the Superbee limiter produces the most convenient results. All other schemes lead to highly oscillatory behaviour or smearing of the interface profile, which causes spurious currents,

$\Phi^{n+1} = \Phi^n + \Delta t\mathcal{L}(\Phi^n, t^n)$	first-order Euler scheme
$\Phi^{n+1} = \Phi^n + \frac{3}{2}\Delta t\mathcal{L}(\Phi^n, t^n) - \frac{1}{2}\Delta t\mathcal{L}(\Phi^{n-1}, t^{n-1})$	second-order Adams–Bashforth scheme
$\Phi^* = \Phi^n + \Delta t\mathcal{L}(\Phi^n)$ $\Phi^{n+1} = \frac{1}{2}\Phi^n + \frac{1}{2}\Phi^* + \Delta t\mathcal{L}(\Phi^*)$ $\Phi^{n+1} = \Phi^n + \frac{1}{2}\Delta t\mathcal{L}(\Phi^n) + \frac{1}{2}\Delta t\mathcal{L}(\Phi^*)$	Total-Variation-Diminishing Runge–Kutta 2
$\Phi^* = \Phi^n + \Delta t\mathcal{L}(\Phi^n)$ $\Phi^{**} = \frac{3}{4}\Phi^n + \frac{1}{4}\Phi^* + \frac{1}{4}\Delta t\mathcal{L}(\Phi^*)$ $\Phi^{n+1} = \frac{1}{3}\Phi^n + \frac{2}{3}\Phi^{**} + \frac{2}{3}\Delta t\mathcal{L}(\Phi^{**})$	Total-Variation-Diminishing Runge–Kutta 3
$\mathcal{L}(\Phi^n, t^n) = \nabla \cdot (\mathbf{u}^n\Phi^n)$ – convective operator (see Eq. 3.54)	

Table 3.1: Four different temporal schemes used in the advection process. The cost–benefit analysis in terms of the expected stability depicted the TVD-RK2 as the most efficient scheme in the tested cases. The increase in numerical stability does not correspond to the increase in computational cost, if the TVD RK3 is employed.

FLUX CALCULATION	SPATIAL SCHEME
$\sum_{i=0}^{i=6} \frac{\Phi_i^{nb}}{\Delta h_i^{\perp}} - 2\Phi_c \sum_{j=0}^{j=3} \frac{1}{\Delta h_i}$	first-order CD scheme
$\sum_i \max(\mathbf{u_f}, 0)\Phi_c + \min(\mathbf{u_f}, 0)\Phi_{nb}$	first-order upwind scheme
$\sum_i \max(\mathbf{u_f}, 0)\Phi_c + \min(\mathbf{u_f}, 0)\Phi_{nb} + \frac{1}{2}\mathbf{u_f}L(\theta)(\Phi_{nb} - \Phi_c)$ $L(\theta) = \max(0, \min(1, \theta))$	second-order upwind scheme with Minmod limiter
$\sum_i \max(\mathbf{u_f}, 0)\Phi_c + \min(\mathbf{u_f}, 0)\Phi_{nb} + \frac{1}{2}\mathbf{u_f}L(\theta)(\Phi_{nb} - \Phi_c)$ $L(\theta) = \frac{\theta+\lvert\theta\rvert}{1+\lvert\theta\rvert}$	second-order upwind scheme with Van Leer limiter
$\sum_i \max(\mathbf{u_f}, 0)\Phi_c + \min(\mathbf{u_f}, 0)\Phi_{nb} + \frac{1}{2}\mathbf{u_f}L(\theta)(\Phi_{nb} - \Phi_c)$ $L(\theta) = \min(0, \max(2\theta, 1), \min(2, \theta))$	second-order upwind scheme with Superbee limiter

Table 3.2: Five different spatial schemes used for the discretisation of the convective terms in the advection (Eq. 3.55) and reinitialisation (Eq. 3.62) equations.

local instability and, in the worse case, global divergence. An overview of the spatial schemes integrated is given in Tab. 3.2, where the parameter θ represents the ratio between upwind and downwind influence of the velocity components and it is mathematically formulated as in Eq. 3.61. The upwind and downwind neighbours are set dynamically for every cell face, depending on the scheme used. The entire 'how-to' procedure is thoroughly explained and depicted in section 4.1.6, while several detailed analysis of an influence of both spatial and temporal schemes on the advection process are conducted and illustrated in Secs. 3.6.3.1–3.6.3.4.

$$\theta = \frac{\mathbf{u_c} - \mathbf{u_{nb}}^{upwind}}{\mathbf{u_{nb}}^{downwind} - \mathbf{u_c}} \tag{3.61}$$

In order to avoid the spurious currents previously mentioned, it is necessary to keep the thickness of the interface profile constant over time, and this has been achieved by adding a small portion of viscosity into the system, which is balanced by a small portion of the compression on the other side, leaving the initial system in an unchanged state.
The mathematical formulation of this, also called the reinitialisation equation, is given in Eq. 3.62:

$$\frac{\partial \Phi}{\partial \tau} + \nabla \cdot [\Phi(1-\Phi)n_\tau] = \nabla \cdot \varepsilon \nabla \Phi \tag{3.62}$$

where the indicator function Φ is advected in pseudo time τ, the term on the right-hand side represents the additional portion of viscosity, resulting in the constant thickness ε of the interface profile and a smooth transition from one fluid phase to another, while the second term on the left-hand side introduces compression, aimed at reducing the non-necessary dissipation, thus sharpening the interface profile. The normal vector:

$$n_\tau = \nabla \Phi / ||\nabla \Phi||$$

is calculated once at the beginning of the reinitialisation process, based on the available indicator function values, and is kept constant during the entire reinitialisation. As in the case of the advection equation (Eq. 3.55), the temporal term can be discretised using one of the four temporal schemes given in Tab. 3.1, whereas the discretisation of the compression and diffusion terms must be done with care, aimed at minimising the tangential deformation introduced numerically. In order to quantify this tangential deformation, five different approaches have been implemented, followed by various test cases. The first approach is proposed by K. K. So et al. [72], and further exploited and modified by V.-T. Nguyen and W.-G. Park [142]. Both the original and alternative formulations preserve the mass in the volume-of-fluid approach, with a difference being that in [72] only one so-called anti-diffusion term (see Eq. 3.63, right-hand side) is used, while the compression term is indirectly included within the VOF OpenFOAM implementation.

$$\frac{\partial \Phi}{\partial \tau} = -\nabla \cdot \nabla \Phi \tag{3.63}$$

In order to ensure boundedness and monotonicity, a Minmod slope limiter is used in the estimation of the gradient of the level-set function at the cell face, combined with the upwind approach for estimation of the divergence operator, once the left and right gradient values are determined. This approach is, as is confirmed by V.-T. Nguyen and W.-G. Park [142], very efficient in multi-dimensional, irregular mesh settings. The implementation in [142] recovers the lost compression term and discusses this approach in a more general sense, independent of the OpenFoam implementation.

$$\frac{\partial \Phi}{\partial \tau} = \nabla \cdot [\Phi(\Phi - 1) - \varepsilon \nabla \Phi] \tag{3.64}$$

where ε represents the desired thickness of the interface profile between two fluids. Moreover, in [142] the formulation of the anti-diffusion term is adopted from E. Olsson et al. [36], allowing only the influence of the normal anti-diffusion correction.

Both authors used the standard validation tests of the static drop, a rotational Zalesak disc and a solenoidal velocity field, and reported losses of up to two percent of the mass. Within the implementation of these methods in our own framework, we can confirm a mass inconsistency varying from 1.5% to 2.4% in the exact same validation tests. Nevertheless, when the tests are simulated for a longer period, the loss of mass does not converge to zero, thus some longer simulation scenarios (e.g. a wave moving back and forth between two solid walls until it reaches a steady state) result in a non-negligible drop in mass. For this reason, this approach is abandoned and will not be further discussed.

The foundation of the nowadays commonly used, conservative level-set method is established by E. Olsson and G. Kreiss [35], further improved by the same author in [36] and later used and modified by many authors, among whom [118], [101] and [134] reported the largest improvements for a general, unstructured, adaptive mesh representation. For that reason, all four methods are briefly explained in the further text and used for validation purposes.

The original formulation of CLS given in Eq. 3.62 guarantees an unchanged state of all the conserved properties, despite the addition of small amounts of compression and diffusion. As stated by the authors themselves in [36], a normal component of the compression balances out the normal component of the diffusion added to the system and this process converges in as few as three to four pseudo iterations depending on the size of the pseudo time step. Nevertheless, in such problem cases where surface tension plays a significant role, diffusion can cause a spurious level-set flux in a direction tangential to the interface. This behaviour is especially accentuated when larger pseudo time steps τ and increased number of iterations to convergence are needed. To handle this problem E. Olsson et al. [36] suggested the exclusion of the tangential diffusion component by projecting the total diffusion added onto the interface normal vector, as follows:

$$\frac{\partial \Phi}{\partial \tau} + \nabla \cdot [\Phi(1-\Phi)n_\tau] = \nabla \cdot [\varepsilon \left(|\nabla \Phi| \cdot n_\tau\right) n_\tau] \tag{3.65}$$

To take into consideration adaptively refined regions, the thickness of the interface profile should ideally depend on the local mesh size, whereas the size of the pseudo time step should depend on both the mesh size and the thickness of the profile.

$$\varepsilon = \frac{\delta}{ln(0.95^2/0.05^2)} \approx \frac{1}{6}\delta \tag{3.66}$$

$$Tolerance = \frac{0.2\delta}{m^2} \tag{3.67}$$

with δ being the interface region and m the number of mesh points belonging to that region. To enhance the numerical stability, a general recommendation is that the interface region δ should have the width of three to four computational cells, which leads to approximate values

of ε and τ as given below (see Eqs. 3.68 and 3.69):

$$\varepsilon = \frac{1}{2}\Delta x^{\alpha}, \quad \alpha = [0.90, 1] \tag{3.68}$$

$$\tau \leq C_{olsson}\frac{\Delta x^2}{\varepsilon} \tag{3.69}$$

The parameters C_{olsson} and α must be calibrated for each particular model; nevertheless, it is suggested that the values $C_{olsson} = 0.25$ and $\alpha = 0.95$ lead to a robust, stable scheme.
The approach depicted in N. Balcázar et al. [101] is based on the earlier formulation of E. Olsson and G. Kreiss, taking into consideration both the normal and tangential components of the diffusion. However, this limits the number of iterations up to ten, in order to control the appearance of spurious currents, thus speed of convergence. Both the advection and reinitialization equations are discretised using the third-order Runge–Kutta Total-Variation-Diminishing scheme in time, first published by S. Gottlieb and C.-W. Shu [123], while the second-order Superbee limited scheme in space is employed in the advection process and the second-order central-difference limited scheme (see Eqs. 3.70–3.73) for reinitialisation purposes. A detailed description of all schemes used within this work is given in Secs. 4.1.6 and 4.1.7.

$$\nabla \cdot [\Phi(1-\Phi)n_{\tau=0}] = \sum_{f=0}^{f=N_f} max(\mathbf{U_f}, 0)\Phi_c(1-\Phi)_c + min(\mathbf{U_f}, 0)\Phi_{nb}(1-\Phi)_{nb} \tag{3.70}$$

$$+ \sum_{f=0}^{f=N_f} \frac{1}{2}\mathbf{U_f}\left[\Phi_{nb}(1-\Phi)_{nb} - \Phi_c(1-\Phi)_c\right] \tag{3.71}$$

$$\mathbf{U_f} = \mathbf{n_f^{\tau=0}} A_f, \tag{3.72}$$

$$\mathbf{n}_{\tau=0} = \text{const.} \tag{3.73}$$

where $\mathbf{U_f}$ represents the flux at cell face f with its normal vector $\mathbf{n_f}$ pointing outwards and face area A_f, N_f is a total number of faces belonging to a cell and, $\mathbf{n}_{\tau=0}$ is the normal vector of the interface at the pseudo time step $\tau = 0$. Indices c and nb represent current and neighbouring cells, respectively, thus Φ_c and Φ_{nb} are the level-set values stored in the current and neighbouring cell-centres.
In contrast to the four already mentioned authors, who established the influence of different spatial and temporal schemes on the acclaimed formulation for the advection and reinitialisation process, the work of R. K. Shukla et al. [118] is based on a reformulation of the level-set property. The main assumption of this approach is that the level-set representation, as originally defined, may still have an excessively sharp interface, where a direct correlation between the normal vector, curvature of the interface and level-set variable is established. The authors propose the introduction of yet another smoother variable θ across the interface, which will be directly correlated to the level-set value and have an identical normal vector estimation. At the same time it delivers far better approximations of the gradient and divergence operator applied in

the reinitialisation process.

$$\theta = \frac{\Phi^{\alpha}}{\Phi^{\alpha} + (1-\Phi)^{\alpha}} \quad \text{for} \quad \alpha < 1 \tag{3.74}$$

Setting a calibration factor α too close to a value of 1 will equiponderate the level-set and smooth θ function, losing all the advantages previously stated. Knowing the fact that the width of the hyperbolic tangent profile of the level-set function δ_{Φ} is correlated to the width of the tangent profile of the smooth function, as follows: $\frac{1}{\alpha}\delta_{\Phi} = \delta_{\theta}$, the choice of too small values of α may yield an inability to resolve the steep level-set gradients accurately.

The derivation of the smooth θ function in Eq. 3.74 can be easily done using the product rule, resulting in the following formulation:

$$\nabla\theta = \frac{\nabla\Phi}{\alpha}\left\{[\Phi(1-\Phi)]^{1-\alpha}\left[\Phi^{\alpha} + (1-\Phi)^{\alpha}\right]^{2}\right\} \tag{3.75}$$

Having at hand the gradient of both the theta and level-set function, the normal vector at the interface between two fluids, pointing outwards, can be calculated as $\mathbf{n} = \frac{\nabla\theta}{|\nabla\theta|} = \frac{\nabla\Phi}{|\nabla\Phi|}$. As is the case in [36], the normal vector is calculated only once per time step and is kept fixed during the rest of the reinitialization process. Within this approach, a new definition of the reinitialisation is suggested:

$$\frac{\partial\Phi}{\partial\tau} = \mathbf{n}\cdot\nabla\left[\varepsilon|\nabla\Phi| - \Phi(1-\Phi)\right] \tag{3.76}$$

as the formulation in [36] may lead to appearance of numerical oscillations, strongly influenced by the accuracy of the interface curvature estimation. The second novelty in R. K. Shukla et al. [118] is a separation of the level-set function from the definition of the transport properties (e.g. density), which is reported to be even more efficient, but it requires a modification in the coupling strategy to the Navier–Stokes system of equations. The latest extension is not examined and used within this work.

The work of T. Wacławczyk [134] is based on the work of R. K. Shukla et al. [118] and represents a further mathematical discussion of constraints, set in that work. Moreover, an improved smooth function θ is proposed:

$$\theta_{wt} = \frac{(\Phi+\xi)^{\alpha}}{(\Phi+\xi)^{\alpha} + (1-\Phi+\xi)^{\alpha}} \tag{3.77}$$

where ξ represents a small value, which is added to prevent the discontinuities in the smooth function, in the limit case when $\Phi \to 0$ and $\Phi \to 1$. Furthermore, it is proven that a new variable determined Φ_{wt}, based on the modified Φ function:

$$\Phi_{wt} = \frac{2\varepsilon}{\alpha}\left(2\theta_{wt} - 1\right) \tag{3.78}$$

has the exact same properties as the signed distance function, used in the standard level-set approach – the magnitude of a gradient of the signed distance function away from the

discontinuity equals to 1, $|\nabla\Phi_{wt}| = 1$. Substituting the expression in Eq. 3.78 into Eq. 3.76, leads to the reformulated reinitialisation equation:

$$\frac{\partial\Phi}{\partial\tau} = \nabla\cdot[\delta(\Phi)(|\Phi_{wt}| - 1)\mathbf{n}_{\tau=\mathbf{0}}] \tag{3.79}$$

where $\delta(\Phi) = \Phi(1-\Phi)$. A careful observation of Eq. 3.79 suggests that the reinitialization process takes place only in the close vicinity of a discontinuous interface, whereas the expression $|\Phi_{wt}| - 1$ tends to zero in every cell occupied only by one fluid phase. In Sec. 3.6.3.1, it is shown that this approach, as well as the approach from [118], offers a qualitatively better approximation of the gradient and divergence operators compared to the approach from [35] and least-square-fitting gradient reconstruction. This results in a slightly better estimation of the interface curvature $\kappa = -\nabla\cdot\left(\frac{\nabla\Phi}{|\nabla\Phi|}\right)$. According to the newer publication of T. Wacławczyk [135], this reinitialisation methodology gives the best results, if combined with a level-set advection Lagrangian scheme, otherwise, it might be necessary to iterate longer to achieve the convergence. For this reason, within this work the proposition on normal vector and calculation of the interface curvature is adopted from this source, whereas the reinitialization process from E. Olsson and G. Kreiss (2004) is applied and further modified for the reasons explained in Sec. 3.6.3.3.

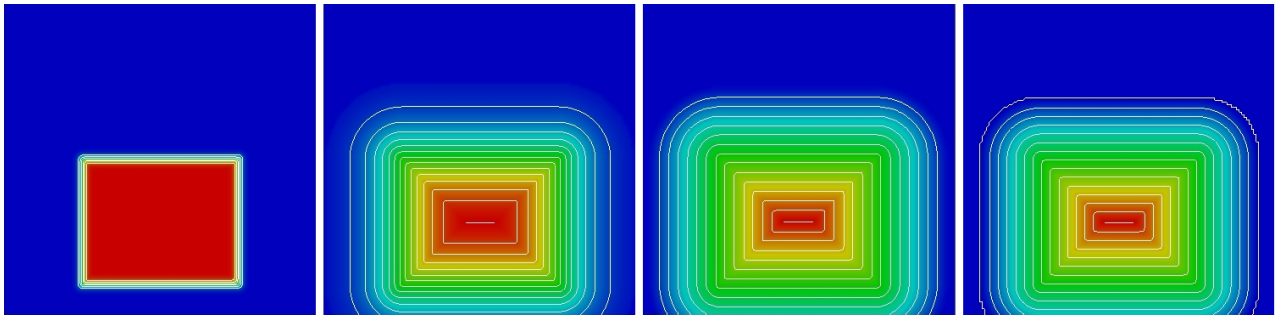


Figure 3.6: Comparison of the interface smooth functions - level set representation, utilised in N. Balcázar et al. [101], smooth theta function introduced by R. K. Shukla et al. [118], modified theta function used by T. Wacławczyk [134] and signed-distance function used in E. Olsson and G. Kreiss [35]; E. Olsson et al. [36]. The least error prone normal vector calculation is performed using the third approach based on the modified Wacławczyk function, whereas the signed-distance function strategy results in the smooth solution in the entire domain except in the narrow region around the level-set values $\Phi = 0.0$ and $\Phi = 1.0$ where the spurious oscillations are to be observed.

3.6.3.1 Static test – estimation of the gradients at the interface

Taking into consideration the fact that an initial level-set distribution corresponds to the analytical solution, an immediate convergence to the solution after one reinitialisation step is expected in the case of a static test case. In order to prove this statement, a static test case, shown in Fig. 3.7, has been performed, using different reinitialisation methods, previously briefly described. The domain is of size $1\times\ 0.1\times 1$m with grid size $\Delta h = 1/128$. Half of the

domain is filled with water, whereas the rest of the domain is treated as empty (air, density $\rho_a = 1.204 kg/m^3$). The accuracy of the gradient and divergence operator has been measured, under the assumption that no advection of the level-set quantity occurs. The reinitialization process is stopped after one single iteration, as done in [134] and a deviation from the analytical solution for the different methods is depicted in Fig. 3.7.

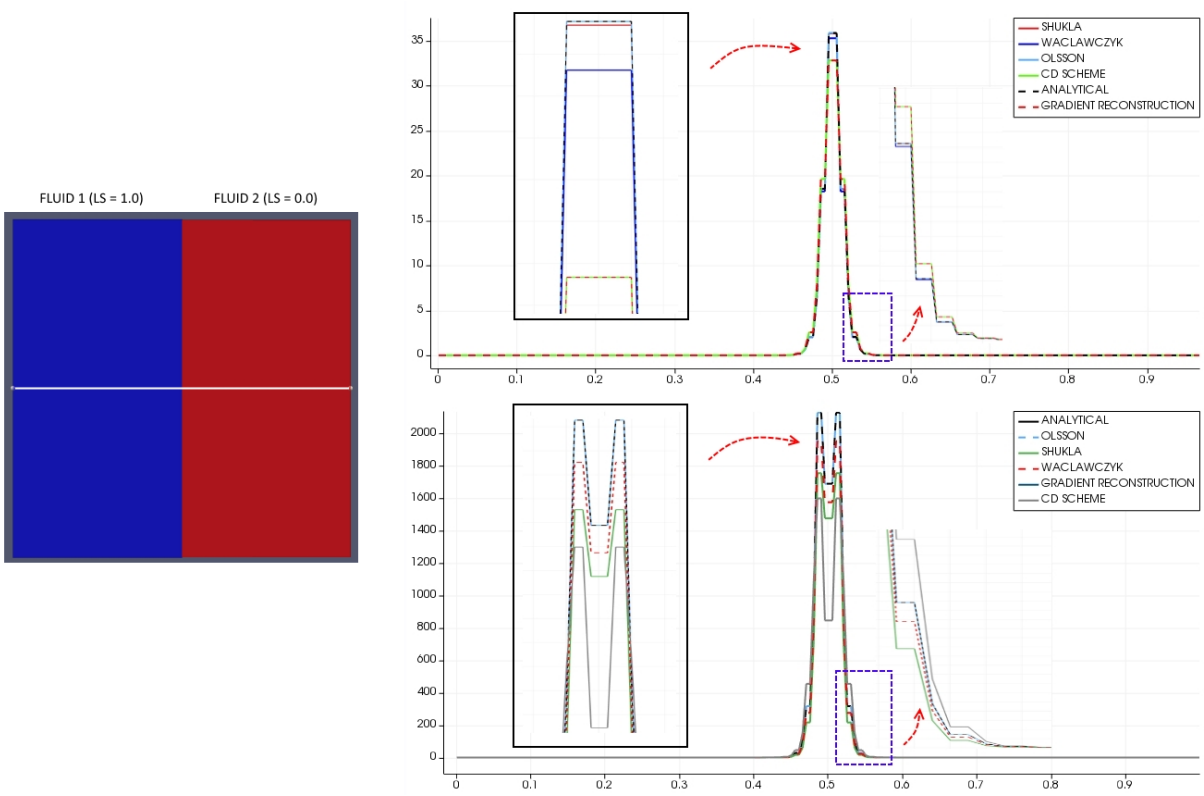


Figure 3.7: Comparison of the different reinitialisation methods withing the estimation of the gradient and divergence operators. Upper plot depicts the magnitude of the gradient function over the level-set scalar value, whereas the lower plot shows the differences in the estimation of the second derivative of the level-set scalar value. The comparison is done after one reinitialization step for the static test case (velocity field $\mathbf{U} = 0$).

Comparing to the analytical solution (solid black line), the worst approximation is obtained using the central-difference scheme, followed by the least-square gradient reconstruction, then the methods introduced by Shukla, Wacławczyk and Olsson. Although an accordance between the analytical solution and the method provided by Wacławczyk seems to be very good in this case, this method often overshoots the analytical solution, causing spurious currents, when an advection of the level-set is included.

3.6.3.2 Pure advection test – comparison of the spatial schemes

Following the instructions from Sec. 4.1.6, out of all the implemented spatial schemes, three are primarily tested in the pure advection test, which is performed in a domain of size 2×1×1m, having an inflow boundary condition imposed on the west side of the domain, an outflow boundary condition on the east side, and all the other sides are represented as an impermeable

solid wall. Due to the fact that open boundary conditions, other than the periodic ones, are imposed, no conservation of mass can be achieved. Instead the focus is put on the preservation of the shape and thickness of the already initialised interface profile. The initial wave is 0.6 m high and 0.5 m long, as shown in Fig. 3.8, and it is transported solely in the x-direction with a constant velocity of $\mathbf{u} = 1m/s$. The cell size is set to $\Delta x = 1/128$ and three different spatial schemes in the level-set advection are compared:

- First-order accurate upwind scheme,
- Second-order accurate upwind scheme with Superbee limiter (implemented by E. Olsson and G. Kreiss [35])
- Second-order accurate upwind scheme with Superbee limiter (implemented by N. Balcázar et al. [101])

The reinitialisation equation (given in Eq. 3.62) is discretised using the second-order CD scheme in space and second-order Runge–Kutta TVD scheme in time. As can be seen in Fig. 3.8, the two second-order upwind schemes perform slightly better than the first-order upwind scheme. Nevertheless, for the orthogonal Cartesian mesh, no significant difference has been observed between those two second-order schemes internally. With an unstructured mesh, an additional projection must be done in order to define the upwind and downwind neighbours, thus the implementation in N. Balcázar et al. [101] offers much more flexibility, retaining the accuracy requirements previously set.

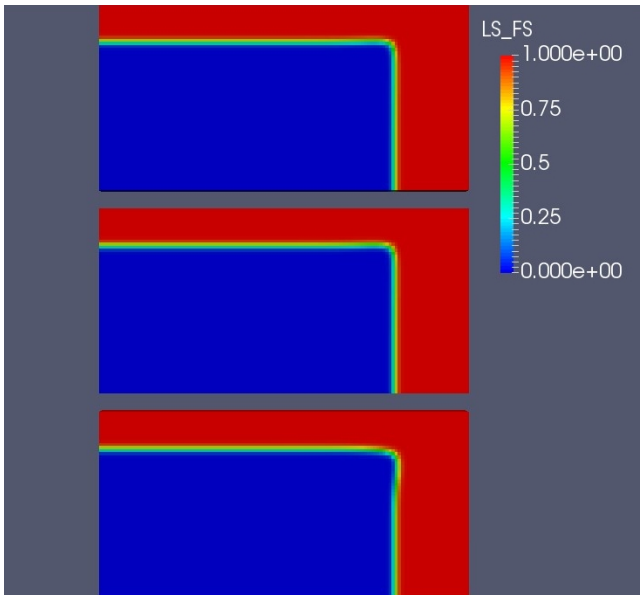


Figure 3.8: Comparison of three different spatial schemes: upwind [101] (topmost), upwind [35] (middle) and first-upwind (bottommost)

3.6.3.3 Pure advection test – cycle and quadratic shape preservation

In order to analyse the influence of various spatial and temporal schemes applied in the process of advection and reinitialisation on the generation of the thin surface profile, a quadratic and circular shape is advected within the 2m × 1m domain. The propagation speed is set to $\vartheta = 1m/s$ in the x-direction, the total simulation time to $t_{end} = 1s$, ensuring that no fluid part reaches the right boundary of the domain. The spatial and temporal discretisations are $N = 512 \times 256$ cells and $\Delta t = 0.001$s, respectively. The circular fluid drop with a radius $r = 0.25$m is initially positioned with its centre at the point $\{0.5, 0.5\}$, whereas the quadratic shape is centred around the same point having an edge $a = 0.5$ m.
As can be seen in Fig. 3.9, twelve different combinations are tested, with three different

temporal schemes: a first-order Euler explicit scheme (EE1), second-order Runge–Kutta (RK2) and third-order Runge–Kutta (RK3), in three respective columns, while the spatial schemes are depicted row-wise in the following order top–down: a first-order upwind scheme, and second-order schemes with, respectively, a Minimod, Sweby and Superbee limiter. In Fig. 3.9 the advection process has been isolated and performed, resulting in significant dissipation of the fluid interface (particularly visible, when the first-order upwind scheme and second-order scheme with the Minimod limiter are employed). The second-order schemes with the Sweby and Superbee limiters, in combination with the EE1 temporal scheme, lead to excessive compression of the interface in the dominant advection direction, whereas all four remaining schemes generate only minimal interface thickness deformation. As already mentioned, the reinitialisation process is responsible for the elimination of unnecessary diffusion or compression within the advection process, thus keeping the interface thickness constant over time. The pure advection results shown in Fig. 3.9 indicate stronger involvement of the reinitialisation process in the eight out of twelve highly diffused and compressed solutions.

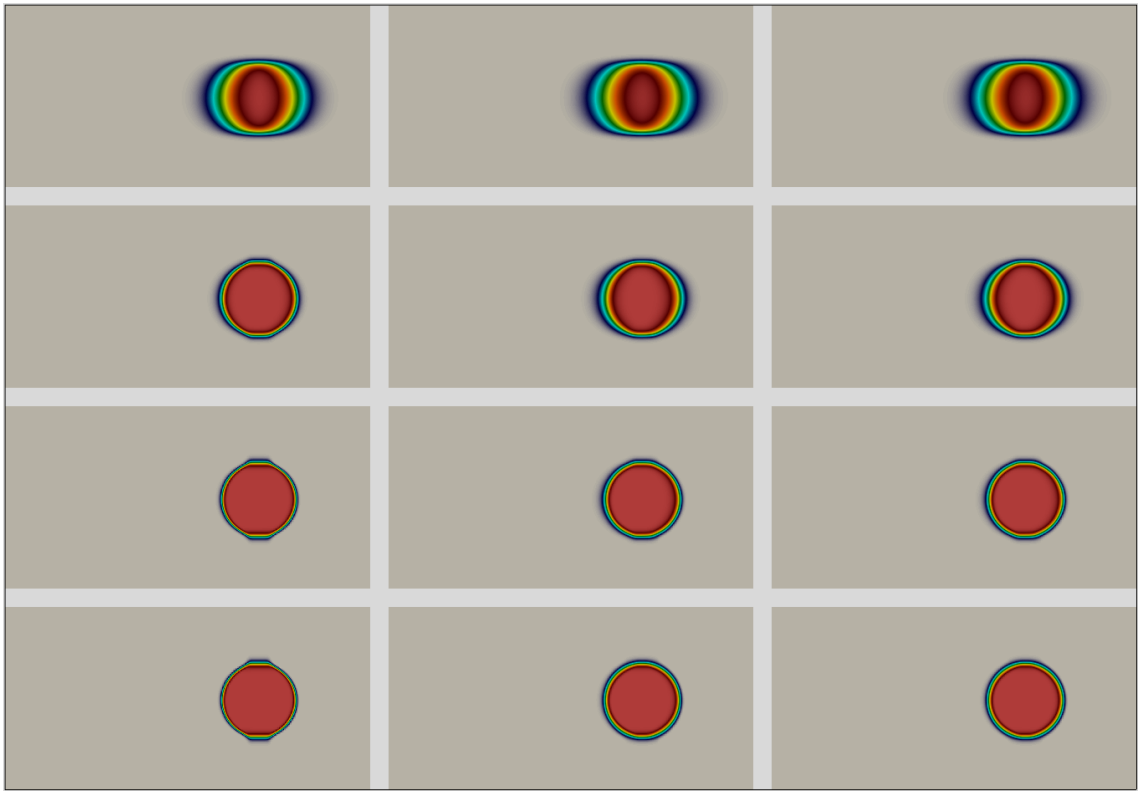


Figure 3.9: Influence of the different temporal and spatial discretization schemes in the level-set advection process onto a dissipation of the thin interface region. Three temporal schemes are depicted column-wise, from left to right: first-order Euler scheme (EE1), second-order Runge–Kutta (RK2) and third-order Runge–Kutta (RK3). The spatial schemes are illustrated row-wise, from top to bottom: first-order upwind scheme, second-order scheme with the Minimod, Sweby and Superbee limiters. The least deformed results are produced using higher-order temporal schemes combined with the Sweby/Superbee limited higher-order spatial schemes.

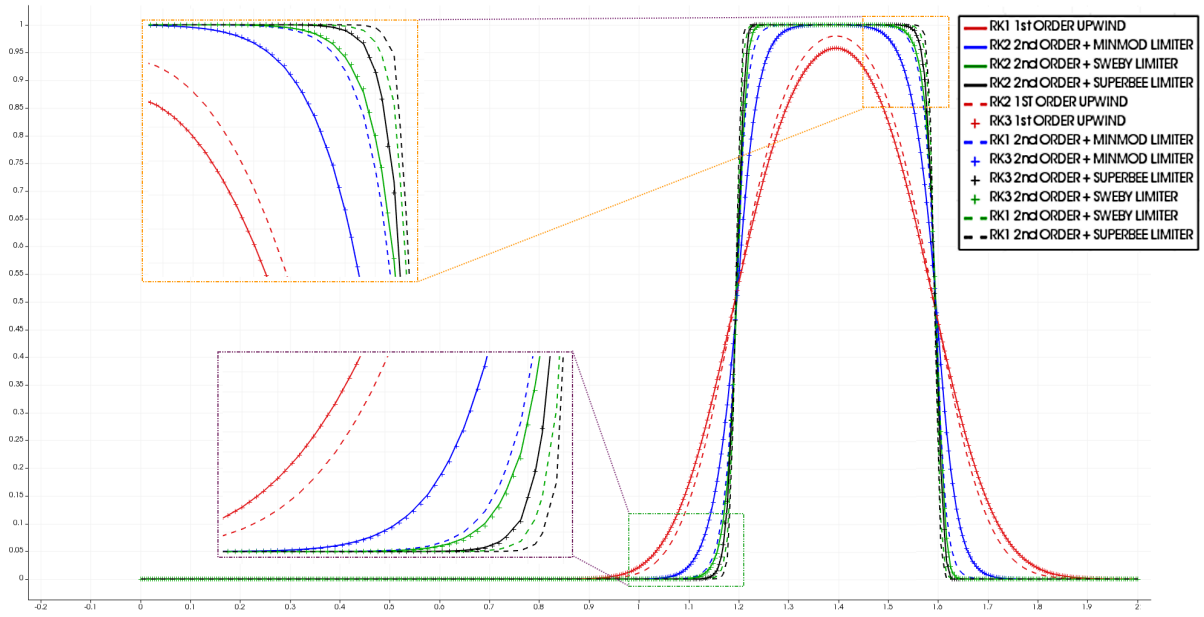


Figure 3.10: Comparison of the amount of diffusion/compression, introduced in the advection process, if no reinitialisation process is used. An analytical solution of non-deformed circular shape coincides with the solid-black-line (RK2 + Sweby limiter). Two schemes are over-compressed (RK1 + Sweby/Superbee limiter), whereas all others introduce additional interface dissipation.

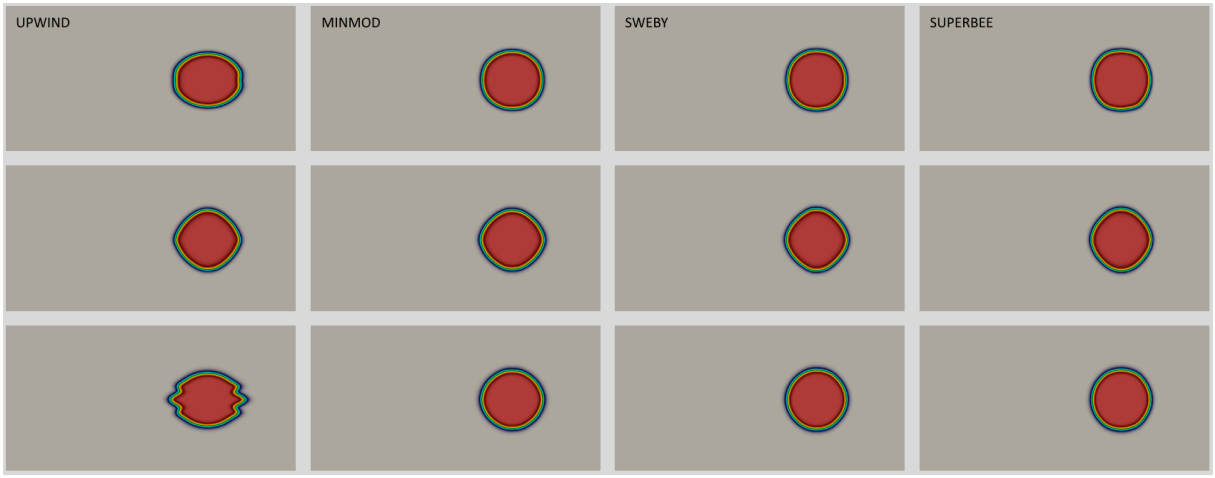


Figure 3.11: Influence of the different temporal and spatial discretisation schemes in the reinitialisation process on the fluid shape deformation. Three reinitialisation spatial schemes are depicted row-wise, from top to bottom: second-order CD scheme, second-order TVD scheme + Superbee limiter, and second-order NON-TVD scheme + limiter $\beta = 0.5$. Four advection spatial schemes are illustrated column-wise, from left to right: first-order upwind scheme, second-order scheme with the Minimod, Sweby and Superbee limiters. The temporal scheme used in both the advection and reinitialisation process is the second-order Runge–Kutta (RK2). The tangential deformation that must be diminished is clearly visible in the second case (TVD + Superbee, see middle column).

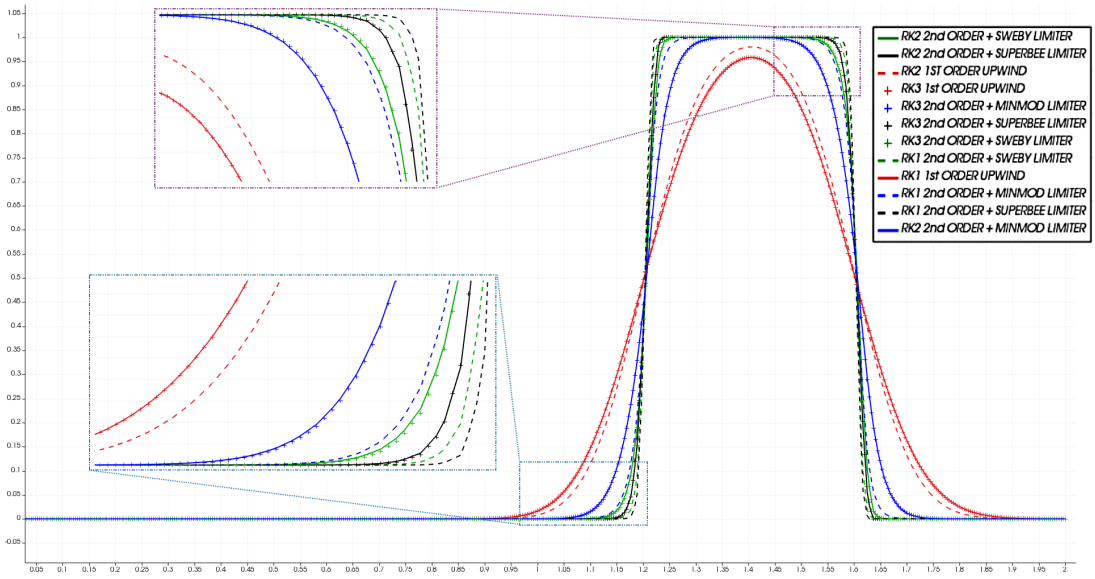


Figure 3.12: Comparison of the amount of diffusion/compression introduced to the advection process, if no reinitialisation process is used. An analytical solution of non-deformed circular shape coincides with the solid-black line (RK2 + Sweby limiter). Two schemes are over-compressed (RK1 + Sweby/Superbee limiter), whereas all others introduce additional interface dissipation.

The amount of interface deformation in all twelve combinations tested is shown graphically in Fig. 3.11, where an analytical solution (no compression and diffusion) coincides with the $RK2 + Sweby$ limiter scheme (solid black line). As the reinitialisation process introduces an artificial tangential diffusion, leading to deformation of the initial fluid domain, the uppermost objective is to keep this process as short as possible. Exhaustive testing, within the 12-scheme scenario indicated that the number of reinitialisation steps increases by up to six times, when a highly diffusive scheme is chosen. The other schemes with minimal interface deformation needed on average two to three reinitialisation steps to reach the same error interface precision ($\epsilon = 10^{-5}$). Furthermore, as the $RK2$ temporal scheme with all four spatial discretisations (Fig. 3.9, middle column) has a reasonable amount of interface deformation, further investigation has been conducted on the influence of the various spatial and temporal schemes within the reinitialisation process.

In Fig. 3.11, twelve further test cases have been conducted, with the four spatial schemes in four respective columns and three spatial (reinitialisation) schemes in the top–down order: second-order CD scheme, second-order TVD scheme + Superbee limiter, and second-order non-TVD scheme + limiter $\beta = 0.5$ (for sake of clarification, the Sweby limiter has the value $\beta = 1.5$). Using the TVD scheme + Superbee limiter (see Fig. 3.11, middle row), the tangential deformation is quite obvious, leading to a change from a circular to a diamond-shape water domain, independent of the spatial scheme used. The remaining two schemes, combined with the second-order spatial schemes, deliver comparable and considerable results. The upwind scheme, employed with the reinitialisation process, results in a rather poor outcome, which becomes even more pronounced when the reinitialisation process is further performed. For sake of completeness, this case is also taken into consideration in the following quadratic case,

which is otherwise abandoned further in the scope of this work.

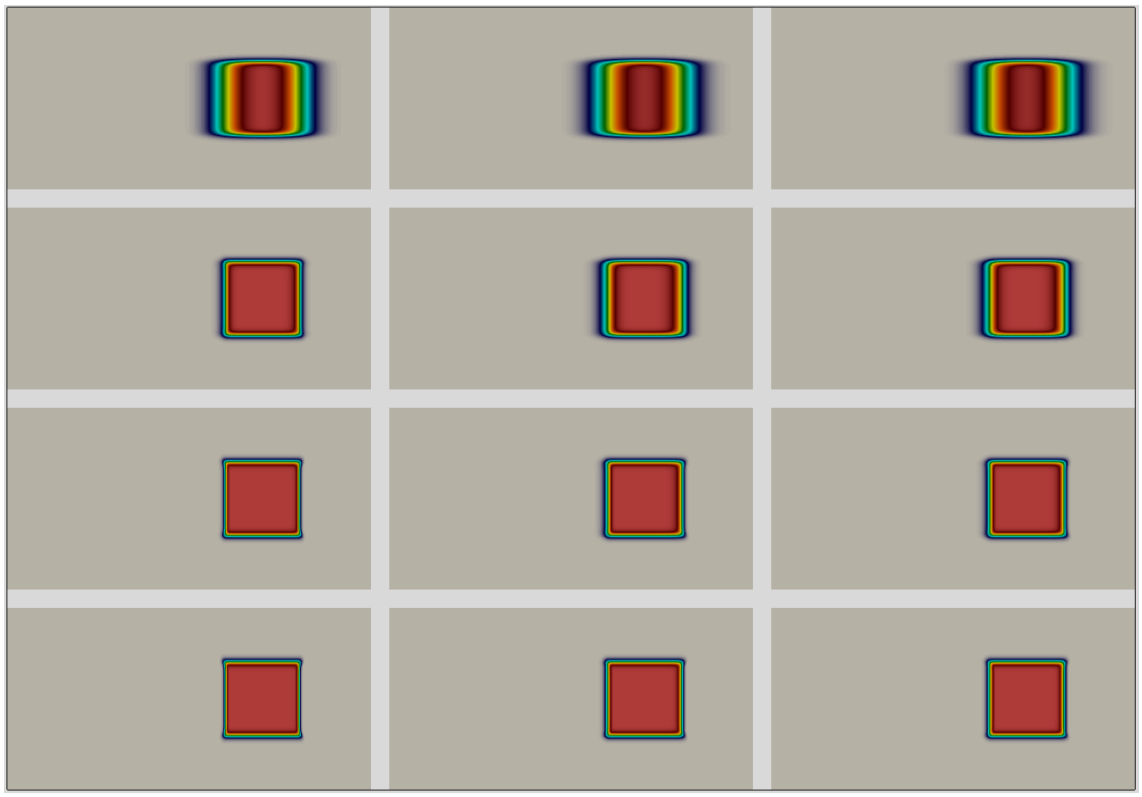


Figure 3.13: Influence of the different temporal and spatial discretisation schemes in the level-set advection process onto a dissipation of the thin interface region. Three temporal schemes are depicted column-wise, from left to right: first-order Euler scheme (EE1), second-order Runge–Kutta (RK2) and third-order Runge–Kutta (RK3). The spatial schemes are illustrated row-wise, from top to bottom: first-order upwind scheme, second-order scheme with the Minimod, Sweby and Superbee limiters. The least deformed results are produced using higher-order temporal schemes combined with the Sweby/Superbee limited higher-order spatial schemes.

The best reinitialised results are achieved using the Sweby/Superbee second-order advection schemes combined with the central-difference and non-TVD reinitialisation schemes, in the case of the circular water domain advected. In order to prove, whether those schemes perform well in general, the second pure-advection test case (quadratic water domain) is performed.

The entire setup, including the spatial and temporal schemes employed is identical. The behaviour observed in the circular pure-advection setup is in good agreement with the observations in the quadratic pure-advection setup. The amount of dissipation and compression of the fluid interface is almost identical, if Figs. 3.10 and 3.13 are compared. Nevertheless, the tangential diffusion introduced in the reinitialisation process is more severe in the quadratic case, leading to the elimination of the non-TVD scheme that performed well in the circular setup. For all the further setups chosen, the advection process is performed using the second-order scheme + Sweby limiter, whereas the reinitialisation process, based on the second-order accurate CD scheme, is adopted, as it delivered the most reliable results.

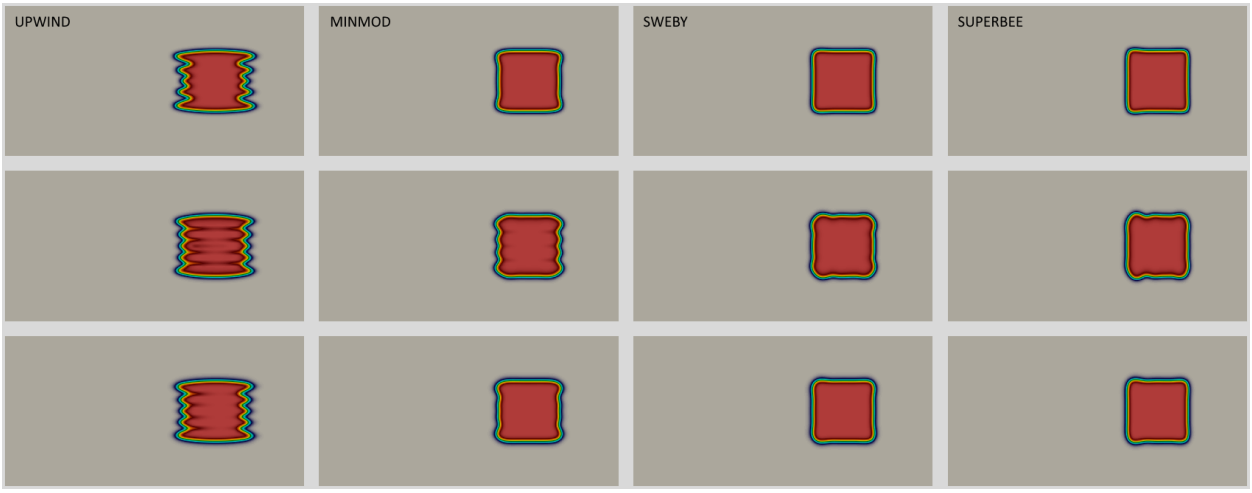


Figure 3.14: Influence of the different temporal and spatial discretisation schemes in the reinitialisation process onto the quadratic fluid shape deformation. Three reinitialisation spatial schemes are depicted row–wise, from top to bottom: second-order CD scheme, second-order TVD scheme + Superbee limiter, and second-order NON-TVD scheme + limiter $\beta = 0.5$. Four advection spatial schemes are illustrated column–wise, from left to right: first-order upwind scheme, second-order scheme with the Minimod, Sweby and Superbee limiters. The temporal scheme used in both the advection and reinitialisation process is the second-order Runge–Kutta (RK2). The tangential deformation that must be diminished is clearly visible in the second case (TVD + Superbee, see middle column).

3.6.3.4 Advection test – Solenoidal velocity field interface transport

The validation setups described in detail in Sec. 3.6.3.3 show the effectiveness of the high-order reinitialisation schemes in the one-directional flow with the constant velocity prescribed. Another standard validation setup that proves the quality of the CLS approach in multiple directions is the rotation of a fluid droplet in 2D space, moved by a solenoidal velocity field, defined as follows:

$$U(x,z) = -sin^2(\pi x) \cdot sin(2\pi z), \qquad V(x,z) = sin^2(\pi z) \cdot sin(2\pi x) \tag{3.80}$$

The initial physical setup, depicted in Fig. 3.15, is used in combination with three different mesh configurations $\{R_1, R_2, R_3\} = \{64^2, 128^2, 256^2\}$, showing the number of computing cells per direction in the uniformly refined domain. The advection of the level-set interface profile is done using the second-order accurate scheme with the Sweby limiter, and the three reinitialisation steps are performed with the second-order central difference scheme in space. For both the advection and reinitialisation processes the second-order Runge–Kutta scheme in time is adopted. The setup runs in total T = 2s, while at t = T/2 the velocity direction is reversed, in order to recreate the initial state at the end of the simulation.
The results obtained within this validation example (see Fig. 3.16) are in good accordance with E. Olsson and G. Kreiss [35] and [36], in which the setup is initially established. Three different mesh configurations $\{R_1, R_2, R_3\}$ are depicted row-wise, showing the flow development in time at $t\,[s] = \{0, 0.5, 1, 2\}$. For $T = 2s$ the initial and end state of the simulation are compared, emphasising the influence of the mesh refinement on the shape deformation. The

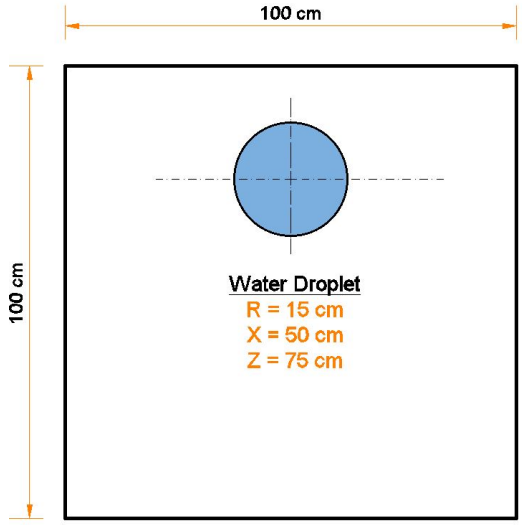


Figure 3.15: Physical setup of the rotating droplet, moved by the solenoidal velocity field. The domain $\{0, 100\} \times \{0, 100\}$ with a single droplet $r = 15cm, C = \{50cm, 75cm\}$ is numerically discretised using three different mesh configurations $\{R_1, R_2, R_3\} = \{64^2, 128^2, 256^2\}$.

coarse representation R_1 is obviously not sufficient to capture all the flow phenomena in this complex behaviour. Refining to R_2 and R_3 reveals that the interface stays compact longer without flow separation at the tail of the spiral form, aimed also at much better concordance between the initial and end states of the simulation (see initial (red) and end (blue) states, Fig. 3.16).

Running the refined validation setup R_3 for a longer period of time ($T = 4s$, $\Delta t = 0.00025s$) would develop the spiral form even further, creating an extremely thin interface in the back part of the spiral, which cannot be resolved in the current mesh configuration, thus separation droplets would start to develop, moving independently of the main fluid body. This behaviour is captured in the original research in E. Olsson and G. Kreiss [35] (see Fig. 3.17) and is reproduced in the current work (see Fig. 3.18). Compression and diffusion in the reinitialisation process affect the total conservation as well as the preservation of both fluid phases that are separated with the thin interface area. In this concrete example, the area conservation measured in the relative error with regard to the initial state in the whole domain varied between $\varepsilon = [10^{-7}, 10^{-10}]$, for three respective mesh setups, while the conservation of the water phase is measured separately in absolute units $[m^2]$ and depicted in Fig. 3.19. For the coarsest mesh R_1 the relative error is 5.28%, rapidly decreasing to 1.03% for the R_2 setting and finally to only 0.032% in R_3.

For the sake of curiosity, one further level of refinement is introduced, resulting in a very fine mesh $R_4 = \{512^2\}$, in order to test the influence of the mesh size on the droplet separation visible in Fig. 3.18 for the duration of the simulation $T = 4s$. Such a fine mesh allows the complete fluid body to remain continuous (see Fig. 3.20), thus no further refinement is necessary, should the simulation not be run longer.

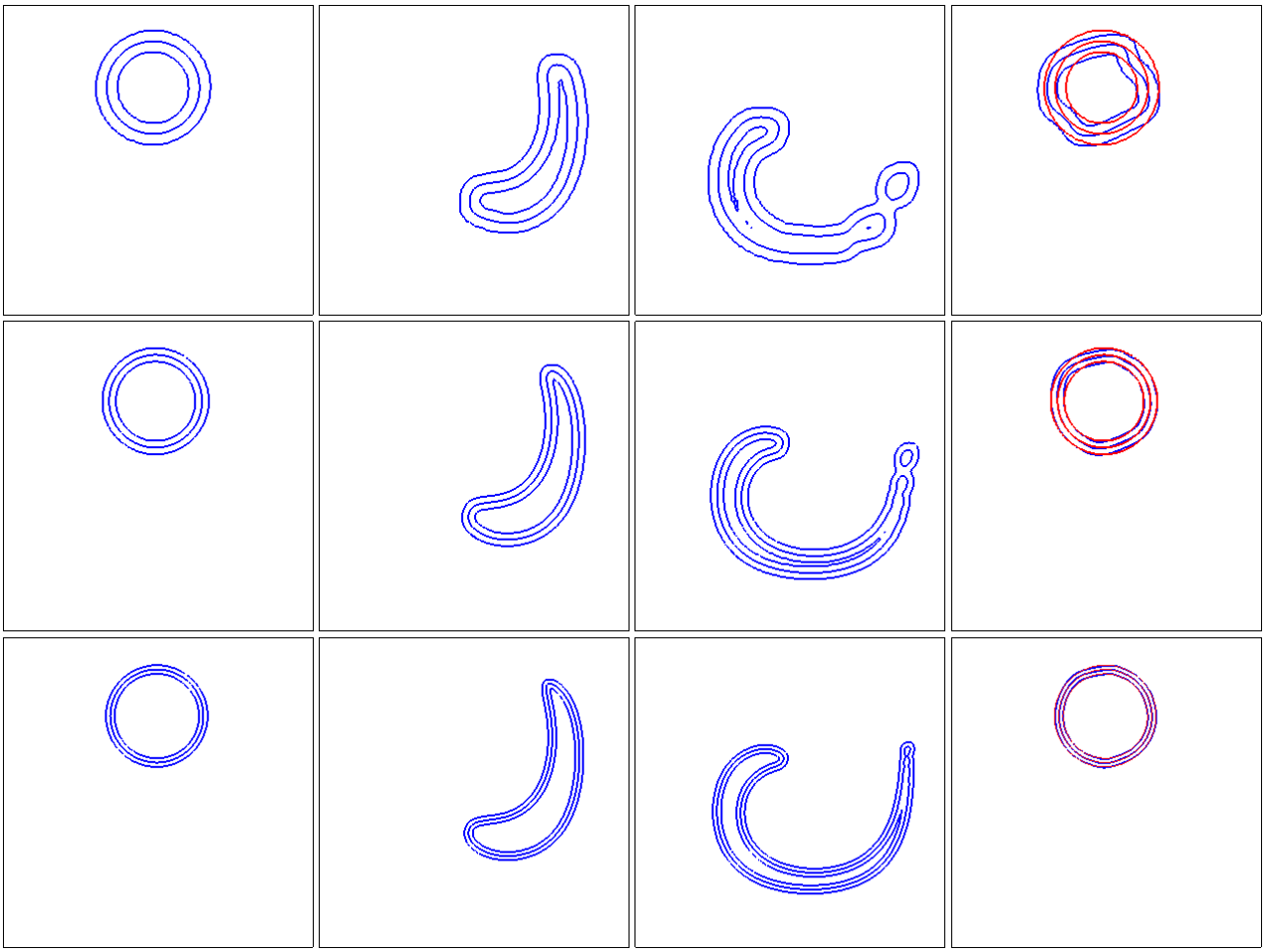


Figure 3.16: Comparison of the $\{R_1, R_2, R_3\}$ – three different mesh configurations with $\{64^2, 128^2, 256^2\}$ computing cells in total, depicted row-wise respectively. The horizontal plots show the development of the fluid vortex at $t = 0s$, $t = 0.5s$, $t = 1s$ and $t = 2s$. The right-most plot is achieved by reversing the flow direction at t = 1s. Herein, the red line depicts the initial state, whereas the blue line is obtained at the end of the simulation. The deformation of the initial state is rapidly diminished with the decrease of the cell size.

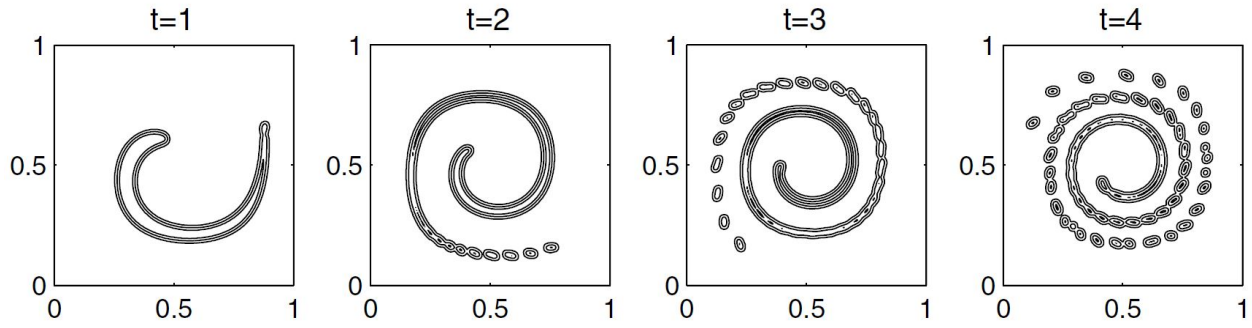


Figure 3.17: Screen-shot of the long simulation of the fluid vortex at times $t = 1s$, $t = 2s$, $t = 3s$ and $T = 4s$, published in [35].

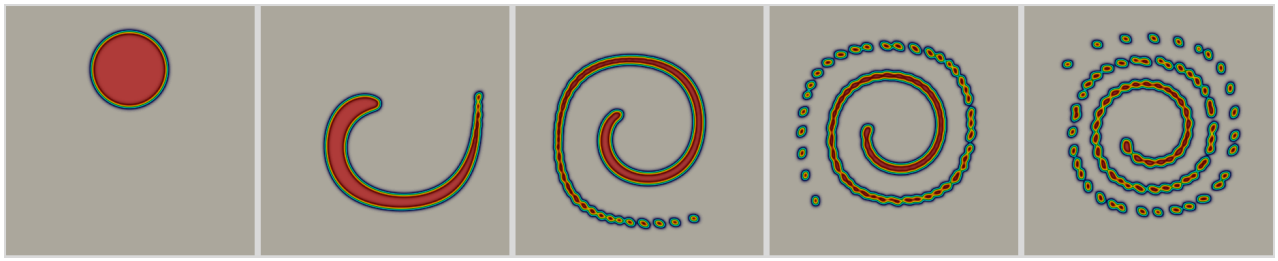


Figure 3.18: Long simulation of the fluid vortex at times $t = 0s$, $t = 1s$, $t = 2s$, $t = 3s$ and $T = 4s$ obtained using the R_3 mesh refinement and $\Delta t = 0.00025s$. A flow direction has remained constant throughout the simulation time. Due to the insufficient mesh refinement, the separation of the fluid droplets appears at the back trace of the spiral form. A good concordance with the results depicted in Fig. 3.17 is achieved.

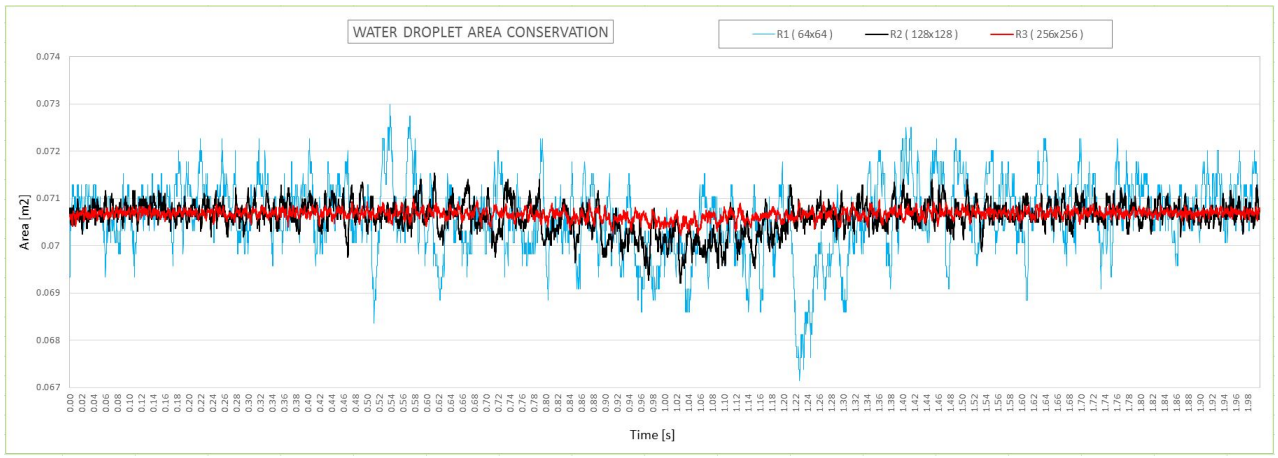


Figure 3.19: Preservation of the fluid phase area for all three mesh refinement setups $\{R_1, R_2, R_3\}$. The absolute units are shown and the relative errors with the respect to the initial fluid area are analysed, resulting in 5.28%, 1.03% and 0.03%, respectively, whereas the cumulative mass conservation in the entire domain reads a negligible relative error of $\varepsilon = \left[10^{-7}, 10^{-10}\right]$.

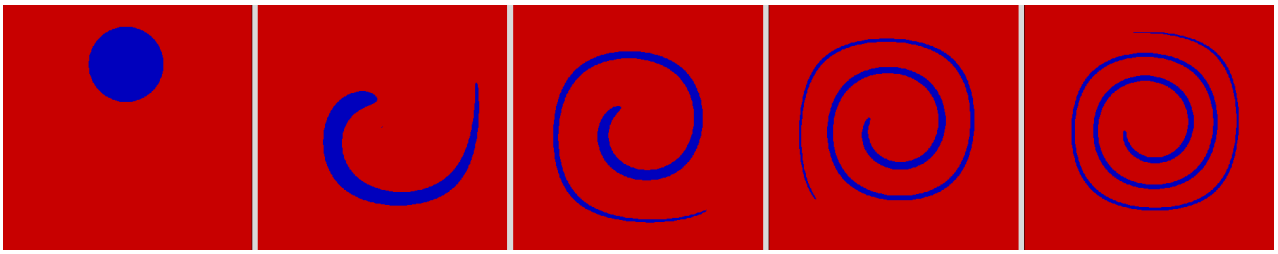


Figure 3.20: Long simulation of the fluid vortex at times $t = 0s$, $t = 1s$, $t = 2s$, $t = 3s$ and $T = 4s$ obtained using the R_4 mesh refinement reveals that the further mesh refinement could resolve even finer scales, allowing the fluid vortex to remain continuous along the entire trace.

Résumé of the schemes and methods used:

Numerous tests done within this section indicated very clear that some methods can be better combined with some particular routines, in order to meet widely used validation criteria. The following conclusion is intended to help further researches to understand strong and weak points of approach applied, while used within the boundaries set at the beginning of this work. All further generalisations must be done with care, followed by appropriate validation and/or verification tests. To the best of our knowledge, the sharp interface capturing method volume-of-fluid promises very good mass conservation, it is, however, due to an inability to calculate precisely the interface normal vector and curvature, not suitable for flows in which the surface tension plays an important role. Moreover, the interface smearing must be addressed, thus different scientific communities suggest an inclusion of an artificial compression term to control the amount of diffusion. Different interface reconstruction techniques can be used to improve fluid behaviour at flow discontinuities. On the other hand, simulations of more than two phases at once can be done with a lot of success using this method. The standard level-set method is still widely used in many industrial applications although the smearing of the distance function calculated can be very severe. To remedy this problem, a re-distance or reinitialisation routine is commonly used; nevertheless the mass conservation is still an issue in majority of the application scenarios. Conservative level-set method reduces significantly the loss of mass and it uses the re-distance function to keep the interface thickness value constant. Throughout the work, the CLS worked satisfactory in combination with the following schemes:

APPLIED PROCEDURE	STATUS	IMPLEMENTED SCHEME
Normal vector & curvature calculation	ok	smooth θ function, [118]
	ok	modified ϵ function, [36]
	×	smooth ψ function, [134]
Level-set reinitialisation (spatial schemes)	×	central difference scheme
	×	first-order upwind scheme
	×	second-order upwind scheme + minmod flux limiter
	ok	second-order upwind scheme + sweby flux limiter
	ok	second-order upwind scheme + superbee flux limiter
Level-set reinitialisation (temporal schemes)	×	first-order Euler explicit
	×	second-order Adams-Bashforth
	ok	first-order TVD Runge–Kutta
	ok	second-order TVD Runge–Kutta
	ok	third-order TVD Runge–Kutta

Table 3.3: An overview of interface capturing methods and schemes used throughout this work.

Chapter 4

Numerical Treatment of NSE and Various Coupling Strategies

4.1 Numerical treatment of Navier–Stokes equations

4.1.1 Numerical discretisation and grid arrangement

Numerical discretisation is a transformation process in which the continuous functions, equations and mathematical models introduced in Sec. 3 are transferred to their discrete counterpart. This step is necessary if computational resources are used to solve various engineering problems defined in continuous space. In order to establish a unique mapping between the discrete and continuous representation, already well-established spatial propositions, such as the finite difference method (FMD), finite element method (FEM), finite volume method (FVM), as well as frequently exploited temporal schemes of different orders of accuracy, can be combined together.
For the sake of completeness, a short overview of the FDM and FVM spatial concepts and several temporal schemes, such as first-order Euler, second-order Adams–Bashforth and second- and third-order Total-Variation-Diminishing Runge–Kutta, will be given next, as those are actively used within this work. For a general understanding and more available options, the reader is referred to the standard literature (J. H. Ferziger and M. Perić [65] and C. Hirsch [22]).

4.1.1.1 Finite difference method

The finite difference approach assumes the existence of two or more families of lines, in general non-equidistant, that cross and create a unique set of discrete intersection points, as shown in Fig. 4.1, in which the variables of interest are calculated, according to the given mathematical model. For the purpose of explanation, a simple transport equation, such as Eq. 4.1 (also Eq. 3.55) will be used:

$$\frac{\partial \Phi}{\partial t} + \nabla \cdot (\mathbf{u}\Phi) = 0 \tag{4.1}$$

where Φ represents any transport scalar property of interest and $\mathbf{u}$ is a velocity vector field with which the transport property is advected in space and time.

The calculation of the advection term in Eq. 4.1 requires the approximation of the first derivative $\partial \mathbf{u}\Phi/\partial x$ at a particular grid point (i, j), as shown in Fig. 4.1, in every spatial dimension. This can be easily done using a Taylor series expansion:

$$f(x) = f(x_i) + (x_i - x)\left(\frac{\partial f}{\partial x}\right)_i + \frac{(x_i - x)^2}{2!}\left(\frac{\partial^2 f}{\partial x^2}\right)_i + \cdots + \frac{(x_i - x)^n}{n!}\left(\frac{\partial^n f}{\partial x^n}\right)_i + H \quad (4.2)$$

where $f(x)$ represents a continuous differentiable function approximated around the point x_i and the term H groups together all the approximations of a higher order. For the sake of concise mathematical writing, the $f(x)$ function is used throughout the entire derivation process and applied onto the transport equation Eq. 4.1 only towards the end of this section. Using the Taylor series from Eq. 4.2 and replacing the value x_i by the neighbouring points $x_{i-1} = x - h$ or $x_{i+1} = x + h$ the well-known backward-difference (BDS) and forward-difference (FDS) schemes are obtained. An equidistant mesh $\Delta x_i = h$ is assumed for the sake of concise mathematical representation.

Forward difference scheme: (4.3)

$$f(x+h) = f(x) + h\left(\frac{\partial f}{\partial x}\right)_i + \frac{h^2}{2!}\left(\frac{\partial^2 f}{\partial x^2}\right)_i + \cdots + \frac{h^n}{n!}\left(\frac{\partial^n f}{\partial x^n}\right)_i + H \quad (4.4)$$

$$\left(\frac{\partial f}{\partial x}\right)_i = \frac{f(x+h) - f(x)}{h} - \frac{h}{2!}\left(\frac{\partial^2 f}{\partial x^2}\right)_i - \cdots - \frac{h^{n-1}}{n!}\left(\frac{\partial^n f}{\partial x^n}\right)_i - H \quad (4.5)$$

$$\left(\frac{\partial f}{\partial x}\right)_i = \frac{f(x+h) - f(x)}{h} + \mathcal{O}(h) \quad (4.6)$$

Backward difference scheme: (4.7)

$$f(x-h) = f(x) - h\left(\frac{\partial f}{\partial x}\right)_i + \frac{h^2}{2!}\left(\frac{\partial^2 f}{\partial x^2}\right)_i - \cdots + \frac{h^n}{n!}\left(\frac{\partial^n f}{\partial x^n}\right)_i + H \quad (4.8)$$

$$\left(\frac{\partial f}{\partial x}\right)_i = \frac{f(x) - f(x-h)}{h} + \frac{h}{2!}\left(\frac{\partial^2 f}{\partial x^2}\right)_i + \cdots + \frac{h^{n-1}}{n!}\left(\frac{\partial^n f}{\partial x^n}\right)_i + H \quad (4.9)$$

$$\left(\frac{\partial f}{\partial x}\right)_i = \frac{f(x) - f(x-h)}{h} + \mathcal{O}(h) \quad (4.10)$$

Central difference scheme: (4.11)

$$\left(\frac{\partial f}{\partial x}\right)_i = \frac{f(x+h) - f(x-h)}{2h} - \frac{h^2}{3!}\left(\frac{\partial^3 f}{\partial x^3}\right)_i - \cdots - \frac{h^{n-2}}{(n-1)!}\left(\frac{\partial^{n-1} f}{\partial x^{n-1}}\right)_i + H \quad (4.12)$$

$$\left(\frac{\partial f}{\partial x}\right)_i = \frac{f(x+h) - f(x-h)}{2h} + \mathcal{O}(h^2) \quad (4.13)$$

The higher-order derivatives are commonly unknown; however, with the reduction of the distance between the neighbouring points x_{i-1}, x_i and x_{i+1} the contribution of these terms becomes very small in the majority of cases, thus truncating those terms out of the series leads to the approximation of the first derivative, as depicted in Eqs. 4.14–4.16. These truncated values are called truncation errors and they represent the rate at which the discretization errors are reduced with the reduction of the spacing between two neighbouring points (see Fig. 4.1).

$$\left(\frac{\partial f}{\partial x}\right)_{forward} \approx \frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i} \tag{4.14}$$

$$\left(\frac{\partial f}{\partial x}\right)_{backward} \approx \frac{f(x_i) - f(x_{i-1})}{x_i - x_{i-1}} \tag{4.15}$$

$$\left(\frac{\partial f}{\partial x}\right)_{central} \approx \frac{f(x_{i+1}) - f(x_{i-1})}{x_{i+1} - x_{i-1}} \tag{4.16}$$

If both the left- and right-hand neighbours are used, the central difference (CDS) scheme (Eq. 4.16) is obtained, guaranteeing second-order accuracy $\mathcal{O}(h^2)$, as stated in [65]. Nevertheless, depending on the complexity of the problem discretised, schemes that guarantee first and second orders of accuracy are often not satisfactory, thus much effort is invested into fitting some more complex shapes (e.g. parabolas or cubic polynomials) through several neighbouring points. Even though it requires a wider computational stencil, it leads to an approximation equal to the degree of the polynomial used.
Applying the formulas given in Eqs. 4.14–4.16 onto the advection term of the transport equation Eq. 4.1 leads to the general form depicted in Eqs. 4.17–4.19. All the assumptions set for the function $f(x)$ must also be fulfilled for the function $\mathbf{u}\Phi(x)$:

$$\frac{\partial}{\partial x}(\mathbf{u}\Phi)^{forward} \approx \frac{\mathbf{u}\Phi(x_{i+1}) - \mathbf{u}\Phi(x_i)}{x_{i+1} - x_i} \tag{4.17}$$

$$\frac{\partial}{\partial x}(\mathbf{u}\Phi)^{backward} \approx \frac{\mathbf{u}\Phi(x_i) - \mathbf{u}\Phi(x_{i-1})}{x_i - x_{i-1}} \tag{4.18}$$

$$\frac{\partial}{\partial x}(\mathbf{u}\Phi)^{central} \approx \frac{\mathbf{u}\Phi(x_{i+1}) - \mathbf{u}\Phi(x_{i-1})}{x_{i+1} - x_{i-1}} \tag{4.19}$$

All the previous formulations of the different schemes are given in one spatial dimension and their extension to the second and third dimensions is straightforward. In the case of a non-orthogonal irregular mesh representation, these expressions tend to become complicated due to the necessary coordinate transformations. For that reason, in such complex cases the finite volume method offers a more general approach and easier numerical treatment.

4.1.1.2 Finite volume method

Unlike in the case of the finite difference approach, where all the data is stored in previously generated computational nodes, the solution domain in the finite volume approach is divided

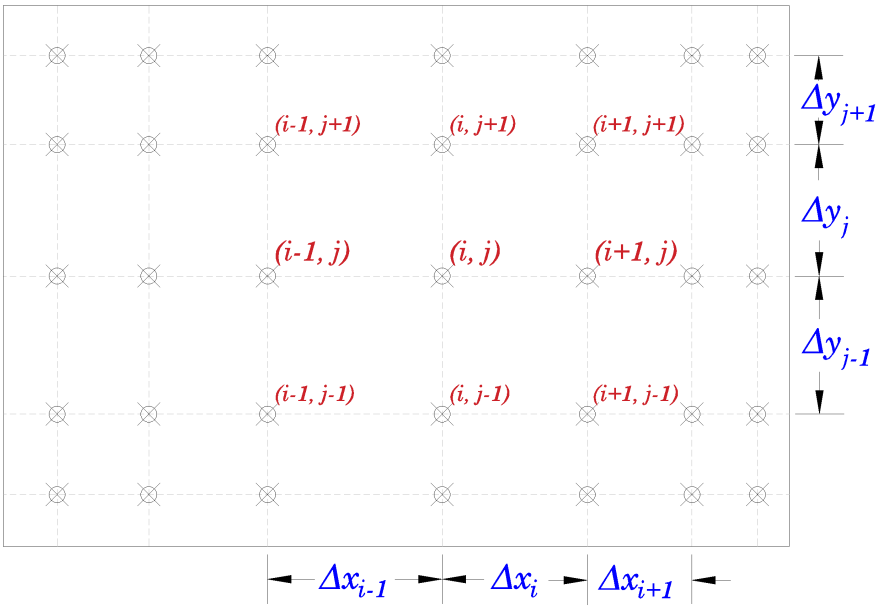


Figure 4.1: A non-uniform grid arrangement typically used to discretise a computational domain, within the FDM approach.

into small control volumes (CV) and the integration of the transport quantity over the surface of those control volumes enforces the local conservation properties. While the integral formulation of the momentum equation is transformed into its differential form within the finite difference method, the finite volume approach directly utilises the integral formulation itself, which is given in the following form:

$$\oint_{F_c} \rho\phi\mathbf{u}\cdot\mathbf{n}_f dF_c = \oint_{F_c} \sigma_\phi\cdot\mathbf{n}_f dF_c + \int_{\Omega_{CV}} f_\phi d\Omega_{CV} \tag{4.20}$$

where ϕ represents a quantity of interest (if $\phi = \mathbf{u}$ a momentum conservation law is obtained, for instance), f comprises all volumetric forces applied to the system, σ_ϕ represents normal and tangential surface forces, Ω_{CV} the control volume size, F_c the area of the faces that bound the control volume (CV), with the normal vector $\mathbf{n}_f$ pointing outwards as depicted in Fig. 4.2. To guarantee the global conservation properties, the integral form given in Eq. 4.20 must be applicable on both a single control volume and the entire solution domain. This is true due to the fact that surface integrals at the common face of two neighbouring CVs cancel out, as illustrated in Fig. 4.2, thus the only contribution to the global conservation comes from boundaries and volumetric source terms.

In order to obtain an algebraic form, the surface and volume integrals in Eq. 4.20 must be approximated, influencing in that way an order of accuracy and the final form of equations, further used.

The simplest way to approximate a volume integral is to multiply an averaged value of the integrand $\mathbb{S}_\phi$ (see Eq. 4.20) with the CV volume Ω_{CV}. If $\mathbb{S}_\phi$ is a constant value or it changes linearly in space, this averaged value can be replaced with the cell-centred value, as depicted in Eq. 4.21, resulting in a second-order accurate approximation. In case that a higher-order accuracy is required, the integrand value must be interpolated using more nodal values.

In order to maintain local conservation properties, an approximation of surface integrals must be done with care. Having a property Q, where Q stands for $\rho\phi\mathbf{u}$ and σ_ϕ in the advection and diffusion terms of Eq. 4.20, respectively, a total flux of that property over a control volume boundary is equal to the sum of the fluxes over all m separate faces of that CV:

$$\text{Volume integral:} \quad \int_{\Omega_{CV}} \mathbb{S}_\phi d\Omega_{CV} \approx \overline{\mathbb{S}_\phi}\Omega_{CV} = \mathbb{S}^c_\phi \Omega_{CV} \tag{4.21}$$

$$\text{Surface integral:} \quad \oint_{F_c} Q \cdot \mathbf{n} dF_c \approx \sum_{f=1}^{m} \oint_{F_c^f} Q \cdot \mathbf{n}_f dF_c, \quad Q = \rho\phi\mathbf{u} \text{ or } Q = \sigma_\phi \tag{4.22}$$

The property Q is commonly saved in the cell-centroid positions, thus to calculate the surface integral with great precision, this value must be approximated in at least one point at every face using already known values. To enhance the accuracy this approximation can be done at several spatial points at a particular face, after which the integration over the points is performed.

Depending on the choice of grid arrangement (i.e. a collocated or staggered approach, see Sec. 4.1.1.3), the amount of interpolation done between the computational cells can vary significantly, thus some possible options with an analysis of their accuracy are given in [65]. An in-depth study published by L. Jofre et al. [80] takes into account the influence of the interpolation procedure onto the conservation of energy as well, and the general conclusions drawn are that:

1. the scalar values stored in the cell centroids should be interpolated using the weighted distance approximation:

$$\psi_f = \sum_{i=c,nb} \left[1 - \frac{|d(f,i)|}{|d(c,nb)|}\right] \psi_i, \quad d(a,b) \rightarrow \text{distance between two points, a and b;} \tag{4.23}$$

2. the vector values stored at the face location can be used without any additional treatment, leading to good conservation properties; and

3. the vector values stored in the cell centroids must be reconstructed, in order to avoid pressure–velocity decoupling, introduced for the first time by C. M. Rhie and W. L. Chow

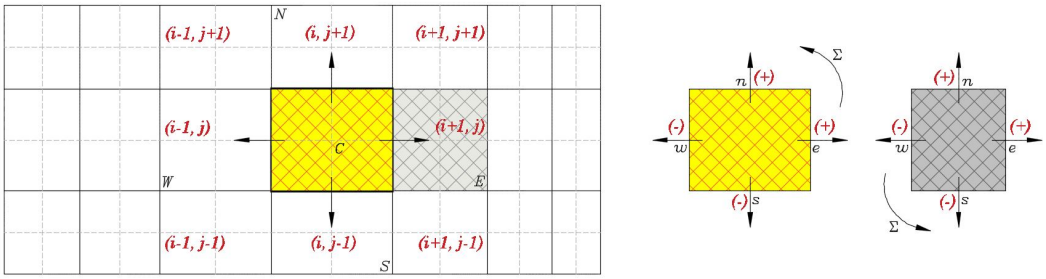


Figure 4.2: A non-uniform grid arrangement typically used to discretise a computational domain, within the FVM approach.

[24], preserving the mass, momentum and energy quantities transferred over the cell faces. The detailed reconstruction procedure is given in Sec. 4.1.3.

Besides the recommendations listed above, the gradient reconstruction and inverse distance weighting methods are often used to approximate all necessary face points. Furthermore, one common practice in the simulation of flows with large scalar gradients is the usage of the upwind scheme (see Eq. 4.24), which, depending on the case simulated, results in a larger or smaller amount of artificial diffusion, thus additional attention is required.

$$\psi_f = \begin{cases} \psi_c, & \mathbf{u_f} \cdot \mathbf{n}_f \geq 0 \\ \psi_{nb}, & \mathbf{u_f} \cdot \mathbf{n}_f < 0 \end{cases} \tag{4.24}$$

A much more accurate approximation $\mathcal{O}(\Delta x^3)$ is obtained using the quadratic upwind scheme (QUICK) given in Eqs. 4.25 and 4.26; however, the memory requirements are increased, as the computational stencil consists of four computing control volumes (one current cell and three direction-dependent neighbours) as shown in Fig. 4.3.

$$\psi_e = \begin{cases} \frac{6}{8}\psi_C + \frac{3}{8}\psi_E - \frac{1}{8}\psi_W, & \mathbf{u_e} \cdot \mathbf{n}_e \geq 0 \\ \frac{6}{8}\psi_E + \frac{3}{8}\psi_C - \frac{1}{8}\psi_{EE}, & \mathbf{u_e} \cdot \mathbf{n}_e < 0 \end{cases} \tag{4.25}$$

$$\psi_w = \begin{cases} \frac{6}{8}\psi_W + \frac{3}{8}\psi_C - \frac{1}{8}\psi_{WW}, & \mathbf{u_w} \cdot \mathbf{n}_w \geq 0 \\ \frac{6}{8}\psi_C + \frac{3}{8}\psi_W - \frac{1}{8}\psi_E, & \mathbf{u_w} \cdot \mathbf{n}_w < 0 \end{cases} \tag{4.26}$$

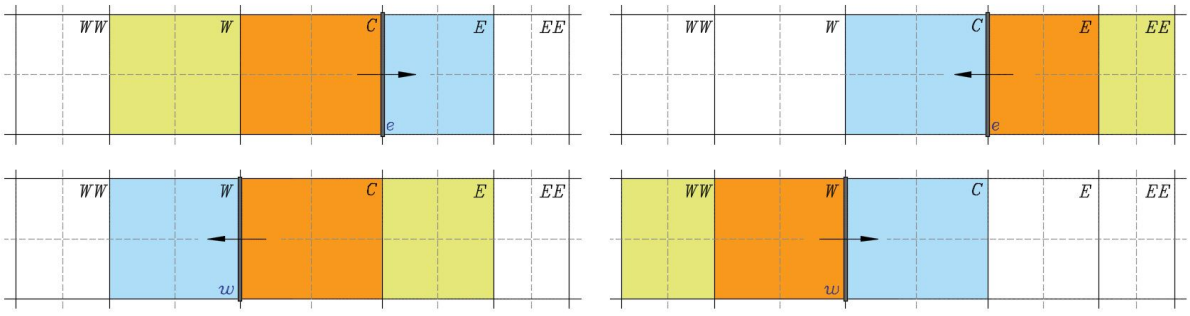


Figure 4.3: Extended computational stencil necessary for the quadratic upwind scheme QUICK (see Eqs. 4.25 and 4.26). A particular influence of three dominant cells: direct upwind (orange), direct downwind (blue) and faraway-upwind (yellow) is well-known in case of the orthogonal equidistant mesh arrangement.

If an irregular non-orthogonal mesh is used, a projection onto the normal vector of the face should be performed, in order to identify the direct-upwind, direct-downwind and faraway-upwind neighbours.

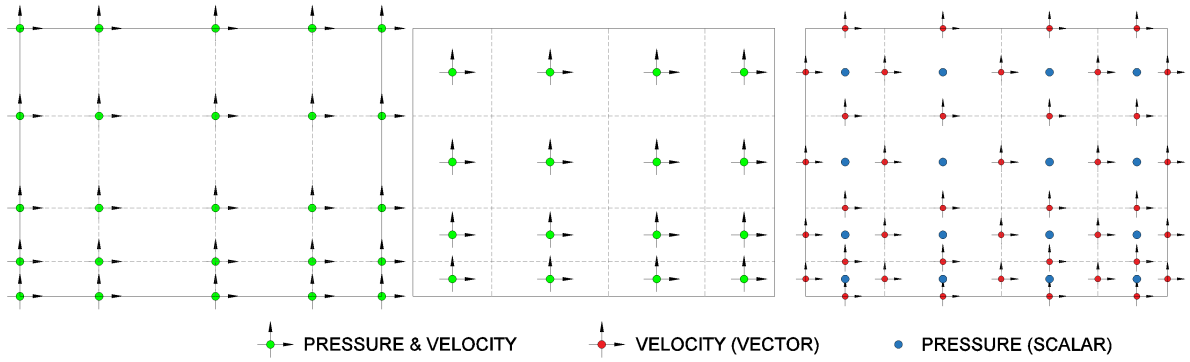


Figure 4.4: Three common mesh arrangement scenarios: collocated mesh for the FD approach (leftmost), collocated mesh for the FV approach (middle plot) and staggered mesh for the FV approach (rightmost). In case of the collocated arrangements, both scalar and vector entities are stored at the same location, whereas in case of the staggered mesh the scalar values are stored in cell centroids and vector variables at the face between two cells, enabling easier calculation of the fluxed quantities.

4.1.1.3 Collocated and staggered grid arrangements

Within the discretisation procedures, the choice of the location where the variables should be calculated, must be done at the very beginning and it significantly influences both the memory requirements and the efficiency of the entire simulation process. The most commonly chosen grid arrangements are: a collocated one, where all the scalar and vector entities are stored in the same location (i.e. cell centroids); and a staggered one, where this location differs (the scalar values are stored in the cell centres and the vector values at the cell faces, as shown in Fig. 4.4). The biggest advantages of the staggered grid arrangement are:

- this is the natural way to compute the advection and diffusion terms without doing much interpolation, as the necessary variables are already stored at the faces where the computation is done;
- decoupling of the velocity–pressure fields is avoided, hence there are very good mass and momentum conservation properties;
- conservation of kinetic energy is also satisfied, which plays an important role within the simulation of highly turbulent flows.

Nevertheless, the memory footprint is much larger and, if there are singularities in one part of the domain, the numerical instabilities generated at those points can spread out and cause convergence problems throughout the entire domain easier than it is the case with a collocated grid arrangement.

On the other hand, a collocated grid arrangement offers a convenient solution when it comes to memory usage efficiency, the programming procedures and practical implementations are simplified, and the treatment of highly complex domains with discontinuities placed at the boundaries can be treated with more success, regarding local singularities and their influence on the rest of the domain. The choice of a collocated grid arrangement is adopted in the current work due to the extensive usage of multigrid procedures (see Sec. 5.1.4), where, in this

case, the same prolongation and restriction operators can be used for the entire set of variables computed.
A significant downside of the collocated grid arrangement is the very strong decoupling between the pressure and velocity fields, leading to large oscillations in both fields locally and poor conservation properties at the global scale. This problem was first tackled by C. M. Rhie and W. L. Chow [24] in 1983 and since then a remarkable amount of work has been done in this field, resulting in the numerous schemes that resolve this decoupling in an efficient way. Despite the slightly larger computational costs, conservation properties are preserved, which has led to a rise of the popularity of the collocated arrangement in the last decade.

In order to make use of the majority of the advantages of a collocated mesh arrangement within this work, the reconstruction of mass fluxes inside the level-set interface advection process is done in such a way that the well-known pressure–velocity decoupling is circumvented, without any additional memory requirements.

The exact procedure is depicted in Sec. 4.1.3, after a brief overview of the Chorin projection method (see Sec. 4.1.2) to which this reconstruction is strongly connected.

4.1.2 Fractional step method: short overview

The Chorin's fractional step method was established in the early 1960s in order to solve incompressible, isothermal flows, defined using incompressible Navier–Stokes equations:

$$\frac{\partial}{\partial t}(\rho\mathbf{u}) + \nabla\cdot(\rho\mathbf{u}\mathbf{u}) = -\nabla p + \nabla\cdot\left[\mu\left(\nabla\mathbf{u} + \nabla^T\mathbf{u}\right)\right] + \rho\mathbf{g}, \tag{4.27}$$

within which a mass conservation is enforced through the incompressibility constraint:

$$\nabla\cdot\mathbf{u} = 0 \tag{4.28}$$

A brief look onto given continuity and momentum equations reveals that the pressure is not present in all four equations, making it impossible to be solved for four unknown variables: pressure scalar field and three components of the velocity vector field. This problem is solved by decoupling the pressure field from the velocity fields in two subsequent steps, described below.
The first 'projection' step consists of choosing an appropriate temporal scheme (commonly used is the first-order explicit Euler scheme, as given in Eq. 4.29) and then introducing a projected velocity $\mathbf{u}^*$ (see Eq. 4.30), such that the momentum equations can be split into two parts, as done in Eqs. 4.31–4.32.
Eq. 4.31 can be solved for the projected velocity $\mathbf{u}^*$, as all other elements can be explicitly calculated from the previous time instance n (see Eq. 4.33). The density value at t^{n+1} is already available, as an advection of the interface for multi-phase flows must be done prior to the solution of this system of equations. For single-phase flows, the density value is constant,

thus no operation is required.

$$\frac{\rho^{n+1}\mathbf{u}^{n+1}-\rho^{n}\mathbf{u}^{n}}{\Delta t}+\nabla\cdot(\rho^{n}\mathbf{u}^{n}\mathbf{u}^{n})=-\nabla p^{n+1}+\nabla\cdot\left[\mu^{n}\left(\nabla\mathbf{u}^{n}+\nabla^{T}\mathbf{u}^{n}\right)\right]+\rho^{n}\mathbf{g} \tag{4.29}$$

$$A^{n}=(\rho^{n}\mathbf{u}^{n}\mathbf{u}^{n})$$

$$D^{n}=\left[\mu^{n}\left(\nabla\mathbf{u}^{n}+\nabla^{T}\mathbf{u}^{n}\right)\right]$$

$$G^{n}=\rho^{n}\mathbf{g}$$

$$\frac{\rho^{n+1}\mathbf{u}^{n+1}-\rho^{n+1}\mathbf{u}^{*}+\rho^{n+1}\mathbf{u}^{*}-\rho^{n}\mathbf{u}^{n}}{\Delta t}+\nabla\cdot A^{n}=-\nabla p^{n+1}+\nabla\cdot D^{n}+G^{n} \tag{4.30}$$

$$\frac{\rho^{n+1}\mathbf{u}^{*}-\rho^{n}\mathbf{u}^{n}}{\Delta t}+\nabla\cdot A^{n}=\nabla\cdot D^{n}+G^{n} \tag{4.31}$$

$$\frac{\rho^{n+1}\mathbf{u}^{n+1}-\rho^{n+1}\mathbf{u}^{*}}{\Delta t}=-\nabla p^{n+1} \tag{4.32}$$

$$\mathbf{u}^{*}=\frac{1}{\rho^{n+1}}\rho^{n}\mathbf{u}^{n}-\frac{\Delta t}{\rho^{n+1}}\nabla\cdot(\rho^{n}\mathbf{u}^{n}\mathbf{u}^{n})+\frac{\Delta t}{\rho^{n+1}}\nabla\cdot\left[\mu^{n}\left(\nabla\mathbf{u}^{n}+\nabla^{T}\mathbf{u}^{n}\right)\right]+\Delta t\mathbf{g} \tag{4.33}$$

The second 'correction' step must include the previously omitted pressure term (Eq. 4.32) and enforce the incompressibility constraint, defined as a divergence-free velocity field $\nabla\cdot\mathbf{u}^{n+1}=0$, as illustrated in Eq. 4.34. By simple mathematical manipulation, done in Eqs. 4.34–4.35, the variable-coefficient Poisson pressure Eq. 4.36 (see also Sec. 5.3) is obtained and must be solved using either direct or iterative procedures. In this work, the Full-Multigrid iterative solver is utilised.

$$\nabla\cdot\frac{\rho^{n+1}\mathbf{u}^{n+1}-\rho^{n+1}\mathbf{u}^{*}}{\Delta t}=\nabla\cdot\left[-\nabla p^{n+1}\right] \tag{4.34}$$

$$\frac{\nabla\cdot\mathbf{u}^{n+1}-\nabla\cdot\mathbf{u}^{*}}{\Delta t}=-\nabla\cdot\left[\frac{1}{\rho^{n+1}}\nabla p^{n+1}\right] \tag{4.35}$$

$$\nabla\cdot\left[\frac{1}{\rho^{n+1}}\nabla p^{n+1}\right]=\frac{\nabla\cdot\mathbf{u}^{*}}{\Delta t}=RHS^{*} \tag{4.36}$$

With the new pressure value p^{n+1} at hand, the final correction of the velocity field $\mathbf{u}^{n+1}$ is performed as shown in Eq. 4.37 (derived from Eq. 4.32):

$$\mathbf{u}^{n+1}=\mathbf{u}^{*}-\frac{\Delta t}{\rho^{n+1}}\nabla p^{n+1} \tag{4.37}$$

The evaluation of the temporal derivative in the first 'projection' phase can also be done using a second-order Adams–Bashforth scheme (see Sec. 4.1.7). The advection and diffusion term

may be evaluated explicitly, as shown above; however, it is reported that employment of an implicit or semi-implicit scheme, such as Crank–Nicolson, contributes to a reduction of the numerical instabilities. In Sec. 6.4, it is shown that the majority of numerical problems come from the discretisation of the advection term and those are successfully resolved using either one of the upwind schemes (as defined in Sec. 4.1.1.2) or HLL/HLLC approximate Riemann solvers for resolution of the shockwaves (see Sec. 4.1.5). For more information, the reader is referred to [5] and [65].

4.1.3 Cell-face flux reconstruction

Following the course of the previous chapter, where the projection of the pressure field is done in the correction step, aimed at a velocity vector field that guarantees minimal error in the kinetic energy, an additional cell-face flux reconstruction must be introduced, if this projection is performed on the collocated grid. This eliminates the well-known velocity–pressure decoupling, which is considered to be one of the major problems of this grid arrangement, as reported in [65].

The reconstruction process starts with the definition of a volume flux at the cell face in the next time instance t^{n+1}, following the formulation given in Eq. 4.37:

$$\mathcal{Q}_f^{n+1} = \mathbf{u_f}^{n+1}\mathbf{n_f} \cdot A_f = \left[\mathbf{u_f^*} - \frac{\Delta t}{\rho_f^{n+1}} \left(\nabla p^{n+1}\right)_f\right] \mathbf{n_f} \cdot A_f \tag{4.38}$$

where $\mathbf{u_f}$ represents the vector velocity field at the cell face, $\mathbf{n_f}$ the normal vector of the face pointing outwards, and A_f is the area of the face through which the flux is transported. The predicted velocity $\mathbf{u_f^*}$ and ρ_f^{n+1} are evaluated using linear interpolation of the adjacent cell-centre values:

$$\rho_f^{n+1} = \frac{1}{2}(\rho_c^{n+1} + \rho_{nb}^{n+1}) \tag{4.39}$$

$$\mathbf{u_f^*} = \frac{1}{2}(\mathbf{u_c^*} + \mathbf{u_{nb}^*}) \tag{4.40}$$

From Eq. 4.37, the projected velocity stored at the cell-centre can be written as shown in Eq. 4.41 and, by plugging this form into Eq. 4.40, the final expression for the projected velocity at the cell face is obtained (see Eq. 4.42)

$$\mathbf{u}^* = \mathbf{u}^{n+1} + \frac{\Delta t}{\rho^{n+1}} \nabla p^{n+1} \tag{4.41}$$

$$\mathbf{u_f^*} = \frac{1}{2}\left[\mathbf{u}_c^{n+1} + \frac{\Delta t}{\rho_c^{n+1}} \nabla p_c^{n+1} + \mathbf{u}_{nb}^{n+1} + \frac{\Delta t}{\rho_{nb}^{n+1}} \nabla p_{nb}^{n+1}\right] \tag{4.42}$$

After substitution of Eq. 4.42 into the expression for the cell-face flux in Eq. 4.38 and simple mathematical grouping, the following formulation for the reconstructed face flux is obtained:

$$\mathcal{Q}_f^{n+1} = \left\{ \frac{1}{2}(\mathbf{u}_c^{n+1} + \mathbf{u}_{nb}^{n+1}) + \frac{\Delta t}{2}\left[\frac{1}{\rho_c^{n+1}}\nabla p_c^{n+1} + \frac{1}{\rho_{nb}^{n+1}}\nabla p_{nb}^{n+1}\right] - \frac{\Delta t}{\rho_f^{n+1}}\left(\nabla p^{n+1}\right)_f \right\} \mathbf{n_f} \cdot A_f \quad (4.43)$$

where

$$\nabla p_f^{n+1} \cdot \mathbf{n_f} \cdot A_f = \frac{p_{nb}^{n+1} - p_c^{n+1}}{\Delta h_{c\to nb}^{\perp}} \cdot A_f; \quad \Delta h_{c\to nb}^{\perp} \in \{\Delta x, \Delta y, \Delta z\} \quad (4.44)$$

A similar expression for the single-phase flows is introduced by L. Jofre et al. [80], and some comparable work in two-phase surface-tension-dominated flows is published by J. Mencinger and I. Zun [67].
According to the work of F. N. Felten and T. S. Lund [47], the linear interpolation introduced in Eqs. 4.39 and 4.40 leads to the minimisation of interpolation errors, although the face values are often approximated using weighted distance interpolation (see Eq. 4.23), where the weighting factors differ and are not always a function of pure geometrical properties. Beside the interpolation errors, it is reported in [47] the so-called pressure errors contribute to errors in the conservation of kinetic energy and these do not depend on any step of the previously introduced reconstruction. In order to reduce the latter, it is necessary to reduce the mesh size $\mathcal{O}(\Delta x^2)$, the time-step size $\mathcal{O}(\Delta t)$ or to use a temporal scheme of a higher order of accuracy. In this work, second- and third-order temporal schemes are used to ensure good overall conservation properties.

4.1.4 Gradient reconstruction techniques

A combination of the collocated grid arrangement with the finite volume discretisation method applied onto Navier–Stokes equations requires a calculation of fluxes at cell faces with the parameters which are not available at those locations. For this reason, a calculation of gradients of any scalar entity ψ or vector component ϑ_i, as well as an interpolation of cell-centred values onto particular cell faces is of the greatest importance for the stability of numerical schemes used. In the vicinity of a discontinuity, a commonly used linear interpolation often introduces a non-physical smoothing of the interface, leading to an inaccurate propagation of the interface profile in the long-term simulation. This effect is even more emphasised in the narrow region where the discontinuity meets the various boundary cells with different boundary conditions imposed. For all these reasons, it is essential to use the entire available set of information around the cell, thus a least-square gradient reconstruction and an inverse distance weighting interpolation method, described in the following paragraphs, are commonly used.

4.1.4.1 Least-square method (LSM)

The least-square gradient reconstruction technique is based on the truncated Taylor series around the current (c) cell as shown in Eq. 4.45

$$\psi_i = \psi_c + \frac{(x_i - x_c)}{1!}\left(\nabla\psi\right)_c + \mathcal{O}(\Delta x^2) \quad (4.45)$$

where ψ_i denotes the value in the adjacent cell $i \in \{1 \dots n\}$ (see Fig. 4.5) and $(\nabla\psi)_c$ a gradient of the scalar value ψ in the current cell. Defining Eq. 4.45 in each computing cell, a over-determined system of linear equations is obtained (see Eq. 4.46)

$$\underbrace{\begin{bmatrix} x_1 - x_c & y_1 - y_c & z_1 - z_c \\ x_2 - x_c & y_2 - y_c & z_2 - z_c \\ x_3 - x_c & y_3 - y_c & z_3 - z_c \\ \vdots & \vdots & \vdots \\ x_n - x_c & y_n - y_c & z_n - z_c \end{bmatrix}}_{\mathbf{M}[n\times 3]} \underbrace{\begin{bmatrix} (\frac{\partial\psi}{\partial x})_c \\ (\frac{\partial\psi}{\partial y})_c \\ (\frac{\partial\psi}{\partial z})_c \end{bmatrix}}_{\mathbf{G}[3\times 1]} = \underbrace{\begin{bmatrix} \psi_1 - \psi_c \\ \psi_2 - \psi_c \\ \psi_3 - \psi_c \\ \vdots \\ \psi_n - \psi_c \end{bmatrix}}_{\mathbf{B}[n\times 1]} \tag{4.46}$$

where the matrix $\mathbf{M}[n \times 3]$ holds the information about geometrical properties of the underlying mesh, the vector $\mathbf{B}[n \times 1]$ the distribution of some variable ψ over that mesh and the vector $\mathbf{G}[3 \times 1]$ is the gradient $\nabla\psi$ that has to be calculated.
This over-determined system is solved using the least-square method introduced by A. Haselbacher and O. V. Vasilyev [4]

$$\mathbf{G} = (\mathbf{M}^T\mathbf{M})^{-1}\mathbf{M}^T\mathbf{B} \tag{4.47}$$

Good results have been also reported by N. Balcázar et al. [101], if this approach is used for the unstructured non-orthogonal grid.

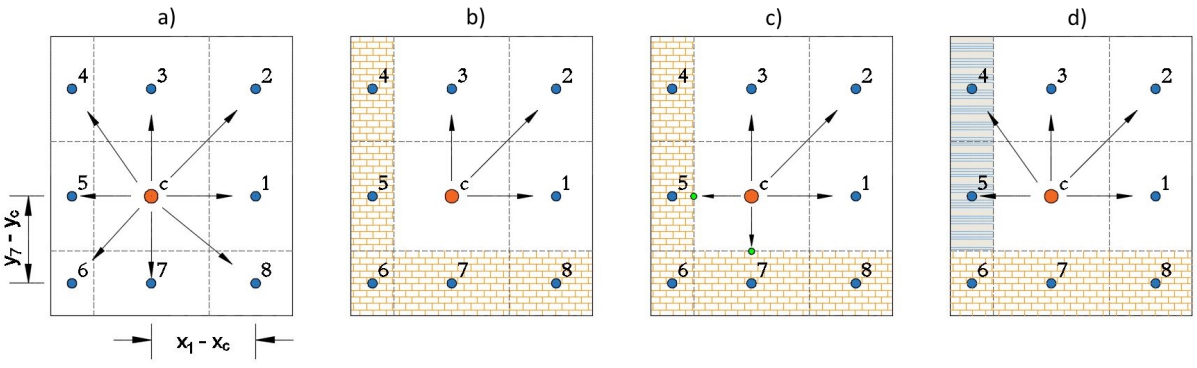


Figure 4.5: Required stencil used for the least-square gradient reconstruction in the case of the: a) inner cell, b) cell at the solid-wall boundary, c) cell at the solid-wall boundary with enriched information at the wall-face and d) cell at the mixed solid-wall and inlet boundary. The enrichment shown under c) does not bring any significant improvements in comparison to the solution under b) for the orthogonal structured mesh. In case of an unstructured non-orthogonal mesh, it is reported that this solution additionally stabilises the gradient calculation in the regions where a lot of adjacent cells must be excluded.

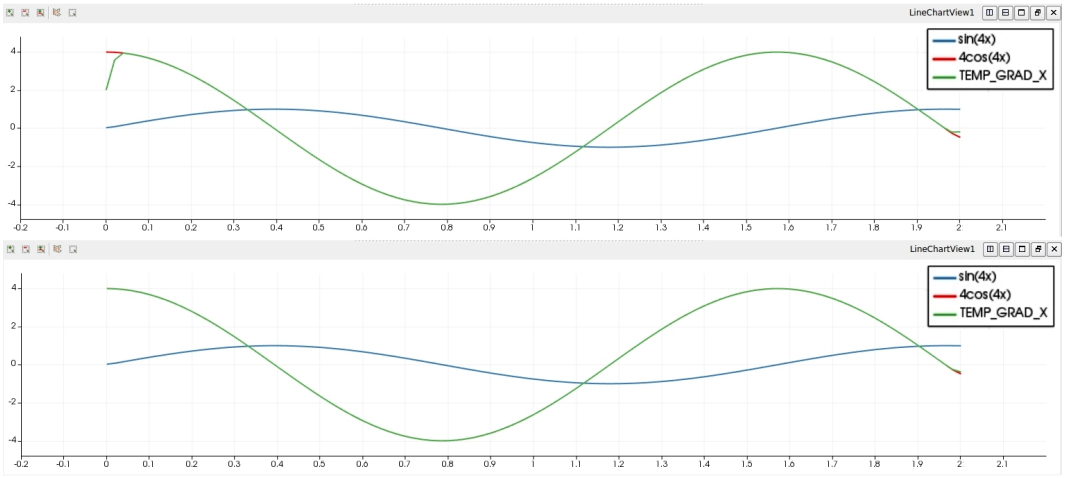


Figure 4.6: Comparison of the wide-stencil approach (see Fig. 4.5, solution under a)) and narrow-stencil approach (Fig. 4.5, solution under b)) using the analytical function $f(x) = sin(4x)$ and its analytically calculated gradient $f'(x) = 4cos(4x)$. The reconstructed gradient function using the least-square method is drawn under the name 'TEMP_GRAD_X' and it is in accordance with the analytical solution in the inner part of the domain. If no boundary cells are excluded, the deviation from the analytical solution is getting larger and it can be see in the upper-left plot. Upon the exclusion of the boundary cells (bottom-left plot), the reconstructed values tend to the analytical solution and this discrepancy goes down with the decrease of the cell size.

4.1.4.2 Inverse distance weighting (IDW)

Similar to the least-square gradient reconstruction method, inverse distance weighting (IDW) uses an enlarged sample of data points around the current point (c) to determine the value or gradient required. This is a deterministic method, which is implemented on a known set of data, with the specific weights w_i applied. These weights contain geometrical information about the proximity of the influencing point to the current point (c), and they can be calculated in several different ways. The two most common ones introduced by D. Shepard [32] are given in Eqs. 4.48 and 4.49:

$$w_i(x) = \frac{1}{d(x_c, x_i)^p} \tag{4.48}$$

$$w_i(x) = \left(\frac{max(0, R - d(x_c, x_i))}{R \cdot d(x_c, x_i)} \right)^2 \tag{4.49}$$

where $d(x_c, x_i)$ represents the distance between the current point (c) and the point (i), whose influence is taken into consideration, and the coefficient p is a real, positive number that stands for the degree of smoothing within the interpolation – the larger the coefficient p, the greater the influence of the closest neighbour. Simple mathematical analysis (see [32]) reveals that for $p \leq N_{dim}$, the IDW interpolation is divergent, thus it is recommended to choose a slightly larger value $p = N_{dim} + \varepsilon$. In the surface terrain data analysis a value of $p = 2.7$ is commonly used, leading to moderate smoothing and the inclusion of several layers of neighbouring cells.

In the case of the gradient reconstruction, this value must be somewhat larger and the results obtained suggest any value in the range $p \in \{3.7, 4.2\}$.
The modified Shepard method in Eq. 4.49 includes only the set of points which are located within the radius R, again suggesting a decrease in the influence from those points that are close to those points that are located further away.

Once the weightings are set, there is no need to generate the system of equations that should be solved, as is the case with LSM; instead the interpolated value is calculated directly as shown below:

$$U_i = \begin{cases} u_c, & d(x_c, x_i) = 0 \\ \frac{1}{\sum w_i u_i} w_i u_i, & d(x_c, x_i) > 0 \end{cases} \tag{4.50}$$

$$\nabla U_i = \begin{cases} 0, & d(x_c, x_i) = 0 \\ \frac{1}{\sum w_i (u_c - u_i)} w_i (u_c - u_i), & d(x_c, x_i) > 0 \end{cases} \tag{4.51}$$

Finally, if an interpolated single value is to be calculated, the expressions in Eq. 4.50 should be considered, while reconstruction of the gradients of an arbitrary value requires the partial gradients defined in Eq. 4.51. The reconstructed value required is hence calculated as depicted in Eqs. 4.52 and 4.53, respectively:

$$U_c = \sum_{i=1}^{i=n} U_i \tag{4.52}$$

$$\nabla U_c = \sum_{i=1}^{i=n} \nabla U_i \tag{4.53}$$

The inverse distance weighting method, as described above, is reported to be very stable, with predictable, smooth results, which can also be confirmed for the set of data calculated in this work. Nonetheless, the amount of artificial smoothness introduced is slightly larger than that which is obtained using the least-square-gradient reconstruction. For that reason, the spatial reconstruction of the bottom elevation (terrain data) is done using IDW, whereas the dynamic gradient reconstruction procedures are completed using LSM.

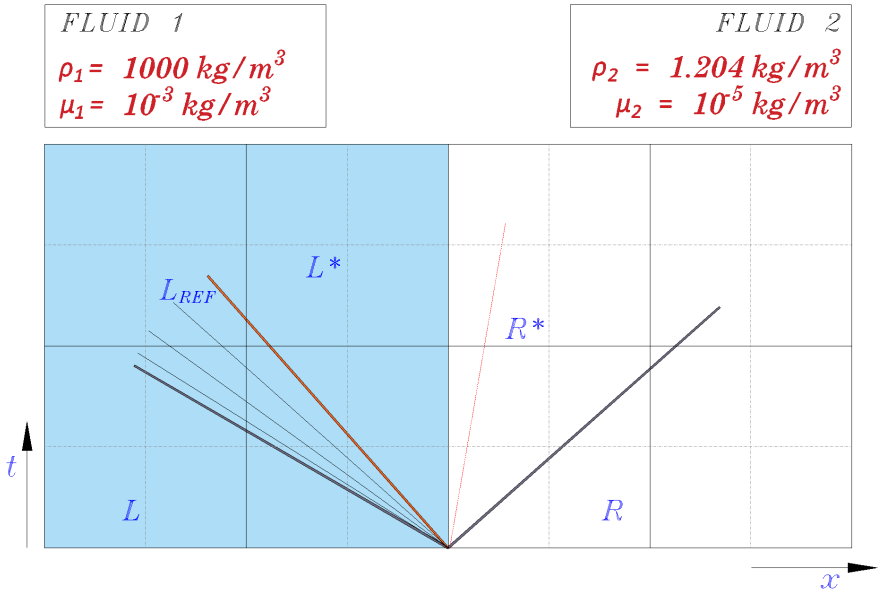


Figure 4.7: Depiction of a Riemann problem at a material discontinuity, where at the common face of two cells, one filled with water (blue coloured), one with air (white coloured), a material flux must be calculated. If a standard linear interpolation between two adjacent cell-centres is applied, a value of density, for instance, would be around $\rho_f = 500\ kg/m^3$, which does not deliver a realistic flux at that face. Therefore, several different values of fluxes are calculated based on the properties of both fluids and assigned to the zones L, L_{ref}, L*, R* and R. Further criteria are applied, as described in Secs. 4.1.5.1 and 4.1.5.2 in order to choose one of those fluxes, in such way that the mass and momentum conservation is locally and globally satisfied.

4.1.5 The Riemann problem and resolution of waves at the contact surface

The evolution of the flow across a material discontinuity, generated in the interaction of two fluids of different properties, poses a great challenge in the simulation process due to the several reasons: disrupted entropy, discontinuities in the pressure and velocity and, difficulties in mass, momentum and energy conservation both locally and globally in the entire domain. These effects are especially pronounced while calculating the fluxes at the faces that are necessary for the Finite Volume discretisation. While having only one material property across the domain, a common second order interpolation of cell-centred values onto the faces shared by those cells, delivers an accurate solution both locally and globally. By introducing a second fluid with significantly different properties compared to the first, existing fluid, the interpolation used so far may produce non-physical fluid properties at the shared faces (see example at Fig. 4.7), leading to an inaccurate computation of the fluxes, which further introduces the violation of mass and momentum conservation. Due to the present material discontinuity, different types of wave patterns may occur close to the contact and if identified correctly, that information can be used to correct the flux computation, thus preserve the accuracy and stability of the defined setup.
This problem is first introduced, and later mathematically well-established, by Bernhard

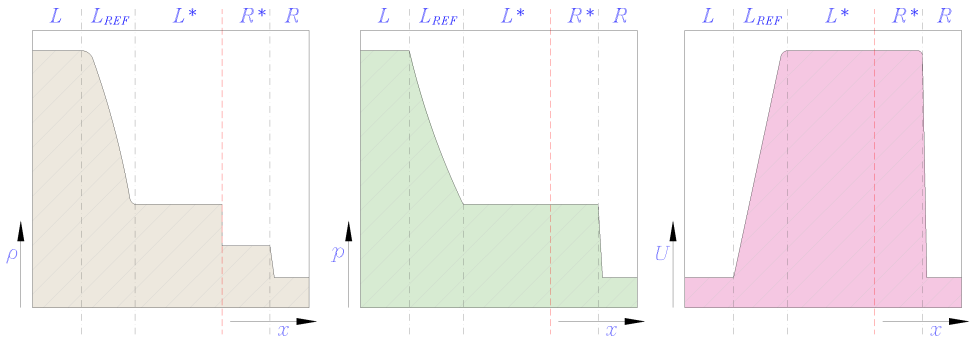


Figure 4.8: Evolution of the density (leftmost), pressure (middle) and velocity (rightmost) across a material discontinuity, in the case of two significantly different fluids, as depicted in Fig. 4.7. All five elementary regions are present, having initially a rarefaction wave followed by a shockwave in the low-density zone.

Riemann, who discovered that the flow over any discontinuity can be separated into several elementary waves, whose position and occurrence depend on the material properties of the involved materials.

As depicted in Figs. 4.7 and 4.8, two undisturbed wave states L and R preserve the original characteristics of the left and right computing cell adjacent to the discontinuity. All other states, L_{ref}, L* and R* are evolved from those two states and are physically formulated as follows: a) in the rarefaction region L_{ref}, the wave travels along the characteristic lines and it causes a strong, smooth decrease in density or any other transport property; b) the intermediate left and right regions L* and R* are characterised by the waves of constant velocity, pressure and density, whereby a sharp drop in density (see Fig. 4.8, leftmost plot) occurs at the exact location of the contact discontinuity (dashed red line, Fig. 4.7). The final shockwave occurs at the border between the two regions R* and R and it is characterised by the waves that travel perpendicularly to the characteristic line and a sudden drop in velocity, pressure and density values. The order of the waves that appear is strongly influenced by the ratios of the pressure, density and velocity in the two fluids and, as reported, there are in total four possible scenarios: 1) two rarefaction waves; 2) two shockwaves; 3) initially a rarefaction wave followed by a shockwave; and 4) initially a shockwave, followed by a rarefaction wave. This mathematical formulation was first developed by Russian mathematician S. K. Godunov [124], suggesting a numerical scheme of first-order accuracy for the calculation of the material flux across the discontinuity. This method is iterative and converges slowly on the exact solution. The next major breakthrough is made by B. van Leer [16] in 1979, by formulating a fast numerical scheme of second-order accuracy in space and time for the same problem (Monotonic Upwind Scheme for Conservation Laws). The fast iterative procedure at the basis of this method is depicted in Fig. 4.9. Since then, many different mathematical formulations have been developed aimed at the physical accuracy and numerical stability enforced by non-oscillatory behaviour at the discontinuity and the suitability of the scheme for large HPC simulation scenarios.
A large number of improvements in this field are introduced by S. F. Davis [121], P. Batten et al. [110], B. Einfeldt et al. [11], P. L. Roe [113] and E. R. Toro [38], among others. For the reason of computational efficiency, the idea of an exact Riemann solver is abandoned in this work and two advanced approximate Riemann solvers are adopted and further analysed: HLL and HLLC. The foundations of both methods can be found in [38], while some additional

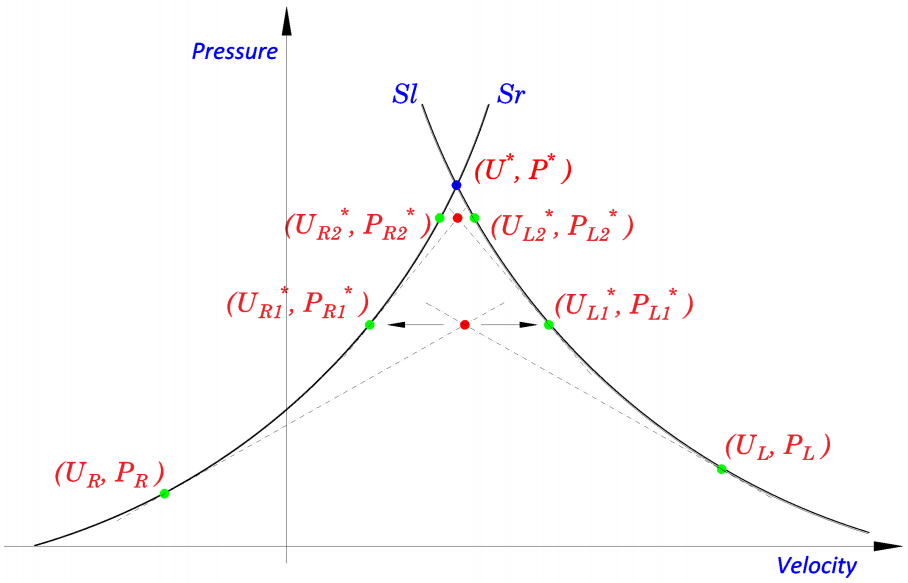


Figure 4.9: First two iterations of the root Riemann solver incorporated in the scheme invented by B. van Leer [16]. Starting from the original states on the left and right side around the discontinuity and using a modified Newton–Raphson method, only a few iterations are necessary to obtain the physically correct intermediate state (U*, P*) that plays a significant role in an accurate estimation of the flux over the discontinuity.

modifications are done according to J. Hou et al. [66]. For the sake of completeness, a short overview of both schemes is given in the following subsections.

4.1.5.1 HLL approximate Riemann solver

Instead of resolving a large range of wave states, as presented in the previous section, the approximate Riemann solver published by A. Harten, P. D. Lax, and B. V. Leer [3] – hence the name HLL – assumes the existence of two wave characteristics, separating in total three isolated wave states, as shown in Fig. 4.10 on the left-hand side. This method is robust, efficient and simple to implement; nonetheless, the accurate estimation of the wave speeds that separate those three states highly influences the accuracy of the entire scheme. The HLL configuration is reported to deliver stable and accurate results only for a hyperbolic system of equations (see [38]); it has nevertheless quite limited usage in the resolution of Navier–Stokes or Euler equations, due to the incapability of resolving the contact discontinuity, thus any material surface or shear waves can be physically very inaccurate. For all these reasons, special attention should be paid and obligatory validation of the simulated results must be performed.

A simple conservation law in 2D space (see Eq. 4.54) must be solved, having the initial and boundary conditions set.

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{F}}{\partial x} + \frac{\partial \mathbf{G}}{\partial y} = 0 \tag{4.54}$$

where $\mathbf{U}$ represents the conservative variable to solve and $\mathbf{F}$ and $\mathbf{G}$, the fluxes of the conservative variable at the cell faces in the two spatial dimensions, respectively, with ρ and $[u, v, w]$ being

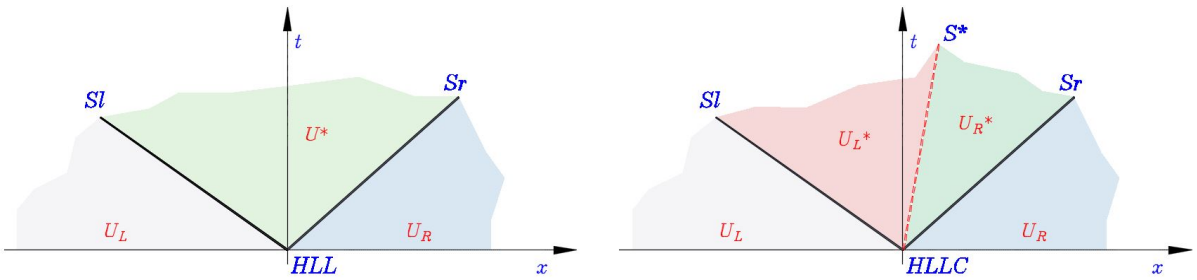


Figure 4.10: HLL and HLLC approximate Riemann solvers. Solution in the intermediate region consists of a single state U* in the case of HLL (left plot), whereas in the case of the HLLC this solution is split into two states U_L^* and U_R^* (right plot).

the interpolated values of density and velocity at that face.

$$\mathbf{U} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \end{bmatrix} \quad \mathbf{F} = \begin{bmatrix} \rho u \\ \rho uu \\ \rho vu \\ \rho wu \end{bmatrix} \quad \mathbf{G} = \begin{bmatrix} \rho v \\ \rho uv \\ \rho vv \\ \rho wv \end{bmatrix} \tag{4.55}$$

The first step in the process is an accurate estimation of the wave speeds S_L and S_R. Out of many different, efficient formulations, the two that have offered the most reliable results are given in Eqs. 4.56–4.62:

$$S_L = \begin{cases} u_R^{\perp} - 2\sqrt{gh_R}, & h_L = 0 \\ min(u_L^{\perp} - \sqrt{gh_L}, u_*^{\perp} - \sqrt{gh_*}), & h_L > 0 \end{cases} \tag{4.56}$$

$$S_R = \begin{cases} u_L^{\perp} + 2\sqrt{gh_L}, & h_R = 0 \\ max(u_R^{\perp} + \sqrt{gh_R}, u_*^{\perp} + \sqrt{gh_*}), & h_R > 0 \end{cases} \tag{4.57}$$

$$u_*^{\perp} = \frac{1}{2}(u_L^{\perp} + u_R^{\perp}) + \sqrt{gh_L} - \sqrt{gh_R}; \quad h_* = \frac{1}{4g}\left[\sqrt{gh_L} + \sqrt{gh_R} + \frac{1}{2}(u_L^{\perp} - u_R^{\perp})\right]^2 \tag{4.58}$$

with g being the gravity acceleration and $u_K^{\perp}$, $K \in (L, R)$ the fluid velocity normal to the face, interpolated from the left and right cell-centred velocity values, respectively. The order of interpolation is defined by the order of the scheme applied (see Sec. 4.1.6). The h_L and h_R represent the depths of a fluid at the L and R locations and, u_* and h_* the values of the velocity and fluid depth in the $*$ region (see Fig. 4.10, left sub-figure).
This set of equations (Eqs. 4.56–4.58) is suitable for the shallow water models in two and three spatial dimensions (see Secs. 3.4 and 3.5), whereas the formulation (Eqs. 4.59–4.62) introduced

by B. Einfeldt et al. [11], is suitable for more general applications and leads to a robust and stable scheme, as reported by P. Batten et al. [110].

$$S_L = \bar{u} - \bar{d}; \quad S_R = \bar{u} + \bar{d}; \tag{4.59}$$

$$\bar{u} = \frac{\sqrt{\rho_L}u_L + \sqrt{\rho_R}u_R}{\sqrt{\rho_L} + \sqrt{\rho_R}} \tag{4.60}$$

$$\bar{d} = \left[\frac{\sqrt{\rho_L}a_L^2 + \sqrt{\rho_R}a_R^2}{\sqrt{\rho_L} + \sqrt{\rho_R}} + \eta_2(u_R - u_L)^2\right]^{\frac{1}{2}} \tag{4.61}$$

$$a_{K\to L,R} = \sqrt{\frac{p_K}{\rho_K}}; \quad \eta_2 = \frac{1}{2}\frac{\sqrt{\rho_L}\sqrt{\rho_R}}{(\sqrt{\rho_L} + \sqrt{\rho_R})^2} \tag{4.62}$$

Having calculated the left and right wave speeds S_L and S_R, the final intercell flux $F^{HLL}_{i+\frac{1}{2}}$ can be directly calculated using the following formulation:

$$\mathbf{F}^{HLL}_{i+\frac{1}{2}} = \begin{cases} \mathbf{F}_L, & S_L \geq 0 \\ \frac{S_R\mathbf{F}_L - S_L\mathbf{F}_R + S_LS_R(\mathbf{U}_R - \mathbf{U}_L)}{S_R - S_L}, & S_L < 0 < S_R \\ \mathbf{F}_R, & S_R \leq 0 \end{cases} \tag{4.63}$$

4.1.5.2 HLLC Approximate Riemann Solver

Due to the inability to represent the material surfaces accurately, the original version of the approximate Riemann solver, HLL, has very limited usage, as mentioned in Sec. 4.1.5.1. To overcome this shortcoming, a new version of the HLL solver, within which the contact discontinuity is successfully restored, is published by E. F. Toro, M. Spruce, and W. Speares [34] under the name HLLC. Similarly to the HLL procedure, the estimation of the left and right wave speeds S_L and S_R plays an important role in the stability of the scheme, and several widely accepted ways of calculating them are published in [38], with a detailed analysis of the shortcomings of many of the others previously introduced. One of these well-established schemes is introduced by Einfeldt (see Eqs. 4.59–4.62), particularly suitable for simulation of the shockwaves at the discontinuity between two incompressible fluids. Besides that, a notably good scheme for the simulation of compressible flows is introduced by E. R. Toro [38] and this is based on pressure–velocity coupling that must be enforced within the entire domain. This scheme is capable of restoring even the rarefaction regions correctly; nonetheless its incompressible formulation has not been efficiently established to date.
Assuming that the left and right wave speeds S_L and S_R are calculated using the mathematical formulation from Einfeldt, an additional step – compared to the HLL pipeline – is calculation of a middle wave $S*$, present in the intermediate * states (see Fig. 4.10, right plot). This value

can be calculated as suggested by E. R. Toro [37]:

$$S* = \frac{S_L\psi_R(u_R^\perp - S_R) - S_R\psi_L(u_L^\perp - S_L)}{\psi_R(u_R^\perp - S_R) - \psi_L(u_L^\perp - S_L)} \tag{4.64}$$

where $u_R^\perp$ and $u_L^\perp$ represent the components of the velocity normal to the cell face and ψ takes the value of water depth $\psi = h$, if the shallow water model is used. In every other general case, ψ denotes the density of the fluid in the respective cell $\psi = \rho$.

Closely observing both sub-figures in Fig. 4.10, the integration over the volume in the HLL * region must be equal to the volume integration over both * regions in the HLLC. This statement is mathematically formulated as follows:

$$\frac{S* - S_L}{S_R - S_L}\mathbf{U}_L* + \frac{S_R - S*}{S_R - S_L}\mathbf{U}_R* = \mathbf{U}_{HLL}* \tag{4.65}$$

$$\mathbf{U}_{HLL}* = \frac{S_R\mathbf{U}_R - S_L\mathbf{U}_L + \mathbf{F}_L - \mathbf{F}_R}{S_R - S_L} \tag{4.66}$$

After the simple mathematical manipulation, the final expressions for the intermediate intercell fluxes $\mathbf{F}_L*$ and $\mathbf{F}_R*$ are obtained:

$$\mathbf{F}_L* = \mathbf{F}_L + S_L(\mathbf{U}_L* - \mathbf{U}_L) \tag{4.67}$$

$$\mathbf{F}_R* = \mathbf{F}_R + S_R(\mathbf{U}_R* - \mathbf{U}_R) \tag{4.68}$$

leading to the final estimation of the flux at the cell face $\mathbf{F}_{i+\frac{1}{2}}^{HLLC}$:

$$\mathbf{F}_{i+\frac{1}{2}}^{HLLC} = \begin{cases} \mathbf{F}_L, & S_L \geq 0 \\ \mathbf{F}_L*, & S_L < 0 < S* \\ \mathbf{F}_R*, & S_* < 0 < S_R \\ \mathbf{F}_R, & S_R \leq 0 \end{cases} \tag{4.69}$$

where the intermediate conservative variables $\mathbf{U}_L*$ and $\mathbf{U}_R*$ can be calculated as suggested in [38] and are, for the sake of simplicity, also denoted below (see Eq. 4.70)

$$\mathbf{U}_{K\to L,R}^{*(east)} = \rho_K\frac{S_K - u_K}{S_K - S*}\begin{bmatrix} 1 \\ S* \\ v_K \\ w_K \end{bmatrix} \qquad \mathbf{U}_{K\to L,R}^{*(north)} = \rho_K\frac{S_K - v_K}{S_K - S*}\begin{bmatrix} 1 \\ u_K \\ S* \\ w_K \end{bmatrix} \tag{4.70}$$

Depending on the case simulated, this approach may trigger some local instabilities close to the solid boundary walls, which are manifested through the locally preserved oscillatory behaviour

in the pressure and velocity fields. For that reason, an additional scheme, introduced by J. Hou et al. [66], is implemented as originally proposed for the 2D simulation setups, including the slope source terms and terrain reconstruction, whereas for all the 3D simulated scenarios further extension and adaptation is necessary (see Eq. 4.72). As can be seen in Sec. 6, the results obtained are smooth and free of any local inconsistencies.
The main differences in the second approach can be seen in the calculation of the intermediate intercell fluxes, whereas the estimation of the left and right wave speeds is almost identical as in the case of the HLL approach.

$$\mathbf{F}^{*} = \frac{S_R\mathbf{F}_L - S_L\mathbf{F}_R + S_LS_R(\mathbf{U}_R - \mathbf{U}_L}{S_R - S_L} \tag{4.71}$$

$$\mathbf{F}_L^{*(i+\frac{1}{2})} = \begin{bmatrix} \mathbf{F}_1^* \\ \mathbf{F}_2^* \cdot n_x \\ \mathbf{F}_2^* \cdot n_y \\ \mathbf{F}_2^* \cdot n_z \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ v_L\mathbf{F}_1^* \\ w_L\mathbf{F}_1^* \end{bmatrix} \tag{4.72}$$

The first part of Eq. 4.72 represents the standardised calculation of the fluxes, as defined in Eq. 4.55, whereas the second part depicts the contribution of the shear waves that are commonly generated at the surface discontinuity. This additional term significantly stabilises the numerical scheme in the vicinity of the material surfaces, including also the solid boundary walls.

4.1.6 High-order spatial schemes and flux limiters

To achieve better accuracy while solving partial differential equations (PDE), high-order temporal schemes implemented (see Sec. 4.1.7) in combination with high-order spatial schemes are often a preferred choice. This family of schemes produces very accurate results in smooth regions of the domain and reduces the smearing of the solution introduced by artificial diffusion, if first-order schemes are adopted. A frequent disadvantage of such an approach is often the appearance of non-physical oscillations (so-called spurious currents) in the region where shockwaves are present. According to Godunov's theorem, only first-order schemes can guarantee boundedness and monotonicity, thus non-oscillatory behaviour; therefore, if higher-order schemes are to be used, either a slope- or flux-reconstruction at the finite-volume interface must be conducted.
In the field of flux-reconstruction methods, the most frequently ones used are Godunov's schemes, which solve a Riemann problem exactly at the boundary between two neighbouring cells. These methods are quite expensive, thus an approximate solution has been investigated for the last two decades, resulting in numerous Riemann- and non-Riemann-type solvers. The two Riemann-type methods used in this work are the HLL and HLLC approximate solvers (see Sec. 4.1.5), which are suitable for parallelisation, as they do not require a global Jacobi matrix to be assembled. In the category of non-Riemann-type solvers, a flux-reconstruction technique also used here, is first published in [101] and is briefly explained in Sec. 4.1.3. While Riemann-type solvers act independently, delivering a reconstructed flux at each inter-cell boundary, the later technique must be combined with one of the finite difference schemes (e.g.

the most commonly used are first and second order accurate upwind methods, see Sec. 4.1.1) to calculate the same quantity. Nonetheless, it should be mentioned that in several simulation scenarios that include a high-density ratio (e.g. water–air interaction), the flux-reconstruction scheme combined with the upwind scheme of first-order accuracy caused slight oscillatory behaviour in the light density phase, far away from the water–air interface, delivering smooth results elsewhere. In order to eliminate such artefacts while convecting the level-set scalar field, this reconstruction technique is combined with different slope-reconstruction methods.
The foundations of such slope-limiting techniques are established by van Leer [144] within the well-known MUSCL scheme (Monotonic Upwind Scheme for Conservation Laws), which, as reported, can produce accurate results even in regions that indicate the presence of discontinuities and high gradients. The basic idea consists of replacing the cell-centred values ϕ_C and ϕ_{NB} (see Fig. 4.11) with the reconstructed left and right states ϕ_C^L and ϕ_{NB}^R while calculating the flux at the inter-cell boundary. Those cell states can be combined with an arbitrary finite volume scheme or used as an input for the Riemann-type solvers. According to the MUSCL scheme from [144], a limited downwind scheme should be used, formulated as depicted in Eq. 4.75.

$$\phi_{i+1/2}^L = \phi_i + \frac{1}{2}\theta(r_i)(\phi_{i+1} - \phi_i) \tag{4.73}$$

$$\phi_{i+1/2}^R = \phi_{i+1} - \frac{1}{2}\theta(r_i)(\phi_{i+2} - \phi_{i+1}) \tag{4.74}$$

$$r_i = \frac{\phi_i - \phi_{i-1}}{\phi_{i+1} - \phi_i}, \tag{4.75}$$

where $\theta(r_i)$ represents the limiter function of the downwind slope and it takes the value zero if $r_i < 0$, otherwise $\theta(r_i) = 1$. This concept is slightly modified in the current work and its mathematical formulation is given in Eqs. 4.78–4.82.
Three described slope-reconstruction techniques, outlined in Eqs. 4.80–4.82 have also a Total-Variation-Diminishing property, which means that they are capable of preserving monotonicity (i.e. they do not generate new local extrema); the same is true with schemes of first-order accuracy, but with significantly more accurate solutions in the smooth parts of the domain.

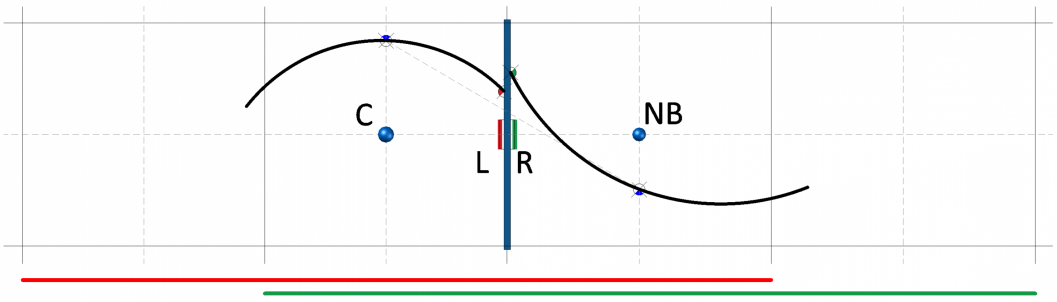


Figure 4.11: Cell-centered values ϕ_C and ϕ_{NB} (blue markers), and their reconstructed states ϕ_C^L and ϕ_{NB}^R (red and green marker, respectively). If an average value between two subsequent cell-centred values was used, the value at the inter-cell boundary would be estimated much lower than the one that can be obtained within the reconstruction process.

Let the Total Variation (TV) quantity be defined as:

$$\mathrm{TV}(\phi^n) = \sum_{f=0}^{m} |\phi_{nb}^n - \phi_c^n| \tag{4.76}$$

where m is the number of the adjacent nodes with respect to the current cell. The quantity TV measures the amount of oscillations introduced by the advection scheme, and in order to satisfy the Total-Variation-Diminishing (TVD) property, the following condition must be fulfilled:

$$\mathrm{TVD}(\phi^{n+1}) < \mathrm{TVD}(\phi^n) \tag{4.77}$$

This requirement is depicted graphically in Fig. 4.13, showing the exact region (in blue), within which every scheme must be Total-Variation-Diminishing and Total-Variation-Bounded (TVB). These findings are published by P.-K. Sweby [112]. The five schemes represented in

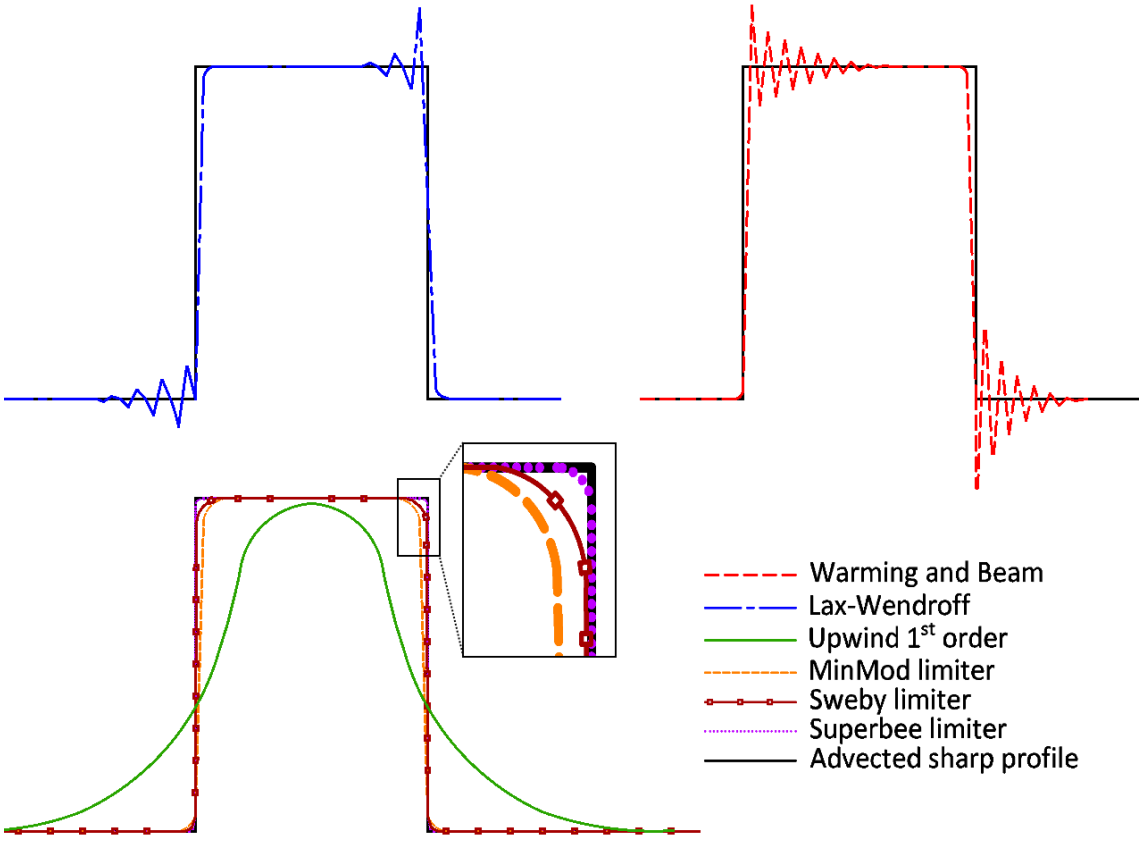


Figure 4.12: Advection of a rectangular domain in horizontal direction, using the first-order upwind scheme, two second-order schemes - Lax–Wendroff and Warming & Beam, and three different slope-reconstruction techniques: MinMod, Sweby and Superbee. The upwind scheme is monotone and non-oscillatory, introducing an interface smearing. Both second-order schemes are oscillatory either before or after the discontinuity, but much more accurate in the smooth regions of the domain, in comparison to the upwind scheme. Three slope-reconstruction techniques are Total-Variation-Diminishing, producing satisfactory results with a minimal smearing in the narrow region around the discontinuity.

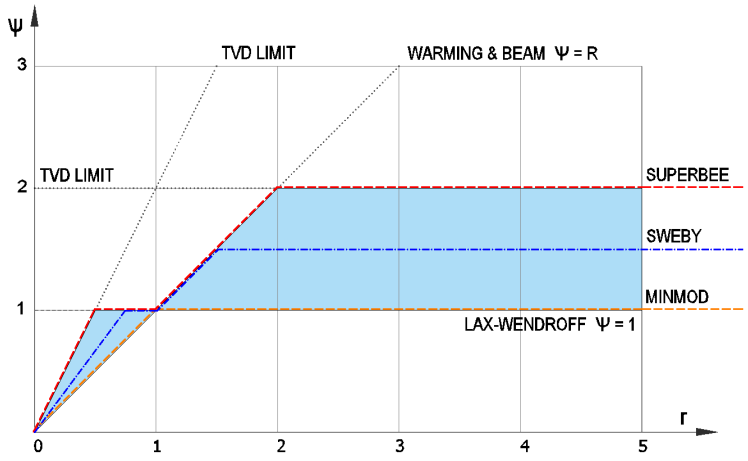


Figure 4.13: Total-Variation-Diminishing diagram published in [112]. The five different schemes tested in Fig. 4.12 are also depicted here, showing the coincidence between the TVD property and essential non-oscillatory behaviour. Many other limiter functions have been invented since the first slope- and flux- reconstruction technique proposed by van Leer and they are still in use in different application areas. For more details, an interested reader is referred to the standard literature.

Fig. 4.13 are also tested in the case of the pure advection of a rectangular domain (see Fig. 4.12) and it can be easily seen that both Lax–Wendroff and Warming & Beam exhibit oscillatory behaviour either before or after the discontinuity. This coincides with the non-convex regions of the Sweby Diagram (outside the blue-coloured part), whereas all the remaining schemes – MinMod, Sweby and Supebee slope reconstructions produce smooth, bounded results that entirely fit into the TVD zone.

$$\phi^{L}_{i+1/2} = \phi_i + \frac{1}{2}\Delta x \cdot \text{LimiterFunction}(\text{SLOPE A}_C, \text{SLOPE B}_C) \tag{4.78}$$

$$\phi^{R}_{i+1/2} = \phi_{i+1} - \frac{1}{2}\Delta x \cdot \text{LimiterFunction}(\text{SLOPE A}_{NB}, \text{SLOPE B}_{NB}) \tag{4.79}$$

Slopes

$$\text{SLOPE A}_C = \frac{\phi_{i+1} - \phi_i}{\Delta x}$$

$$\text{SLOPE B}_C = \frac{\phi_i - \phi_{i-1}}{\Delta x}$$

$$\text{SLOPE A}_{NB} = \frac{\phi_{i+2} - \phi_{i+1}}{\Delta x}$$

$$\text{SLOPE B}_{NB} = \frac{\phi_{i+1} - \phi_i}{\Delta x}$$

Limiter functions:

$$\text{MinMod}(a,b) = \begin{cases} 0, & a \cdot b \leq 0, \\ a, & |a| \leq |b|, \\ b, & \text{otherwise}, \end{cases} \tag{4.80}$$

$$\text{Sweby}(a,b) = \begin{cases} 0, & a \cdot b \leq 0, \\ sign(a) \cdot \max(|a|,|b|), & |a| \leq 1.5|b| \ \& \ |b| \leq 1.5|a|, \\ 1.5 sign(a) \cdot \min(|a|,|b|), & \text{otherwise}, \end{cases} \tag{4.81}$$

$$\text{Superbee}(a,b) = \begin{cases} 0, & a \cdot b \leq 0, \\ sign(a) \cdot \max(|a|,|b|), & |a| \leq 2|b| \ \& \ |b| \leq 2|a|, , \\ 2 sign(a) \cdot \min(|a|,|b|), & \text{otherwise}, \end{cases} \tag{4.82}$$

In case of a uniform, orthogonal mesh that conforms with the coordinate axes, all three previously defined limiters can be straightforwardly applied within a multi-dimensional simulation scenario, by simple iterating over the cell faces. This particular extension is stable and supported by the fact that a 4-cells-wide stencil (see Fig. 4.11) is used in every direction. Having an unstructured, non-orthogonal mesh combined with the FVM approach, a maximum and minimum region must be defined based on all the cell faces, and only then the limiting can be performed at each face within those boundaries (M. E. Hubbard [85]). In case of the FEM approach, it is proved that this straightforward extension on non-orthogonal, unstructured meshes results in a reduction of order of accuracy in smooth regions of the domain (D. Kuzmin [30]), thus additional steps must be conducted. Interested user is referred to T. Barth and D. Jespersen [132] and P. Batten et al. [109].

4.1.7 Temporal schemes of the first, second and third orders of accuracy

The complexity of the spatial and temporal discretisation within one numerical procedure is a key factor that significantly affects the stability, conservation and convergence of the whole process. The variation in how these schemes are built depends on the process that is being simulated; thus it is difficult to name those schemes that would generally perform very well, without discussing their disadvantages in every particular case. For that reason, this Sec. will encapsulate only a brief overview of several temporal schemes, which are, according to the broad scientific society, extremely important for multi-phase three-dimensional flows within which shock and rarefaction waves occur. Different temporal schemes have been applied in different parts of the computational kernel, bearing in mind their numerical limitations. Explicit first-order Euler and second-order Adams–Bashforth schemes have been utilised for the transport of momentum (NSE), whereas within the advection of the scalar, which is used to represent the interface between two fluids, second and third order Total-Variation-Diminishing Runge–Kutta schemes are applied. The sensitivity analysis of several time schemes with respect to the preservation of the interface shape and the long-time stability is given in Sec. 3.6.3.3.

4.1.7.1 First-order explicit Euler and second-order Adams–Bashforth schemes

Starting from the general advection-diffusion equation, given in the form:

$$\frac{\partial \phi}{\partial t} + \nabla \cdot (\phi \mathbf{u}) = \frac{\mu}{\rho} \Delta \phi \tag{4.83}$$

the Euler forward explicit temporal discretisation is applied to the time derivative, and the second-order central difference spatial scheme is also applied to both the diffusion and convection term, yielding the form:

$$\frac{\phi_i^{n+1} - \phi_i^n}{\Delta t} + \mathbf{u}_i \frac{\phi_{i+1}^n - \phi_{i-1}^n}{2\Delta x} = \frac{\mu}{\rho} \frac{\phi_{i+1}^n - 2\phi_i^n + \phi_{i-1}^n}{\Delta x^2} \tag{4.84}$$

Eq. 4.84 can be further rearranged:

$$\phi_i^{n+1} = (1 - 2\mathcal{D})\phi_i^n + \left(\mathcal{D} - \frac{\mathcal{C}}{2}\right) \phi_{i+1}^n + \left(\mathcal{D} + \frac{\mathcal{C}}{2}\right) \phi_{i-1}^n \tag{4.85}$$

where $\mathcal{C}$ and $\mathcal{D}$ are non-dimensional parameters that impose the time-step size δt, such that the transport value ϕ is conveyed within the limits of the spatial interval Δx, if the transport is driven by pure convection and diffusion, respectively. The third, fairly important parameter is the ratio of $\mathcal{C}$ and $\mathcal{D}$, called the Peclet number ($\mathcal{P}_e$), and it can be seen from the middle term in Eq. 4.85 that this value must be less than 2 for stability reasons, if the central difference scheme is used.

$$\mathcal{D} = \frac{\mu \Delta t}{\rho \Delta x^2}, \qquad \mathcal{C} = \frac{\mathbf{u} \Delta t}{\Delta x} \qquad \textit{and} \qquad \mathcal{P}_e = \frac{\mathcal{C}}{\mathcal{D}} = \frac{\rho \mathbf{u} \Delta x}{\mu} < 2 \tag{4.86}$$

Restrictions on these three numbers impose well-established limitations, formulated as follows:

$$\Delta t_{conv} \leq \frac{\Delta x}{|\mathbf{u}_{i,max}|}, \qquad i \in \{1,2,3\} \tag{4.87}$$

This condition is further restricted in the case of the appearance of strong shockwaves, which travel along characteristic lines with the speed of propagation $\mathbf{c}$, leading to the formulation:

$$\Delta t_{riemann} \leq \frac{\Delta x}{|\mathbf{u}_{i,max}| + \mathbf{c}}, \qquad i \in \{1,2,3\} \tag{4.88}$$

The straightforward limitation that arises from the limitation of the diffusion non-dimensional parameter $\mathcal{D}$ results in:

$$\Delta t_{diff} \leq \frac{\rho \Delta x_i^2}{\mu}, \qquad i \in \{1,2,3\} \tag{4.89}$$

If the flow velocity is reduced in such way that capillary forces, surface tension and gravity play an important role, two additional restrictions must be imposed:

$$\Delta t_{grav} \leq \left[\frac{\Delta z}{|g|}\right]^{\frac{1}{2}}, \qquad \Delta t_{surf} \leq (\Delta x_i)^{\frac{3}{2}} \sqrt{\frac{\rho_1 + \rho_2}{4\pi\sigma}} \qquad i \in \{1,2,3\} \tag{4.90}$$

In order to take into consideration all the above-mentioned restrictions, a minimum value among them all is chosen:

$$\Delta t = \gamma \cdot min\left\{\Delta t_{conv},\ \Delta t_{riemann},\ \Delta t_{diff},\ \Delta t_{grav},\ \Delta t_{surf}\right\} \tag{4.91}$$

with $0 \leq \gamma \leq 1$ being a safety factor, introduced to additionally control the maximum time-step, as the stability criteria listed are necessary, but not sufficient (the higher-order Runge–Kutta schemes further reduce the maximum value of the CFL condition (see [123]), without any insight into the grid-size analysis previously done).
Similar analysis could be done for the Adams–Bashforth second-order accurate scheme in time, where the convective and diffusion terms are taken from two subsequent time intervals (see Eq. 4.92), but the vast amount of tests show that the conditions previously derived will be only slightly relaxed, depending on the specific topic simulated.

$$\phi_i^{n+1} = \phi_i^n + \frac{1}{2}\frac{\Delta t}{\Delta x}\left[3 \cdot (A+D)^{(n)} - (A+D)^{(n-1)}\right] \tag{4.92}$$

4.1.7.2 Runge–Kutta higher-order schemes

The classical Runge–Kutta method is a multi-step method of higher-order used in the time discretisation of ordinary differential equations. This method is also referred to as a step-wise Euler method, where in every subsequent step the estimated solution of the previous

intermediate step is used, leading to the better approximation to the exact solution and reduction of the truncation error, up to the required precision.
Let the Φ be a value advected with the velocity $\mathbf{u}$, following the formulation:

$$\frac{\partial \Phi}{\partial t} + \nabla \cdot (\mathbf{u}\Phi) = 0 \tag{4.93}$$

$$\frac{\Phi^{n+1} - \Phi^n}{\Delta t} + \nabla \cdot (\mathbf{u}\Phi) = 0 \tag{4.94}$$

$$\Phi^{n+1} = \Phi^n + \Delta t \nabla \cdot (\mathbf{u}\Phi) \tag{4.95}$$

$$\mathcal{L}(\Phi, t_k) = \nabla \cdot (\mathbf{u}\Phi) \tag{4.96}$$

An operator $\mathcal{L}(\Phi, t_k)$ in Eq. 4.96 represents the convection through the cell surface in a discrete amount of time Δt and determines in general whether the numerical treatment is going to be explicit ($k = n$) or implicit ($k = n + 1$).
A step-wise procedure, depicted in the following steps, uses an explicit formulation:

$$\Phi^* = \alpha_1 \Phi^n + \beta_1 \Delta t \cdot \mathcal{L}(\Phi^n, t_n) \tag{4.97}$$

$$\Phi^{**} = \alpha_2 \Phi^* + \beta_2 \Delta t \cdot \mathcal{L}(\Phi^*, t_{n+1/2}) \tag{4.98}$$

$$\Phi^{***} = \alpha_3 \Phi^{**} + \beta_3 \Delta t \cdot \mathcal{L}(\Phi^{**}, t_{n+1/2}) \tag{4.99}$$

$$\Phi^{n+1} = \alpha_4 \Phi^{***} + \beta_4 \Delta t \cdot \mathcal{L}(\Phi^{***}, t_n), \tag{4.100}$$

This set of equations can be written in the general form:

$$\Phi^n = \sum_{k=0}^{k=n-1} \left[\alpha_{nk} \Phi^k + \Delta t \beta_{nk} \mathcal{L}(\Phi^k)\right] \quad \text{and} \quad \text{CFL} = \min \frac{\alpha_{nk}}{\beta_{nk}} \tag{4.101}$$

where the CFL represents the minimum sufficient stability condition, should an explicit scheme be used. If the two parameters α_{nk} and β_{nk} are non-negative, Φ^n in Eq. 4.101 is a convex combination of first-order forward Euler operators, thus the entire formulation results in a monotone, bounded solution and no local extrema can be generated.
As published in [26] and further investigated by S. Gottlieb and C.-W. Shu [123], there is no guarantee that the coefficients α_{nk} and β_{nk} will always remain positive, yielding oscillatory behaviour, if higher-order schemes have been used. In order to achieve monotonicity and boundedness, and at the same time harvest the benefits of the properties of higher-order schemes, the operator $\mathcal{L}(\Phi_s, t_n)$ must be modified from a forward to a backward one, generating a total-variation-diminishing (TVD) scheme in time, which, if combined with a higher-order TVD spatial scheme (see Sec. 4.1.6), provides at least the second-order accuracy and essentially non-oscillatory behaviour in the vicinity of a discontinuity. The concept of total variation diminishing is thoroughly explained in Sec. 4.1.6.

Runge–Kutta 2:

$$\Phi^* = \Phi^n + \Delta t \cdot \mathcal{L}(\Phi^n) \tag{4.102}$$

$$\Phi^{n+1} = \frac{1}{2}\Phi^n + \frac{1}{2}\Phi^* + \frac{1}{2}\Delta t \cdot \mathcal{L}(\Phi^*) \tag{4.103}$$

$$\boxed{\Rightarrow \Phi^{n+1} = \Phi^n + \frac{1}{2}\Delta t \cdot \mathcal{L}(\Phi^n) + \frac{1}{2}\Delta t \cdot \mathcal{L}(\Phi^*)}, \tag{4.104}$$

Runge–Kutta 3:

$$\Phi^* = \Phi^n + \Delta t \cdot \mathcal{L}(\Phi^n) \tag{4.105}$$

$$\Phi^{**} = \frac{3}{4}\Phi^n + \frac{1}{4}\Phi^* + \frac{1}{4}\Delta t \cdot \mathcal{L}(\Phi^*) \tag{4.106}$$

$$\Phi^{n+1} = \frac{1}{3}\Phi^n + \frac{2}{3}\Phi^{**} + \frac{2}{3}\Delta t \cdot \mathcal{L}(\Phi^{**}) \tag{4.107}$$

$$\boxed{\Rightarrow \Phi^{n+1} = \Phi^n + \frac{1}{6}\Delta t \cdot \mathcal{L}(\Phi^n) + \frac{1}{6}\Delta t \cdot \mathcal{L}(\Phi^*) + \frac{2}{3}\Delta t \cdot \mathcal{L}(\Phi^{**})}, \tag{4.108}$$

As reported in S. Gottlieb and C.-W. Shu [123], isolated utilisation of either a spatial or a temporal TVD scheme may suffice, if no severe shock discontinuities are present, but it does not guarantee monotonicity unconditionally, thus special care should be taken.
For the sake of completeness, the Runge–Kutta second- and third-order schemes which are implemented and utilised in this work are listed in Eqs. 4.102–4.108.

4.2 Coupling strategies

4.2.1 Micro–macro scale coupling: Darcy and Navier–Stokes model

In this section the coupling between the macroscopic Darcy's Law (see Sec. 3.3) and the microscopic Navier–Stokes model (see Sec. 3.2) is illustrated, allowing calculation of discrete permeability values on the meso-scale, instead of conducting dozens of fairly expensive in-situ laboratory measurements. As shown in Fig. 4.14, this coupled model operates on three different scales: the micro-scale, where the Navier–Stokes model is used as a fundamental law; the meso-scale, where the appropriate volume averaging has been performed; and the macro-scale, where the Darcy model is applied in combination with a particular transport equation.
In order to predict a flow through a homogenous, isotropic porous medium with the variable porosity and permeability values, S. Whitaker [130] proposed a volume-averaging of the Navier–Stokes equations (VANS), starting from the general form:

$$\frac{\partial}{\partial t}(\rho \mathbf{u}) + \nabla \cdot (\rho \mathbf{u}\mathbf{u}) = -\nabla p + \nabla \cdot \left[\mu \left(\nabla \mathbf{u} + \nabla^T \mathbf{u}\right)\right] + \rho \mathbf{g} \tag{4.109}$$

where ρ and μ represent the density and kinematic viscosity of a fluid that flows through a porous medium, p is the pressure, $\mathbf{u}$ is the velocity vector field and g is the gravity acceleration.

Assuming an arbitrary physical quantity Ψ, a volume averaging can be done by integrating over the pore volume V_i and the total volume V_s of a representative element (REV), resulting in an intrinsic and superficial values, respectively:

$$\langle\psi\rangle_i = \frac{1}{V_i}\int_{V_i}\psi_i dV \tag{4.110}$$

$$\langle\psi\rangle_s = \frac{1}{V_s}\int_{V_i}\psi_i dV \tag{4.111}$$

$$\langle\psi\rangle_s = \epsilon\,\langle\psi\rangle_i \tag{4.112}$$

The correlation between the superficial and intrinsic average (see Eq. 4.112) defines the ratio of the pore volume V_i and total volume V_s. It is referred to as a porosity ϵ of a soil sample and can be understood as its capacity to retain or transmit a corresponding amount of fluid in between the soil particles. Exploiting the fact that every parameter can be written as a sum of its averaged and fluctuating component $\Psi = \langle\Psi\rangle_i + \widetilde{\Psi}$ and starting from Eq. 4.109, S. Whitaker [129] derives the following form of the volume-averaged Navier–Stokes equations:

$$\rho_i\frac{\partial}{\partial t}\langle\mathbf{u}_i\rangle + \rho_i\langle\mathbf{u}_i\rangle\nabla\cdot\langle\mathbf{u}_i\rangle + \overbrace{\frac{\rho_i}{\epsilon_i}\nabla\cdot\langle\widetilde{\mathbf{u}}_i\widetilde{\mathbf{u}}_i\rangle}^{\text{VOLUME FILTER}} = -\nabla\langle p_i\rangle + \rho_i g + \tag{4.113}$$

$$\overbrace{\mu_i\nabla^2\langle\mathbf{u}_i\rangle}^{\text{BRINKMAN CORRECTION}} \overbrace{+\mu_i\frac{1}{\epsilon_i}\nabla\epsilon_i\nabla\langle\mathbf{u}_i\rangle + \mu_i\frac{1}{\epsilon_i}\langle\mathbf{u}_i\rangle\nabla^2\epsilon_i}^{\text{BOUNDARY REGIONS}} +$$

$$\underbrace{\frac{1}{V_i}\int_{A_{is}}\mathbf{n}_{is}\cdot(-\mathbf{I}\widetilde{p_i} + \mu_i\nabla\widetilde{\mathbf{u}_i})\,dA}_{\text{SURFACE FILTER}}$$

Closely observing Eq. 4.113, the terms grouped under the boundary regions exist only if there are solid-fluid interfaces where the gradient of porosity exists, and they are, together with the Brinkman correction, resolved using a *momentum jump condition* [130]. Assuming a steady state analysis, the terms grouped under surface filter can be represented using the Forchheimer approximation (T. Zhu et al. [136]):

$$\frac{1}{V_i}\int_{A_{is}}\mathbf{n}_{is}\cdot(-\mathbf{I}\widetilde{p_i} + \mu_i\nabla\widetilde{\mathbf{u}_i})\,dA = -\frac{\mu}{\kappa}(1 + F|\langle\mathbf{u}\rangle_i|)\langle\mathbf{u}\rangle_s \tag{4.114}$$

If the Reynolds number is small enough ($R_e \leq 10^{-3}$) the second term on the left side in Eq. 4.113 (convective term) as well as $F|\langle\mathbf{u}_i\rangle|$ in Eq. 4.114 can be neglected. Furthermore, the first and second terms on the right side (pressure and gravity terms, Eq. 4.113) are grouped together, leading to the final form used on the macro scale:

$$0 = -\nabla\langle p_i\rangle - \frac{\mu}{\kappa}\langle\mathbf{u}_s\rangle \tag{4.115}$$

$$\langle\mathbf{u}_s\rangle = \frac{\kappa}{\mu}\nabla\langle p_i\rangle \tag{4.116}$$

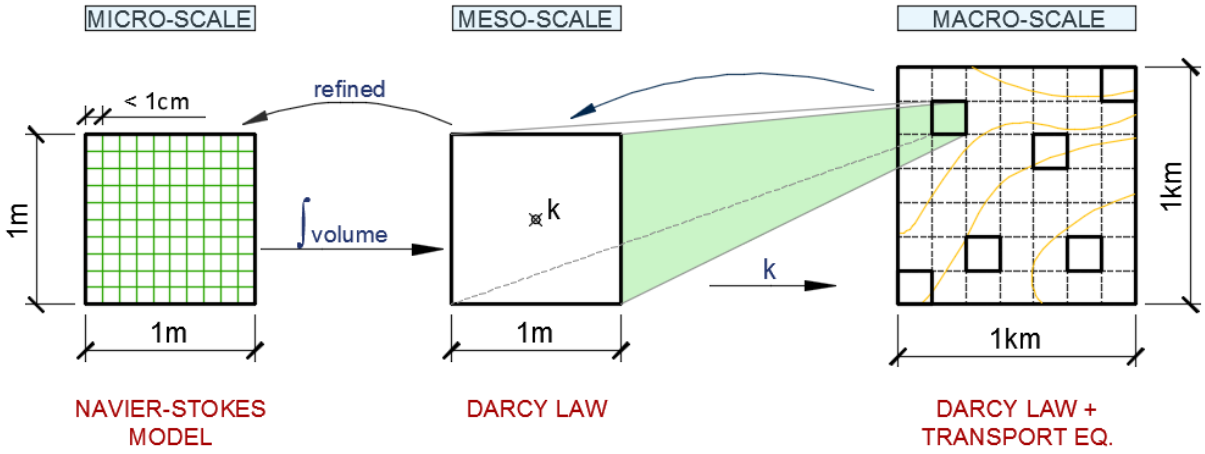


Figure 4.14: Schematic representation of coupling schema between a micro-scale domain (left-hand-side plot) of typical size 1 m × 1 m × 1 m (with grid resolution h of $\mathcal{O}$(1cm) for the numerical treatment(128 × 128 × 128)) and a macro-scale domain (right-hand-side plot) of surface size 1 km × 1 km and hundreds metres in depth, accomplished via a so-called meso-scale (middle plot) of fixed size that is equivalent to the size of an experimental pit on the macro-scale.

Taking into account that only one incompressible fluid is considered, generating a flow through porous media, a variability of density and viscosity values can be neglected. Furthermore, constant permeability and porosity values are assumed on the macro scale, although in general these can vary in space and time. Following all the assumptions made, the final form of the volume-averaged Navier–Stokes momentum equations appears to be equal to the Darcy law on the macro scale (see Eq. 4.116). In case that an unsteady porous media flow is considered, these assumptions seem to be too strict, leading to the increase of relative error while calculating superficial velocities. Two other methods used instead are a virtual mass approach (C. K. Sollitt and R. H. Cross [23] and K. R. Rijagopal [74], among others) and a volume-averaging of the kinetic equation (VAKE), published by T. Zhu et al. [136], who compares all three methods with the DNS results of a carefully chosen simulation scenario, and proved that all three methods are very accurate for the steady state analyses, whereas a virtual mass approach and VAKE are more in concordance with the DNS outcome for an unsteady porous media flow.

Calculating the superficial velocity $\langle \mathbf{u} \rangle_s$ and volume-averaged pressure field $\langle p \rangle_i$, and substituting those values into Eq. 4.116, a discrete value of permeability κ is obtained on the meso-scale (see Fig. 4.14, middle plot), and as such is used on the macro-scale. In order to conform to the concept of porous media as a continuum on a large scale, a Kriging interpolation method (i.e. the Kolmogorov–Wiener prediction) should be applied, providing, as reported in [82], [68] and [28], the best linear unbiased estimation of interpolated values at the unobserved spatial points of the domain, based on available, previously calculated, discrete permeability values. According to [105], this method is widely used in the field of spatial analysis, yielding a continuous estimated field of an analysed parameter, on the basis of which various further analysis, for instance, chemical or pollution transport or even chemical reactions between the porous material and a chemical present in the fluid, can be performed. Further research on

2D SWE MODEL		Exchange Interface Border	3D SWE MODEL		EXCHANGE 2D ⇔ 3D
Terrain bed z_B	*given*		Terrain bed z_B	*given*	DONE
Density ρ	*given*		Density ρ	*given*	DONE
Depth H_{2D}	*Calculated* *Shallow Water Equation*		Depth H_{2D}	*Calculated* *Shallow Water Equation*	DONE
Mean velocity $\overrightarrow{U_{2D}}$	*Calculated* *Shallow Water Equation*		Static Pressure $p_{st} = f(H_{2D})$	*Reconstructed* $p_{st} = \rho g H_{2D}$	NOT NECESSARY
			Dynamic Pressure P_{dyn}	*Calculated* *Poisson Equation*	NOT NECESSARY *(Neumann BC applied)*
			Velocity $\overrightarrow{U_{3D}}$	*Calculated* *Chorin Projection Method*	NOT NECESSARY *(Neumann BC applied)*
			Mean velocity $\overrightarrow{U_{MEAN}}$	*Integrated* $\overrightarrow{U_{MEAN}} = \int_{Z_B}^{H_{2D}} \overrightarrow{U_{3D}}\, dz$	DONE
		Exchange Interface Border			

Table 4.1: Calculation procedures within a 2D and 3D computational kernel. Exchange of the variables is performed for all parameters used on both sides of the coupling interface. All other parameters are set to have a zero-gradient condition.

this topic is published in ([8], [87], [130], [147]).

4.2.2 2D–3D Coupling: Shallow water models

In comparison to the complete bidirectional coupling between two different models of various complexities and dimensional representations, coupling between the 2D shallow water (see Sec. 3.4) and 3D shallow water models (see Sec. 3.5) can be done straightforwardly, due to the fact that the 3D-SWE is derived directly from the 2D-SWE. For the sake of clarity, the derivation scheme is illustrated graphically in Fig. 4.15 and explained step-by-step in the given implementation pipeline.

All the necessary variables in 2D and 3D space, which must be exchanged at the coupling interface, are present and calculated in every time sequence. It should be emphasised that the vertical component of the velocity vector $\mathbf{U}_{3D}$ and the dynamic component of the total pressure p_{dyn} are only used in the 3D model, thus no specific treatment of these variables is necessary after they have been calculated. A brief overview of all the parameters calculated and exchanged is given in Table 4.1, whereas the Neumann boundary condition has been applied to all variables which are exclusively calculated within one of the models and, as such, are not passed to the second model. Within the current work, the coupling interface and physical solid boundaries are not allowed to coincide.

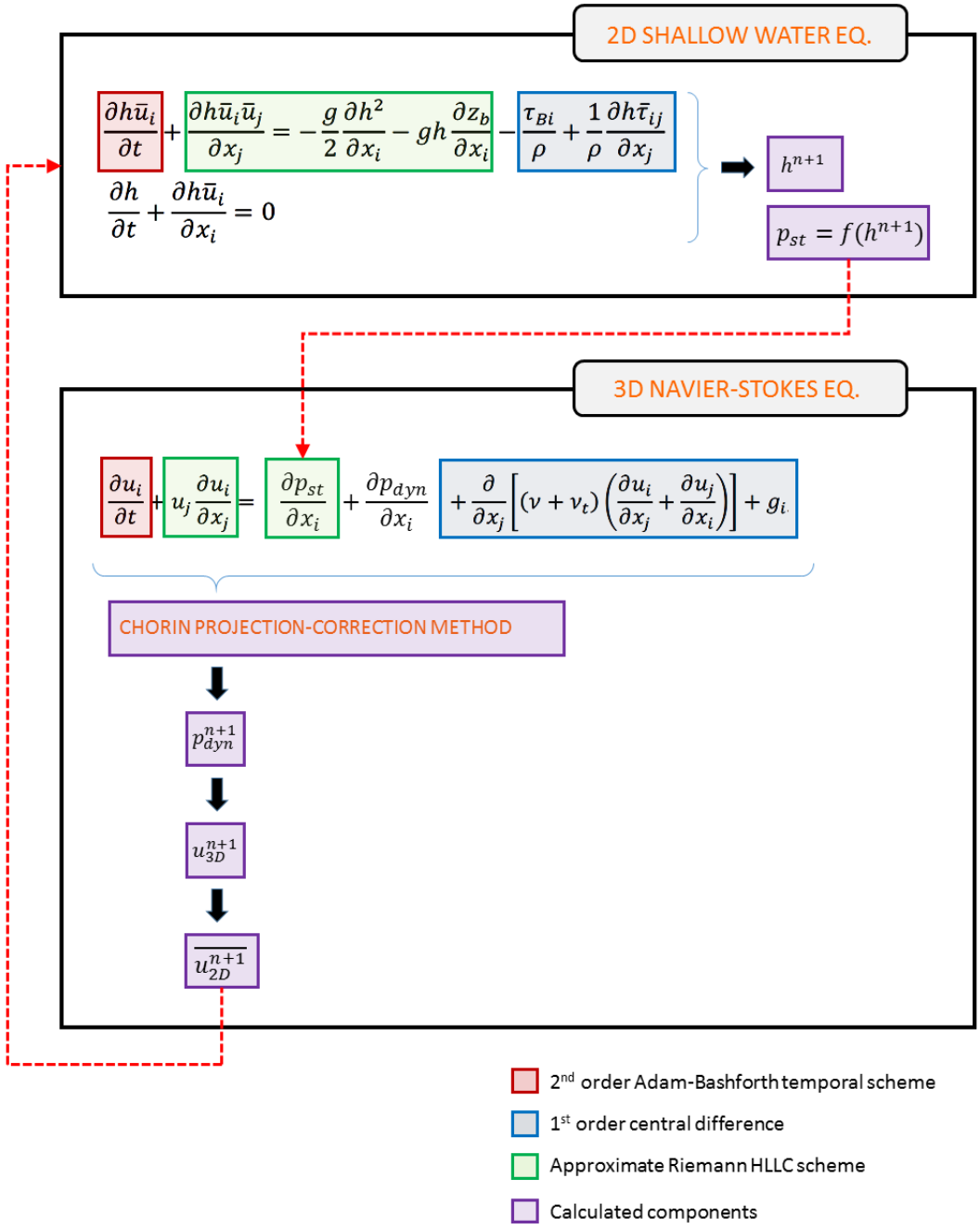


Figure 4.15: Detailed scheme of all intermediate steps within the coupling process between the two- and three-dimensional shallow water equations. The coupling is bidirectional in both horizontal directions, whereas the vertical velocity component and dynamic pressure value are exclusively calculated inside the 3D module.

3D-SWE Implementation pipeline:

1. Determine the time step Δt, small enough to avoid possible numerical instability,
2. Advance the total elevation η^n to η^{n+1} by solving Eq. 3.46,
3. Calculate the depth $h^{n+1} = \eta^{n+1} - Z_{terrain}$,
4. Calculate the hydrostatic component of the pressure $p_{st}^{n+1} = \rho^{n+1} g_z h^{n+1}$,
5. Solve the Navier–Stokes Eqs. 3.47–3.49 using the fractional step method:
   - Calculate the predicted velocity field $\mathbf{u}^*$ by neglecting the hydrodynamic pressure component p_{dyn}^{n+1} (p_{st}^{n+1} previously calculated is included),
   - Solve pressure Poisson equation for p_{dyn}^{n+1},
   - Correct the predicted velocity field, adding up the contribution of the dynamic pressure component p_{dyn}^{n+1},
6. Recalculate the time-step Δt based on the CFL constraints, listed in Sec. 4.1.7,
7. Repeat the whole procedure from step 2 to step 5, until the total simulation time is reached.

4.2.3 Coupling of the 2D Shallow Water and 3D Navier–Stokes models

The two-dimensional shallow water model, as described thoroughly in section 3.4, is a very simple and cost-effective model, when it comes to simulation of the fluid flow over large inundation areas, where the horizontal wave length is much larger than the vertical one. In that case, integration over the height can be performed without much loss of accuracy. This model, nevertheless, cannot perform well if a complex geometric description with concave, hollow regions is introduced. On the other hand, the three-dimensional Navier–Stokes model, explained in detail in Sec. 3.2, is a powerful, very expensive model in terms of memory and CPU resources, but is capable of resolving all kind of different geometric descriptions, working with multi-phase flows and the large gradients of the essential transport properties. Combining these two models in such a way that the 3D-NS model is used only in regions, where the 2D-SW model would otherwise introduce large errors and deviations, should result in an accurate, simple enough and cost-effective coupled model, capable of resolving a complex geometry and delivering fast, accurate solutions in the large flat areas. As shown in Fig. 4.16, the 2D-SW model can be used within the large blue-coloured flat area, whereas in the narrow red-colored region a complex geometry may be introduced and the 3D-NS model must be applied. The coupling of these two models is based on reconstruction of the missing 3D parameters inside the 2D kernel, non-available 2D parameters within the 3D model, and their subsequent exchange, before the solution for the next temporal instance is calculated. A general scheme of the execution procedure is illustrated in Fig. 4.17, suggesting an exchange of data only once after the 2D model has been run and then a second time in the direction from the 3D to

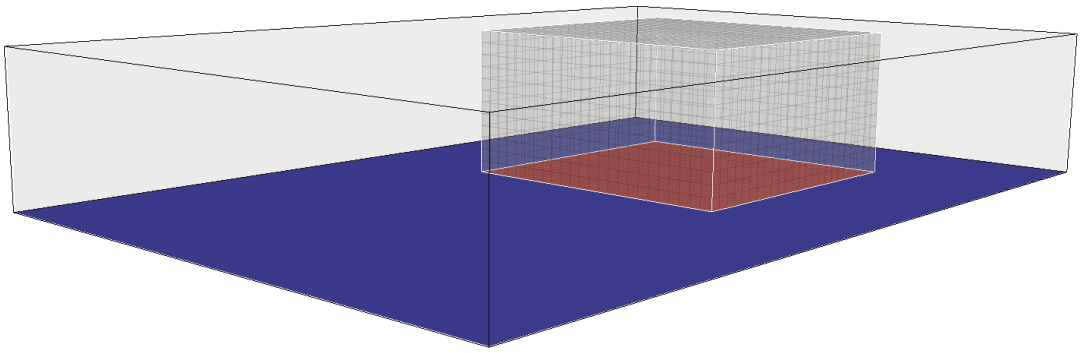


Figure 4.16: Example of the spatial coupling between the 2D–SWE model (outer, larger, blue-coloured area) and the 3D–NSE model (inner, smaller, red-coloured region). The complete domain (2D3D) is divided into grids as a part of the parallelisation strategy, and both 2D and 3D sub-domains contain a whole number of grids (no grid can be divided between 2D and 3D sub-domain). An additional layer of padded, ghost cells, created previously for the data exchange (see Sec. 5.1.2) is exploited here to also exchange all the missing parameters for both models.

the 2D model, just before the advance in time is done. This type of procedure can only be accurate if an infinitesimally small time step is assumed; thus an update of the variables within the separate models should not generate large discontinuities and gradients at the coupling interface. This is solely a numerical effect and it has nothing to do with the large gradients and discontinuities resolved within the 3D-NS model. The entire scheme consists of 12 steps, as depicted in Fig. 4.18, within which the following procedures are implemented:

- *Step 1*: **Solve 2D-SWE** results in the depth H_{2D}, elevation η_{2D} and averaged velocity field U^{2D}_{MEAN}
- *Step 2*: **Reconstruct p_{2D} & LS$_{2D}$** generates a hydrostatic pressure field and level-set representation based on the results obtained in *Step 1*

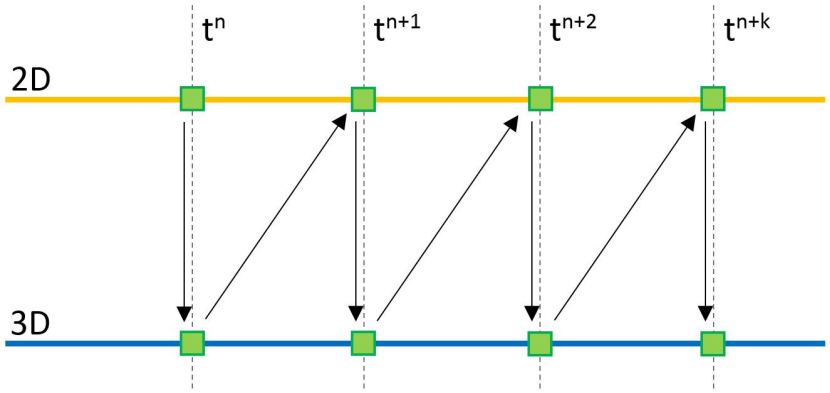


Figure 4.17: Order of execution of the coupled models throughout the simulation time. The steps of both the 2D and 3D models are illustrated in detail in Fig. 4.18.

$$depth = H_{2D} - Z_{cell}^{BOTTOM} \tag{4.117}$$

$$if(depth \geq 0) \Rightarrow \; LS_{2D} = \frac{1}{2}\left[\tanh\left(\frac{depth}{2\varepsilon}\right) + 1\right] \tag{4.118}$$

$$p_{2D} = \mathbf{g}\rho_f \cdot max(depth, 0) + \mathbf{g}\rho_a \cdot min(depth, 0), \tag{4.119}$$

- *Step 3*: **Diffuse sharp interface** yields a smeared interface representation, created using the reinitialization process, explained in section 3.6.3. In the 2D region, the interface normal vector cannot be calculated, thus a smearing of the interface is caused by an artificial diffusion. In comparison to the full reinitialisation process, characteristic for the 3D region, the number of iterations should be somewhat lower – up to three iterations, and no convergence criteria should be taken into consideration.

- *Step 4*: **Update $\rho_{\mathbf{2D}}$, $\mu_{\mathbf{2D}}$** recalculates density and viscosity parameters based on the newly reinitialised level-set value, calculated in *Step 3*. Besides that, a velocity block distribution (i.e. constant over the depth) is used as an interface boundary data towards the 3D model, for all the cells with the LS ≥ 0.001.

$$\rho = \rho_f \cdot LS_{2D} + \rho_a \cdot (1 - LS_{2D}) \tag{4.120}$$

$$\mu = \mu_f \cdot LS_{2D} + \mu_a \cdot (1 - LS_{2D}) \tag{4.121}$$

$$if(LS_{2D} \geq 0.001) \Rightarrow \; U_{3D} = U_{MEAN}^{2D}, \tag{4.122}$$

- *Step 5*: **Pass values 2D » 3D** performs the exchange of ghost layers, making the values previously calculated in the 2D model available for the 3D kernel.

- *Step 6*: **Solve 3D LS-Advection** solves Eq. 3.54 using second-order Total-Variation-Diminishing Runge–Kutta temporal scheme (see section 4.1.7.2) and second-order MUSCL slope-limiting spatial scheme with Sweby limiter (see section 4.1.6). As an alternative, both a first-order Euler explicit temporal scheme and a first-order upwind spatial scheme can be used.

- *Step 7*: **Compute normal vector $\mathbf{n_{3D}}$** computes $\mathbf{n}_\tau = \frac{\nabla LS}{||\nabla LS||}$, where the gradient of the level-set variable may be computed in many different ways. The least square gradient reconstruction, described in section 4.1.4.1, is used within this work.

- *Step 8*: **Reinitialize $\mathbf{LS_{3D}}$** is done entirely according to E. Olsson et al. [36]. This method is thoroughly tested against several other proposed methods, resulting in the most stable sharpening/diffusing process. See section 3.6.3 for more details.

- *Step 9*: **Update $\rho_{\mathbf{3D}}$, $\mu_{\mathbf{3D}}$** performs the same equations as in *Step 4*, with the difference that no velocity vector field U_{3D} is changed at this stage.

- *Step 10*: **Reconstruct $\mathbf{H_{3D}}$** integrates the reinitialised level-set scalar field, for all the computational points that belong to the fluid phase.

$$\text{if } (\text{LS}_{3D} \leq 0.5) \Rightarrow \; H_{3D} = \int \text{LS}_{3D} \cdot dz \tag{4.123}$$

- *Step 11*: **Solve NSE + Riemann** is based on the Chorin projection method, briefly explained in section 4.1.2, within which the advection term in the first projection phase is evaluated using Riemann HLL and HLLC solver (see. sections 4.1.5.1 or 4.1.5.2), while the diffusion term is separated into the normal diffusion, calculated using central difference scheme and cross diffusion, resolved using the LSM gradient reconstruction of tangential velocity components.
- *Step 12*: **Reconstruct $\mathbf{Q_{3D}}$ and $\mathbf{U^{2D}_{MEAN}}$** is done according to the following formulation:

$$Q_{3D} = \int \mathrm{LS}_{3D} U_{3D} \cdot dz \tag{4.124}$$

$$if(H_{3D} \geq 0) \Rightarrow \; U^{2D}_{MEAN} = \frac{Q_{3D}}{H_{3D}}, \tag{4.125}$$

At the end of one time instance $t^n \rightarrow t^{n+1}$, both the 2D and 3D kernels have been called only once, operating on their primitive variables and reconstructing the corresponding values in the opposite sub-domain (e.g. $U^{2D}_{MEAN} \Leftrightarrow U_{3D}$). This includes almost all the parameters of interest; nevertheless, some values from one domain, such as a complex geometric representation in 3D, cannot be easily reconstructed into the 2D terrain definition. For that reason, some parameters must be extrapolated directly from the sub-domain, to which they belong. All the extrapolation rules used within this work are listed in Table 4.2.

GRID type	Variable	Action
2D	Bed slope	Second-order linear extrapolation $\phi_{i+1} = 2 \cdot \phi_i - \phi_{i-1}$
3D	Minimum Distance Function	
2D & 3D	Gradient reconstruction	
3D	Velocity component U_x if $\mathrm{U}_x = 0$	
3D	Velocity component U_y if $\mathrm{U}_y = 0$	
3D	Velocity component U_z	
3D	Normal vector	
3D	Velocity component U_x if $\mathrm{U}_x \neq 0$	$\phi_{i+1} = \frac{1}{2}(\phi_{2D} + \phi_{3D})$
3D	Velocity component U_y if $\mathrm{U}_y \neq 0$	
2D	Mean velocity U_{MEAN}	

Table 4.2: Reconstruction and extrapolation of the missing variables within the 2D–3D coupled model.

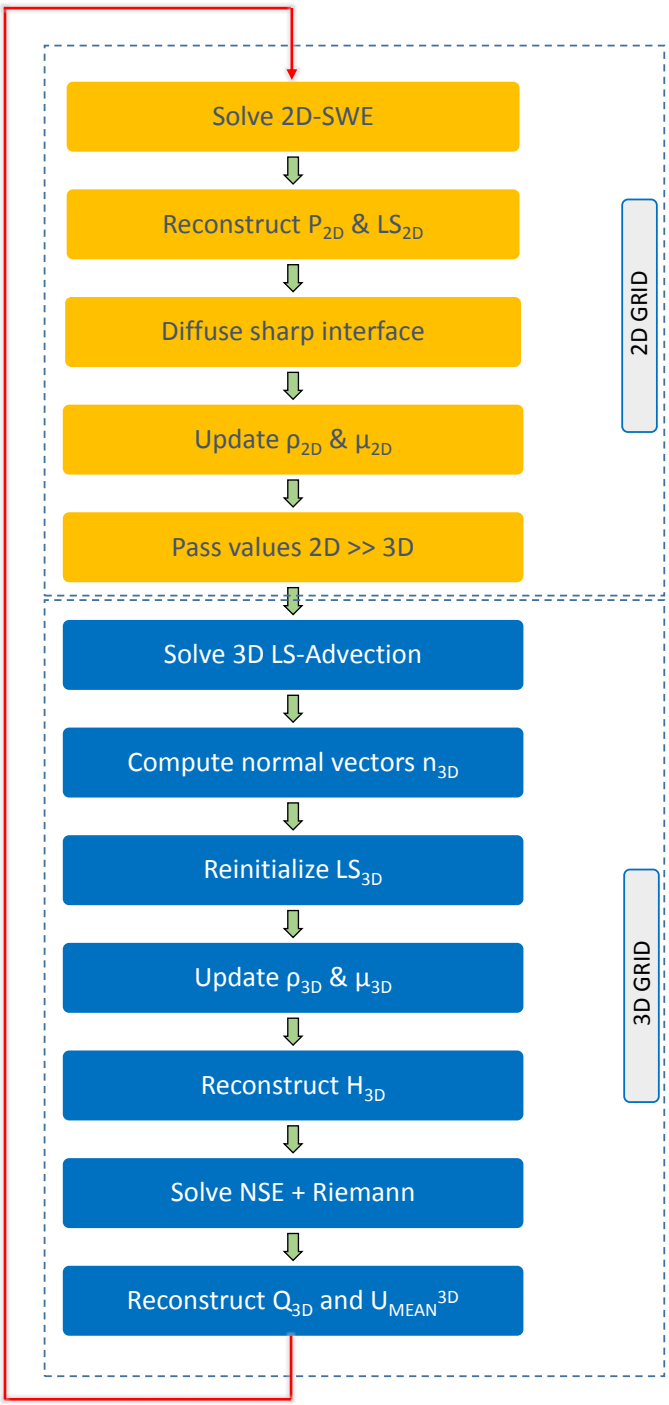


Figure 4.18: Detailed scheme of all intermediate steps at one time instance, within the coupling process between the two-dimensional shallow water equations and three-dimensional incompressible Navier–Stokes model. The coupling is bidirectional and it can be performed in both horizontal directions x and y, as can be seen in Fig. 4.16.

Chapter 5

High-Performance Framework: MPIFluid

5.1 Inherited state and further developments

5.1.1 Data structure

Due to the advantages discussed in Sec. 2.1, the mesh generation is performed using an adaptive space-tree approach, with a particular modification that allows uniform grid refinement even for non-cubic-shaped physical domains. In the simulation process of various engineering phenomena, the accuracy requirements often impose the size of cells and time advancement implicitly. For any spatially larger simulation domain, this leads to a vast number of computational cells, easily reaching up to several billions of computational volumes, which becomes cumbersome or even impossible to deal with sequentially. Domain decomposition and the employment of multiple cores and processors on a single machine or multiple nodes in a high-performance computing setup arise as natural solutions, in order to avoid both memory limitations and computing inefficiency.
The space-tree generator, as introduced in [100] and further adapted by Frisch [51] generates a volumetric model out of a triangulated, surface-based model within a single recursive procedure, once the physical boundaries of the volumetric domain have been set. To account for non-cubic domains, the initial division of the volumetric model is executed once, independently of the recursive procedure, resulting in regular, cubic sub-domains, which can subsequently be further divided, aiming for the required precision. This initial division is limited to rectangular domains and can be understood as a regularisation of the domain prior to subsequent refinement. As long as the initial volumetric domain can be described as the union of regular cubic sub-domains, the implementation of a non-cubic-shape domain should not cause any difficulties. Nevertheless, highly irregular surface models that can be found in aerodynamic analyses or one-direction-dominant models, for instance, cannot be simplified by applying this rule, thus a different strategy must be developed. The recursive procedure, as explained briefly in Sec. 2.1, consists of inexpensive intersection tests producing a narrow band of cells close to the triangulated surface that should be further refined. All the remaining cells are categorised as cells inside or outside of the geometry and are left coarse as much as possible. More precise intersection tests are performed only in the vicinity of the triangulated model, significantly increasing the efficiency of the refinement process, also referred to as the ‘voxelisation process’. The efficiency

analysis of this entire procedure is published in [100] and further tested in [119]. This basic idea is incorporated and adapted in [51], leading essentially to two stages:

- Generation of logical grids, also called L-grids (see Fig. 5.1), up to the required precision; and
- Division of the logical grids into data grids, also called D-grids (see Fig. 5.2), on which the actual computation is conducted.

The size of the D-grids $\{b_x, b_y, b_z\}$ can be chosen almost arbitrarily, with one important constraint: namely, the size of b_i for $i = \{x, y, z\}$, that must be divisible by the division rate on every refinement level r_i and the initial division rate s_i that accounts for the non-cubic shape of the domain. This constraint guarantees the concurrence of boundaries on every refinement level of the D-grids, and thus a straightforward linkage between the coarse and fine computational values, without the need for extensive interpolation and extrapolation procedures.
With a unique link between the L- and D-grids, the finite volume approach, in combination with the mid-point rule, degenerates into a finite difference approach, if no other requirements are imposed. The successful validation benchmarks are presented in [51], showing no loss in accuracy in the case of the interchangeable usage of both methods. Nevertheless, in the present work, despite the simplicity in implementing the finite difference scheme, the finite volume approach is adopted due to difficulties in satisfying both mass and momentum conservation criteria within the control volume (CV), particularly at the interface between two fluids of different physical properties.

5.1.2 Parallelisation

The generated data structure consists of non-overlapping, block-structured, orthogonal Cartesian grids which are embedded into the logical data structure as depicted in Fig. 5.2. According to Schwarz [131], the domain partitioning can be successfully done if every sub-domain generated is padded with a layer of additional ghost cells, which incorporate information from the neighbouring sub-domain as a boundary condition for the current subset. The convergence of such an approach while solving the Laplace equation $\Delta \mathbf{u} = 0$ is proven in the later work of [131] and is further extended by S. G. Mikhlin [122] for second-order elliptic partial differential equations. The necessary exchange and communication routines must be carried out according to the data parallelisation paradigm. Communication routines are completely separated from the computation procedures and have been carried out using the Message Passing Interface (MPI). This exchange and synchronisation process is deeply embedded into the generated data structure, allowing the grids to communicate in three different directions, e.g. with their coarse parent grid, the finer child grids, and neighbouring grids at the same level of the division, before any computation is conducted.
As already described in detail in [51], in the bottom–top synchronisation the data values, such as pressure, velocity, interface tracking quantity, etc. are aggregated, sent upwards and stored in the parent grid. This process is done entirely from the leaf nodes of the data structure until the root node is reached in a blocking manner, not allowing any interruption, before the last data is stored in the destination grid. Similarly, the top–bottom synchronisation is done in the opposite direction, with one difference being that only the ghost layer cells

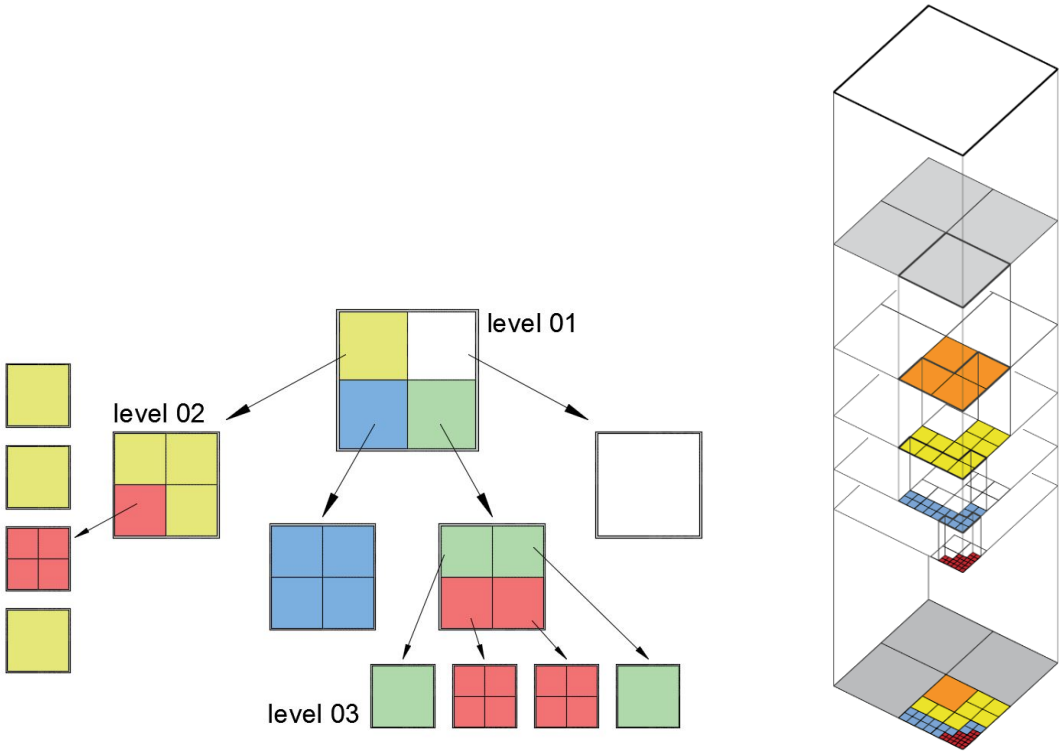


Figure 5.1: On the left-hand side, the logical data structure is shown in a tree-like fashion. Every L-grid has its own parent and child grids, except the coarsest, or root grid, depicted as the grid on level 01, and the finest, or leaf grids, which are not further refined, thus have only a direct coarse parent grid. The amount of refinement levels is influenced by the problem's solution and accuracy requirement. On the right-hand side, a complete adaptively refined data structure is depicted, where the refined topological grids on every level are marked with a different colour. The remaining grids on each level are unrefined and labelled as leaf grids.

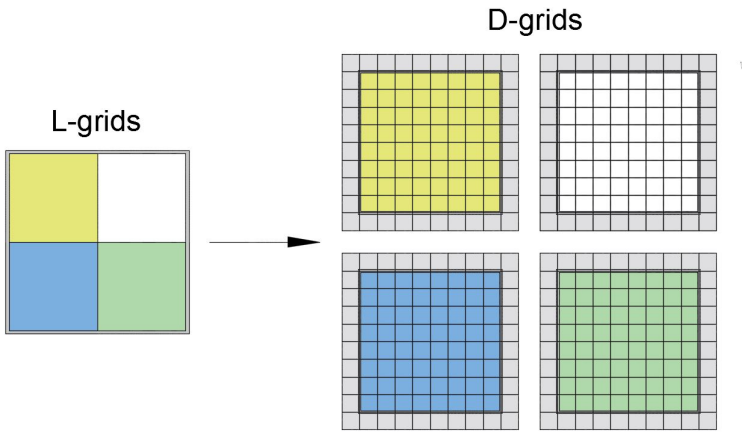


Figure 5.2: All logical grids independent of their location in the refinement tree are refined into $b_x \times b_y \times b_z$ data grids and surrounded with a single layer of ghost cells, required for the domain partitioning, according to the Schwarz method [131].

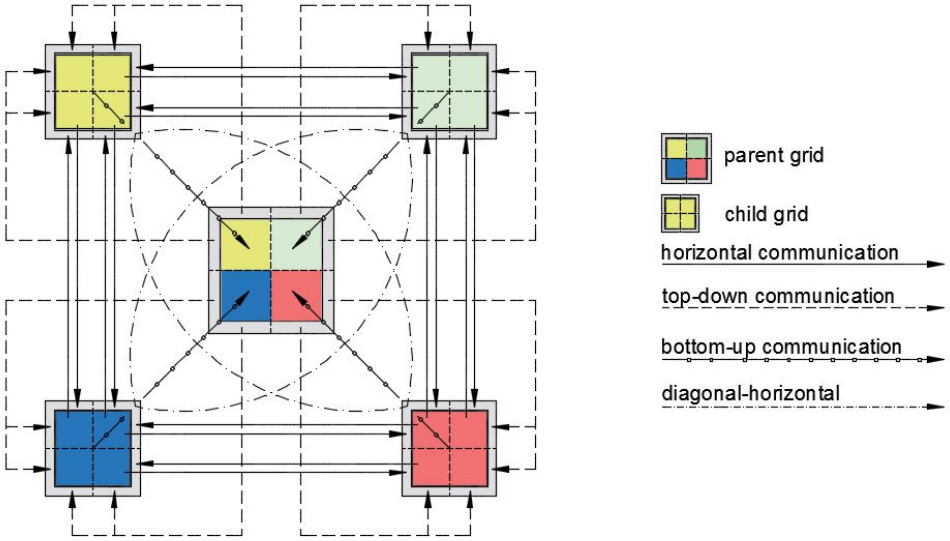


Figure 5.3: Three different grid synchronisation procedures depicted on a single level of division. The fourth diagonal–horizontal communication is an amendment to the existing horizontal type, which must be conducted, as the edge values exchanged are used in the least-square (LSM) scalar gradient reconstruction.

which are not included in the first synchronisation process are sent and received at this point. Finally, communication between each of the two neighbouring grids is done locally with no synchronisation collision being able to happen, thus this part can be executed fully in parallel.

This synchronisation process is important to maintaining the proper exchange of data between different grids in the data structure and it is completely independent of the distribution of grids to the multiple computational nodes. In other words, the synchronisation and exchange procedures cannot be omitted even if the simulation is done sequentially (number of processes $n = 2$). The exception to this rule is the case in which the complete simulation is done on a single logical grid linked to sufficiently divided data grids. In order to increase the efficiency of such exchange routines, only communication between adjacent grids is allowed. This is achieved using a grid management processor called a neighbourhood server, which holds the information about the grids' identification tags and their neighbouring grids, as well as their parents and children; thus before any synchronisation process a grid requires data about the other grids with which it is allowed to communicate, significantly reducing the amount of data sent and received, as would be the case if the communication was done globally in an all-to-all manner. Such an approach, despite the obvious advantages, should be used with care, as it might lead to bottleneck problems in the case of an extensive amount of inquiries being directed to the neighbourhood server within a single synchronisation session. One of the logical improvements concerning this problem is the employment of several neighbourhood servers, which would be responsible only for a particular part of the data structure and linked all together either in a global or a tree-like manner. More work done in this direction can be found in [84], while some measurements of the inquiry frequency in the case of the employment of one, two and four neighbourhood servers in the MPFluid framework itself are published in [51].

In order to setup a massively parallel simulation, apart from the already mentioned exchange and synchronisation procedures, the distribution of logical grids to multiple processors must be conducted, preserving the locality of the grids as much as possible, in order to reduce non-necessary communication between two physically distant parts of the domain. This problem is well-known and has been extensively studied in the field of mathematics and informatics since the end of the 19th century, when Giuseppe Peano discovered a geometric curve that passes through every point of a unit square divided into smaller parts, continuously mapping one-dimensional unit interval $[0, 1]$ onto a multidimensional one $[0, 1] \times [0, 1] \times \ldots \times [0, 1]$. This mapping method is named the Peano space-filling curve, after its discoverer, and since then many variations have been formulated, motivated by the original concept and constrained by various problem requirements. Among the most frequently used ones for general problems in engineering are the Hilbert, Morton and Sierpinski curves, while the Gosper curve, E-curve and H-tree curve have been developed for rather specific fields of application. Newly developed types, their comprehensive analyses, different application validations and recent advancement have been well documented by Bader [17]. These mappings established between one- and multi-dimensional spaces allow a simple manipulation of a multi-dimensional set and a generation of an one-dimensional array of data, which is then divided into equal data chunks and assigned to the available number of processes. In the MPFluid framework, the Morton curve, also known as the Z-curve, is employed, preserving a unique link between logical and data grids, and a significant number of validation tests concerning dynamical load balancing and ghost layer exchange efficiency are conducted in [51]. For the sake of completeness, one of the validation diagrams is shown in Fig. 5.4.

5.1.3 Inherited numerical treatment of Navier–Stokes equations

Having in mind that the MPFluid framework is initially developed to efficiently solve different fluid flow phenomena in the field of thermodynamics and hydrodynamics, it is no surprise that the underlying mathematical laws are formulated using Navier–Stokes momentum equations in their general form and one or more additional transport equations, which describe the particular phenomenon simulated. To date, there have been many different approaches to how to treat the whole system of equations together with different mass conservation formulations, some of which stand out in terms of accuracy, effectiveness or simply ease of numerical implementation. In this context, a rather standardised fractional step method, introduced by Chorin (see Sec. 4.1.2 for more detail), has been also utilised as a basis for the MPFluid framework. The well-known procedure combines the predicted, explicitly calculated velocity field and implicitly computed pressure field in order to satisfy the incompressibility constraint and to create a divergence-free velocity field for the additional energy-conserving equation. The implicit computation of the pressure field requires the solution of a system of linear equations and this can be resolved using either direct or iterative methods. It is proven to be a numerically sensitive and computationally expensive component of the entire simulation process. The standard procedure for the solution of the Poisson pressure equation in MPFluid involves direct Gauss elimination, the Jacobi and Gauss–Seidel iterative approaches and a multigrid-like solver that utilises the space-tree data structure already available in the code. Which of the solvers will be used depends largely on the problem to be solved and whether the considered flow is treated as incompressible or whether the compressibility effects should be accounted for. For the sake of completeness, in the next section a general overview of the

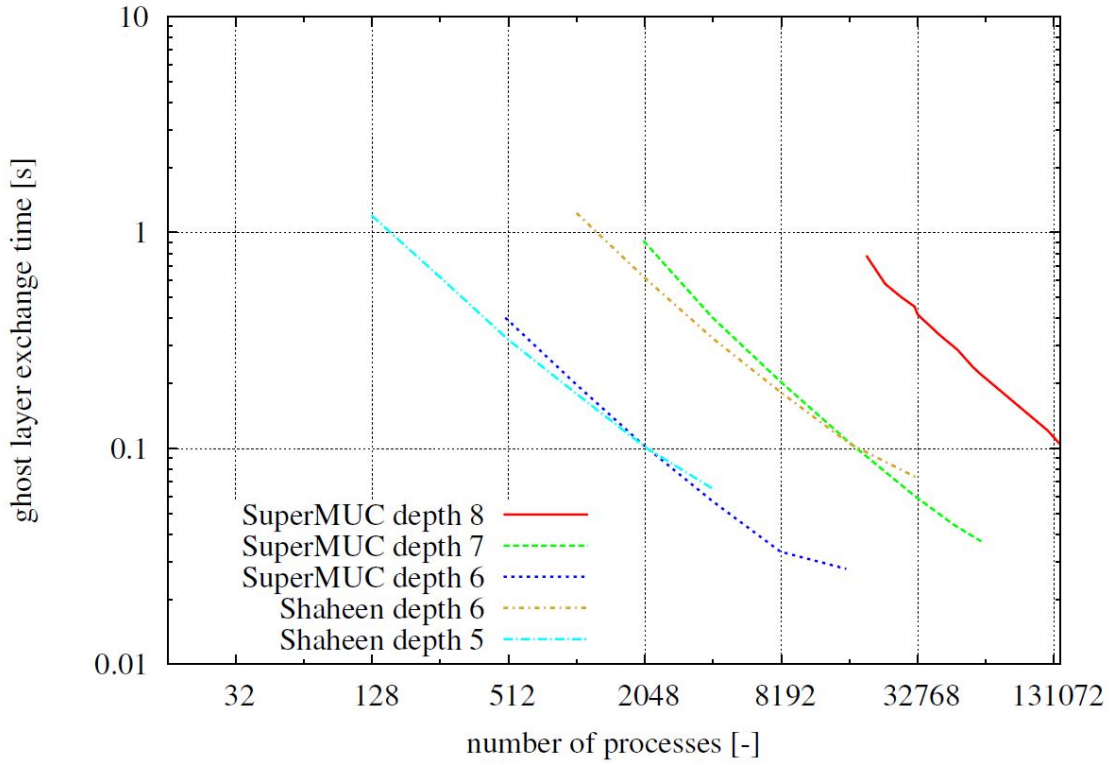


Figure 5.4: Ghost layer exchange efficiency, measured up to 8 levels of refinement, which is directly correlated to the increase of the distributed L-grids. Up to 131072 computing nodes are employed on two different high-performance computing architectures. This result is originally published in [51].

multigrid-like solver, which, together with the Jacobi solver, dominantly used throughout this work, is depicted. Nevertheless, for the implementation details and convergence tests for the solution of Poisson equations solely, the reader is referred to [51].

5.1.4 Multigrid-like solver

In order to understand why multigrid solvers are a natural choice for the solution of such setups including the space-tree data structure, a brief overview of the geometric multigrid method (GMG) will be given next, enclosing some similarities and the adaptations undertaken. For more details, the reader is referred to the standard literature, W. Hackbusch [148], P. Wesseling [114], V. V. Shaidurov [143], U. Trottenberg et al. [137], as well as to the lecture notes published in [33].

Employment of the Jacobi or Gauss–Seidel iterative approaches on a fine grid for the solution of a system of equations, which is built from the partial differential equations that describe generalised fluid flow, is commonly very expensive $\mathcal{O}(n^3)$, where n represents a number of computing points per one coordinate direction. It has been proven that these kinds of solvers reduce high-frequency errors very efficiently, while the smooth component remains, and for some classes of problems this can be diminished if iterated sufficiently long. Nevertheless, there are some examples in which this smooth error part at some stage becomes independent of

the number of iterations and leads to stagnation. Multigrid methods eliminate this remaining smooth component by introducing two or more grids, each one coarser than the previous one and all created from the original fine-grid representation. The basic principle of a multigrid is to combine the solution from two or more grids within one cycle instead of solving the system on the finest grid up to the required precision. In that way the high-frequency error is rapidly removed at each level and, due to the frequent exchange between finer and coarser grids, the smooth component of error is converted into a high-frequency form and is reduced systematically to the requirement set.

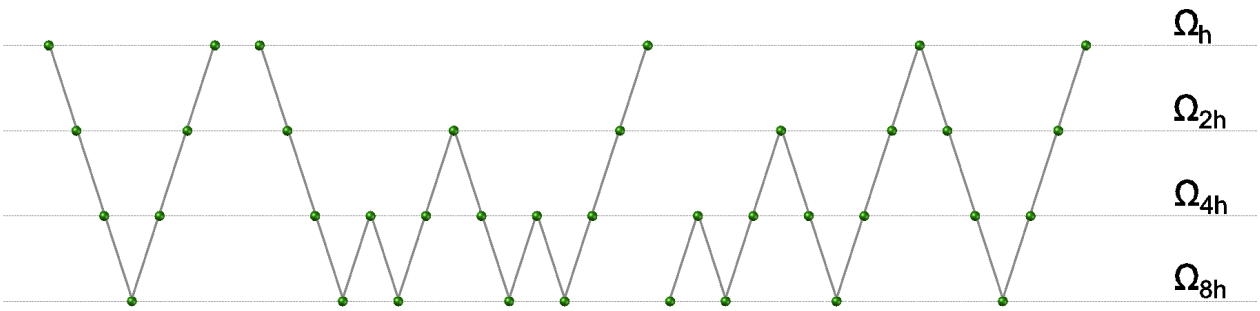


Figure 5.5: Three different variations of geometric multigrid: V-cycle (on the left), W-cycle (middle subfigure) and FMG-cycle (full multigrid cycle, on the right). A convergence rate conclusion (see [69])

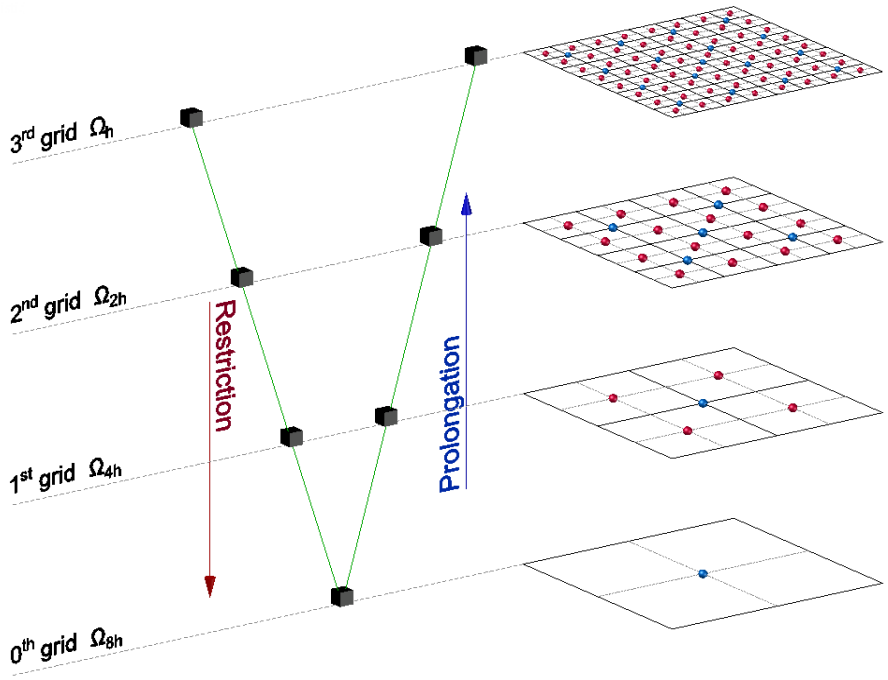


Figure 5.6: A typical 'V' multigrid cycle that operates on four grids of different refinement, always advancing from the finest (3rd grid) to the coarsest 0th grid. Within two characteristic stages, restriction and prolongation, a reduced system linear equations (only red points involved) is solved either partially (smoothing) or fully to the exact solution.

As shown in Fig. 5.6 the first step of the procedure starts at the finest level of refinement, in this case the third level, and consists of several iterations of any iterative solver, e.g. Jacobi or Gauss–Seidel applied on the system $A_h u = b$. As a general rule, within those several steps no exact solution of the system can be obtained, therefore this phase is commonly referred to as a smoothing procedure and it leads to an estimation of both the solution u_h and the residual $r_h = Au_h - b = A(u_h - u)$. In the second phase, this estimated residual r_h is transferred to the coarser grid (the second level of refinement, in Fig. 5.6), using a restriction operator R_h^{2h}, following the mathematical formulation $r_{2h} = R_h^{2h} r_h$. Once the estimation of the residual on the coarser grid is available, the system of equations $Ae_{2h} = r_{2h}$ is solved. If no further coarser grids are available the computed error e_{2h} is transferred back to the finer grid using a prolongation or interpolation operator I_{2h}^h, such that $e_h = I_{2h}^h e_{2h}$. With an estimation of the error on the finest grid, the approximated solution u_h can be corrected. This process is repeated as many times as necessary until the required accuracy of the solution is reached. In the case of multiple grids, the V-cycle is extended naturally by the additional definition of restriction and prolongation operators for each of the two following grids. Note that, until now no particular definition of those two operators has been given. Different variations of the multigrid are based on a different formulation of the operators R and I, and various solutions can be found in theory. The actual choice of those two operators depends on the data structure used, the problem to be solved and moreover the mathematical properties, allowing the initial problem to stay symmetric and positive-definite. Nevertheless, the pure injection and the full weighting operators as defined in [96] are commonly used formulations that satisfy the above mentioned requirements and achieve the maximum efficiency from the process. For that reason, those two operators are also used in this work, and more details on different multigrid practice in the field of computational fluid dynamics can be found in P. Wesseling and C. W. Oosterlee [115].

Having in mind that the systems solved on coarser grids contain fewer elements, and thus require less time to solve, it is advisable to combine as many coarse grid swaps as possible and to return back to the fine-grid representation only from time to time for the necessary correction. This is the reason why, besides the common V-cycle, the more powerful W-cycle (see Fig. 5.5, middle plot) is also frequently used. The third, even more advanced solution, illustrated in Fig. 5.5, right-hand plot, is a full multigrid cycle (FMG) that achieved a rather promising increase in the convergence speed within our MPFluid framework. The basic idea of the FMG approach is to create a more accurate initial guess, onto which a classical V-cycle can be applied, as previously depicted. This improvement is achieved by starting at the coarsest grid and then by using restriction and prolongation routines to advance one level-of-depth at the time, until the deepest level of refinement is reached. The smoothing procedures in this preparation phase are done level-wise. Detailed analysis is published by Jungblut [69] in his Master's thesis. Within the multi-phase fluid-flow kernel, both the V-cycle and FMG multigrid are regularly used and the promising increase in convergence can only be confirmed.

Having defined the data structure as depicted in Fig. 5.1 and the basic multigrid principle shown in Fig. 5.6, it is obvious that both procedures contain a tree-based structure. Closely observed, the typical multigrid V-cycle starts at the finest level, iterates over coarse grids and returns back to the fine-grid representation. On the other hand, the MPFluid data structure is constructed from the coarse-grid representation downwards and the vertical linkage between the grids is established as bidirectional. If the V-cycle is turned upside down, the restriction

operator would coincide with the aggregation operation in the bottom–up synchronisation process, while the prolongation operation would match up with the extrapolation procedure within the top–down communication (see Fig. 5.3). This means that the inherited data structure synchronisations can be used to define the restriction (R) and interpolation (I) operators, and the smoothing procedures that occur in every multigrid swap can be conducted on already defined grids of different levels of refinement. This comes with no additional costs for the implementation of the multigrid-like cycle into the data structure and the flexibility to choose different iterative solvers, in that way influencing the speed of reduction of high-frequency errors.

5.2 Poisson equation – constant coefficient formulation

Using the well-established and, for a variety of simulation setups, numerically stable Chorin fractional step method, the Navier–Stokes momentum equations are successfully linked to an equation that guarantees conservation of mass. This linkage is possible due to two additional constraints, namely incompressibility of the fluid and a rotation-free velocity field. These two conditions are mathematically expressed as:

$$\frac{\partial \rho}{\partial t} = 0 \tag{5.1}$$

$$\nabla \cdot \mathbf{u}^{n+1} = 0, \tag{5.2}$$

respectively, which results in the following transformation of the correction step of the momentum equations:

$$\frac{(\rho \mathbf{u})^{n+1} - \rho^{n+1}\mathbf{u}^*}{\Delta t} = -\frac{\partial p^{n+1}}{\partial \xi_i} \tag{5.3}$$

$$\frac{\nabla \cdot \mathbf{u}^{n+1} - \nabla \cdot \mathbf{u}^*}{\Delta t} = -\nabla \cdot \left(\frac{1}{\rho^{n+1}} \frac{\partial p^{n+1}}{\partial \xi_i}\right) \tag{5.4}$$

Substituting Eq. 5.2 into Eq. 5.4 and reordering the constant, known values under the arbitrary parameter Q_{RHS}, the following form is obtained:

$$\nabla \cdot \left(\frac{1}{\rho^{n+1}} \frac{\partial p^{n+1}}{\partial \xi_i}\right) = Q_{RHS} \tag{5.5}$$

where n+1 stands for the succeeding simulation time step, * is utilised to label the predicted velocity vector field, $\mathbf{u} = \{u, v, w\}$ with its orthogonal Cartesian components, ρ is fluid density, p represents the pressure field, and $\xi_i = \{x, y, z\}$ is the corresponding Cartesian spatial step annotation.

This general formulation is suitable for both compressible and incompressible fluids, including simulation of single-phase and multi-phase flows (e.g. simulation of two or more inviscid,

non-reacting fluids). However, it must be clearly stated that the constraint, given in Eq. 5.1, is no longer valid for compressible fluids, thus the term $\nabla \cdot \mathbf{u}^{n+1}$ in Eq. 5.4 is not negligible and must be approximated using the mass conservation law $\nabla \cdot \mathbf{u}^{n+1} = -\frac{\partial \rho}{\partial t}$.
Extensive research has been done concerning this issue and it has been reported by J. B. Bell and D. L. Marcus [60], Y. Morinishi et al. [154] and F. Nicoud [48] that this term often introduces numerical instabilities. Whether this will happen in a particular simulation case depends on the density rate in a single computational cell, as well as the density ratio between two adjacent grid cells. Although no general closed form suitable for all simulation scenarios has been developed, a rather robust solution is offered in [48] and [98], with no significant rise in computational costs, yet the memory requirements are slightly increased due to the second-order temporal approximation scheme applied (see Eq. 5.6).

$$\left.\frac{\partial \rho}{\partial t}\right|^{n+1} = \frac{\left[(\Delta t_n + \Delta t_{n-1})^2 - \Delta t_n^2\right] \rho^{n+1} - \left[(\Delta t_n + \Delta t_{n-1})^2\right] \rho^n + \Delta t_n^2 \rho^{n-1}}{\Delta t_n \Delta t_{n-1} (\Delta t_n + \Delta t_{n-1})} \tag{5.6}$$

After suitable approximation of the above-mentioned term is done, in order to avoid a loss of mass inside the control volume (under the premise that an additional body source term contribution is not present), the general form of Poisson equation is obtained as shown in Eq. 5.5. In case of a single-phase incompressible flow, the variation of density in all three spatial dimensions is negligible, thus Eq. 5.5 can be simplified by moving the density value out of the divergence operator and to the right-hand side:

$$\nabla \cdot \left(\frac{\partial p^{n+1}}{\partial \xi_i}\right) = \rho^{n+1} Q_{RHS} \tag{5.7}$$

$$\Delta p^{n+1} = Q^c_{RHS} \tag{5.8}$$

For the sake of simplifying the depiction of all the important parameters, Eq. 5.8 will be derived in two-dimensional orthogonal Cartesian space (see Fig. 5.7), whereas its extension into three-dimensional space is rather straightforward. After integration over the control volume and application of Gauss' divergence theorem, a discretised form is obtained and shown in Eqs. 5.10 and 5.11.

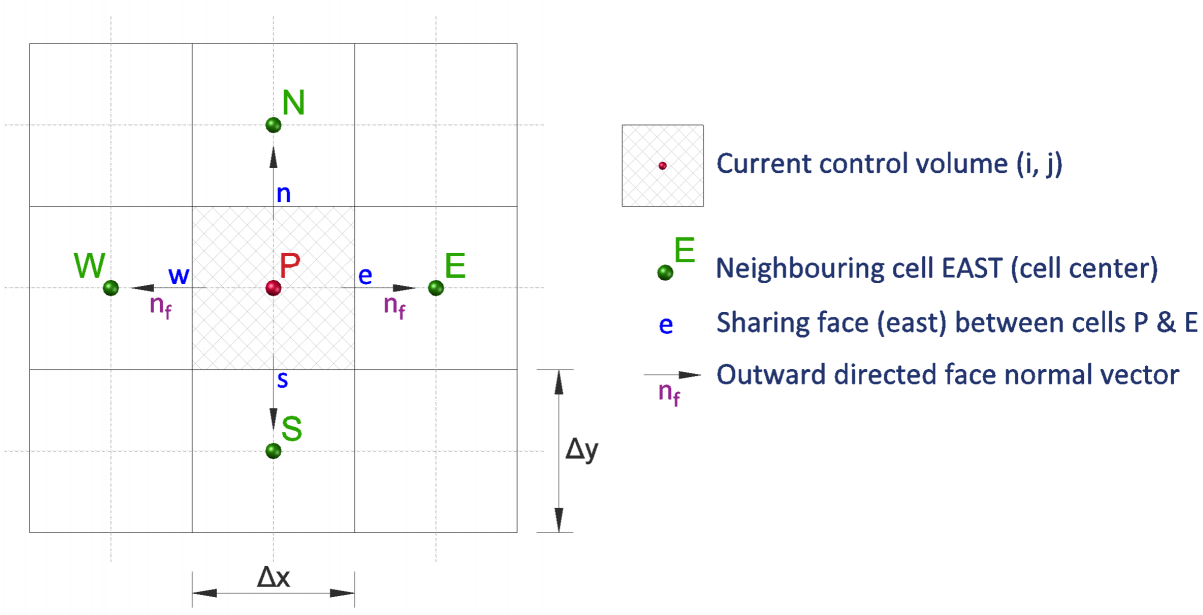


Figure 5.7: Collocated grid arrangement with computed values saved in the cell centres P, E, W, N and S is used for the discretisation of the pressure Poisson equation, after the integration of Eq. 5.5 over the P cell control volume (CV) marked.

$$\Delta p^{n+1} = \sum_{f \in F(c)} \left(\frac{\partial p^{n+1}}{\partial \xi_i} \right)_f \cdot \mathbf{n}_f = \tag{5.9}$$

$$= \frac{1}{\Delta x^2}(p_E^{n+1} - p_P^{n+1}) + \frac{1}{\Delta x^2}(p_W^{n+1} - p_P^{n+1})$$

$$+ \frac{1}{\Delta y^2}(p_N^{n+1} - p_P^{n+1}) + \frac{1}{\Delta y^2}(p_S^{n+1} - p_P^{n+1})$$

$$= Q^c_{RHS} \tag{5.10}$$

$$Q^c_{RHS} = \underbrace{\frac{1}{\Delta x_e^2}}_{a_1} p_E^{n+1} + \underbrace{\frac{1}{\Delta x_w^2}}_{a_2} p_W^{n+1} + \underbrace{\frac{1}{\Delta y_n^2}}_{a_3} p_N^{n+1} + \underbrace{\frac{1}{\Delta y_s^2}}_{a_4} p_S^{n+1} - \underbrace{\left[\sum_{i=1}^{2d} a_i \right]}_{a_P} p_P^{n+1} \tag{5.11}$$

Closely observing Eq. 5.11, it is obvious that the coefficients a_i for $i \in \{1, 2, \ldots 6, P\}$ provide geometrical information of the grid on which the discretisation has been performed. This physical interpretation is only valid in the case of single-phase incompressible flow and, if further simplified by the introduction of an equidistant mesh in all three dimensions $\Delta h = \Delta x = \Delta y = \Delta z$, the parameters a_i become constant and equal to each other. For this reason, although not always justified, the formulation depicted in Eqs. 5.8–5.11 is often called the 'constant-coefficient Poisson system of linear equations'.

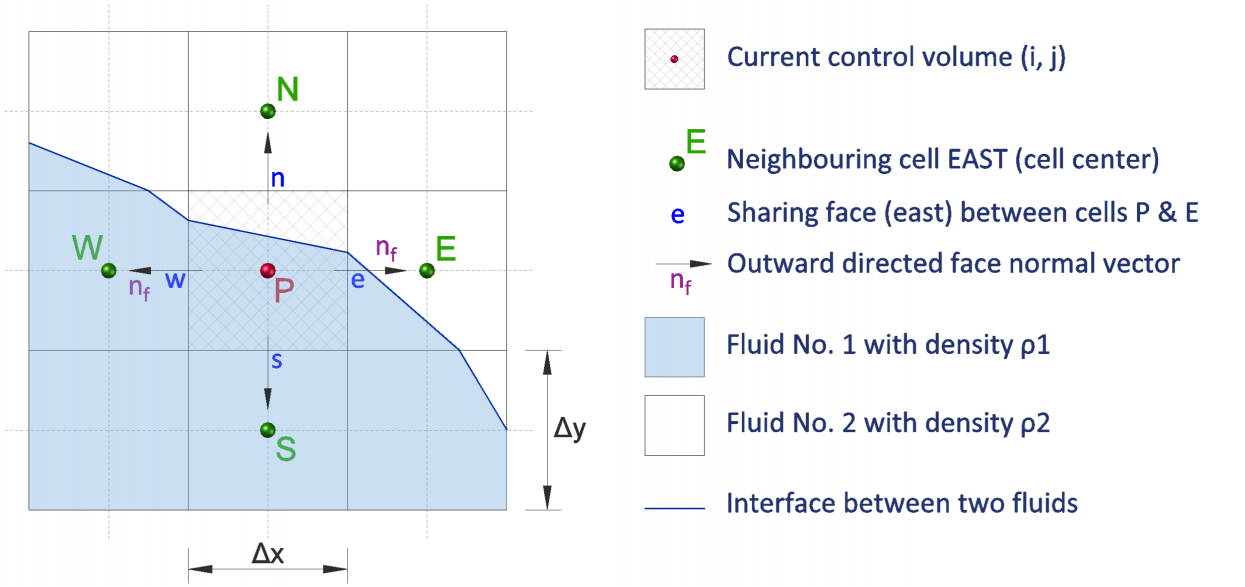


Figure 5.8: The two-phase flow setting does not influence already adopted collocated grid arrangement. In comparison to Fig. 5.7 the discretisation of pressure Poisson equation becomes more complicated due to the fluids' additional properties introduced.

Having the general formulation of pressure Poisson equation for incompressible single-phase flow, its extension to the more complex form for multi-phase flow should not be difficult to comprehend. The differences that arise and the possible consequences will be discussed in the next section.

5.3 Poisson equation – variable coefficient formulation

As stated above, both the constant- and variable-coefficient Poisson equations are derived from the general form given in Eq. 5.5. Comparing Figs. 5.7 and 5.8, it can be seen that an additional piece of information about the two or more fluids is introduced. In order to keep this depiction concise, it is assumed that the first fluid with density ρ_1 is the heavier one, e.g. water (in the further text referred to as the 'fluid phase'), while the second fluid with density value ρ_2 is usually called the lighter phase, in the further text also named the 'gas phase'. In accordance with this declaration, an annotation with the subscript 'w' (for example, ρ_w) will be reserved for the heavier fluid, while the subscript 'g' (e.g. ρ_g) shall refer to the gaseous or air phase.
Common scientific practice in simulating two or more different fluids in conjunction is to track or reconstruct an interface between those fluids and to define the transport properties in the transition cells as a function of the interface identified (see Eq. 5.12).

$$\rho = (1 - \phi)\rho_w + \phi\rho_g \tag{5.12}$$

where ϕ represents a reconstructed or tracked interface. An overview of some frequently used interface recognition methods is given in Sec. 3.6, while a detailed description of the

conservative level-set method used throughout this work is given in Sec. 3.6.3. It is important to understand that this kind of numerical tactic leads to the generation of three types of cells: namely, pure water cells with $\phi \to 0$ and $\rho = \rho_w$, pure air cells with $\phi \to 1$ and $\rho = \rho_g$, and a narrow band of transition cells, where $0 < \phi < 1$ and $\rho_g < \rho < \rho_w$. These cells will be referred to as 'surface cells', and the difficulties in solving the pressure Poisson equation are located exactly in this region.
Observing Eq. 5.5 more precisely, it is obvious that the density value under the divergence operator is no longer constant, due to the different fluid types simulated, therefore the simplification applied in the case of the constant-coefficient pressure Poisson equation (shown in Eqs. 5.7 and 5.8) cannot be applied here. The complete form derived, starting from Eq. 5.5 and taking into account the set fluid properties, is shown below:

$$\nabla \cdot \left(\frac{1}{\rho^{n+1}} \frac{\partial p^{n+1}}{\partial \xi_i}\right) = \sum_{f \in F(c)} \left(\frac{1}{\rho} \frac{\partial p^{n+1}}{\partial \xi_i}\right)_f \cdot \mathbf{n}_f = Q_{RHS} \tag{5.13}$$

$$\begin{aligned} Q_{RHS} = {} & \frac{1}{\rho_e^{n+1} \Delta x^2}(p_E^{n+1} - p_P^{n+1}) + \frac{1}{\rho_w^{n+1} \Delta x^2}(p_W^{n+1} - p_P^{n+1}) \\ & + \frac{1}{\rho_n^{n+1} \Delta y^2}(p_N^{n+1} - p_P^{n+1}) + \frac{1}{\rho_s^{n+1} \Delta y^2}(p_S^{n+1} - p_P^{n+1}) \end{aligned} \tag{5.14}$$

$$Q_{RHS} = \underbrace{\frac{1}{\rho_e^{n+1} \Delta x_e^2}}_{a_1} p_E^{n+1} + \underbrace{\frac{1}{\rho_w^{n+1} \Delta x_w^2}}_{a_2} p_W^{n+1} + \underbrace{\frac{1}{\rho_n^{n+1} \Delta y_n^2}}_{a_3} p_N^{n+1} + \underbrace{\frac{1}{\rho_s^{n+1} \Delta y_s^2}}_{a_4} p_S^{n+1} - \underbrace{\left[\sum_{i=1}^{2d} a_i\right]}_{a_P} p_P^{n+1} \tag{5.15}$$

where p_Q, for $Q = \{P, E, W, N \text{ and } S\}$, as shown in Fig. 5.8, represents the cell-centred pressure value in the corresponding control volumes, $\Delta\xi_f^2$ for $\xi = x, y, z$ is the normal projected distance between two adjacent cell centres, and $f \in F(c)$ is a matching control volume face. The fluid density ρ_f is evaluated at each face of the CV, and according to work of F. N. Felten and T. S. Lund [47] and L. Jofre et al. [80] the simple interpolation $\rho_f = 0.5(\rho_P + \rho_{NB})$ results in the minimisation of the kinetic energy loss, thus better conservation properties.

Eq. 5.15 reveals that coefficients a_i, in contrast to their counterparts in Eq. 5.11, do not have a purely geometrical meaning and also contain information about the transport properties of the fluids simulated in a particular case. The coefficients a_i are grouped together and represented as a part of the matrix K of the system of linear equations $Kp^{n+1} = Q_{RHS}$.

Dealing with the 'constant-coefficient' pressure Poisson equation, both a highly skewed, non-orthogonal mesh representation and the necessity of discretising the numerical domain in one spatial direction more often than in the two other directions, may lead to the generation of 'ill-conditioned' matrix. This behaviour is even more pronounced in the case of the 'variable coefficient' pressure Poisson equation where, besides the two mentioned reasons, additional problems are introduced by a very high density ratio, typically 1:1000, between the two adjacent control volumes.

These sorts of systems are solved either with direct or iterative solvers, and depending on the structure of the matrix K, some of them might be more suitable than others. The majority of the commonly used direct numerical solvers have complexity $\mathcal{O}(n^3)$, with n being a number of computing points per coordinate direction, which becomes very inefficient for any larger problem. On the other hand, iterative methods require an initial guess and in every subsequent iteration a new approximation to the exact solution is calculated. Moreover, an integral part of every iterative solver is a termination criterion, with which the quality of an obtained approximation is controlled. Both the termination criterion and choice of initial guess can heavily influence the final result, although a good practice is developed which minimises the negative impact in case of many such solvers for the specific groups of problems. As a general rule, iterative solvers are more efficient, but some formal requirements that each of them imposes on the matrix K should not be neglected. Strictly speaking, the steepest descent calls for the symmetric positive-definite matrix K, otherwise no quadratic minimisation problem can be defined. In case of violation of the positive-definite conditions, a saddle point will be obtained and, if the symmetry is not fulfilled, the gradient no longer corresponds to our initial system

$$\nabla f \neq K p^{n+1} - Q_{RHS}$$

but to

$$\nabla f = 0.5(K + K^T) p^{n+1} - Q_{RHS},$$

thus a different problem is solved. The Jacobi (JS) and Gauss–Seidel (GS) iterative solvers require a diagonally dominant matrix K with non-zero values on the main diagonal, in order to guarantee convergence and, although GS is somewhat more robust than JS, successive updates of the calculated values and their immediate usage in the calculation of the remaining matrix elements makes it extremely difficult for parallelisation. The Conjugate Gradient (CG) method requires a real, symmetric, positive-definite matrix and it is applicable to the sparse matrix representation. An additional generalisation of the CG for non-symmetric and non-linear systems of equations has been developed as well.

Throughout this work, the Jacobi (JS) and the multigrid-like solver (see Sec. 5.1.4) have been used. The problems arising from the high density ratio have been addressed using a simple diagonal preconditioning technique, although this does not solve the problem entirely, as the high density ratio also influences several off-diagonal elements. For this reason, the number of iterations necessary to reach the required accuracy is somewhat larger than in the case of solution of the constant-coefficient pressure Poisson equation. Another approach to this serious issue is the so-called numerical reduction from two-phase to single-phase flow, putting the emphasis on the simulation of one more important phase. In that case, the elements of the matrix K which correspond to the second, light gaseous phase are attenuated, significantly reducing the number of iterations to the 'exact solution'. The latter approach and its efficiency analysis are published in [39] and [40].

Chapter 6

Application setups

6.1 Flows through porous media

Within this chapter the coupling between the Darcy Law that describes flows through porous media on the macro scale and Navier–Stokes model that depicts flow phenomena on the micro scale, has been done. For the sake of clarity, it is important to state that this coupling does not allow the quantities of interest to be exchanged bi-directionally; instead, the simulations on the micro scale are performed, delivering the results that can be parametrized and used within the macro scale model.

The focus of the study has been placed on realistic pore velocity (i.e. between two grains of sand) on the micro-scale and virtual Darcy flux on the macro-scale; thus the cell refinement of both the fluid and solid regions must be conducted with care. A general recommendation in the broad scientific literature is to generate between 10 and 20 computing points per size of characteristic diameter (e.g. D_{60} or D_{90}, depending of the type of the study). A characteristic diameter shows the percentage of smaller fractions present in the soil sample, thus $D_{60} = 120$ mm reveals that 60% of total sand grains has diameter smaller than 120 mm. Even if properly conducted, this recommendation can lead to an insufficient number of computing points between two adjacent sand grains, resulting in an ill-conditioned problem. Further refinement may reduce this constraint; nonetheless, for the phenomena investigated in this case study ($\mathrm{D}_{10} = 60mm$), a minimal spatial step $\Delta = 100/128 = 7.8125$ mm is adopted, resulting in less than 2% of grains being insufficiently refined. Further finer scales are not taken into consideration.

In the mesh generation process, two different refinement strategies are used – uniform and adaptive ones – yielding a significant difference in the number of computing cells in both the fluid and solid regions. In order to depict this variability, 10 different porous samples are generated with porosity values of 50%, 55%, 60%, 65%, 70%, 75%, 80%, 85%, 90% and 95%. The samples generated are of the same size – 1 m × 1 m × 1 m – with the same volumetric ratio as depicted in Tab. 2.1. For all 10 cases an adaptive and uniform mesh refinement is performed, leading to a total reduction of 80% of the computing cells (i.e. from five to one million cells) and a mesh generation process up to four times as fast, in favour of the adaptive setting. A detailed comparison of all 10 setups studied, in both the fluid and solid regions, can be seen in Fig. 6.1. Further explanation has been published by N. Perović et al. [105].

As documented in detail in Sec. 4.2.1, the purpose of the micro–macro-scale coupling is the

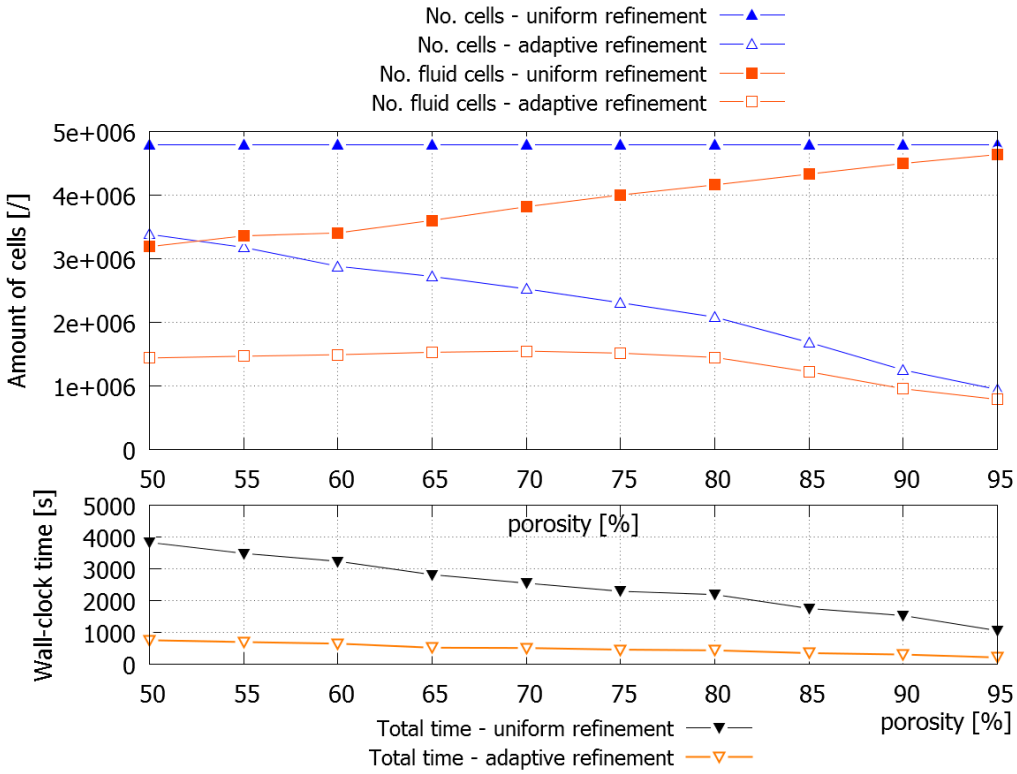
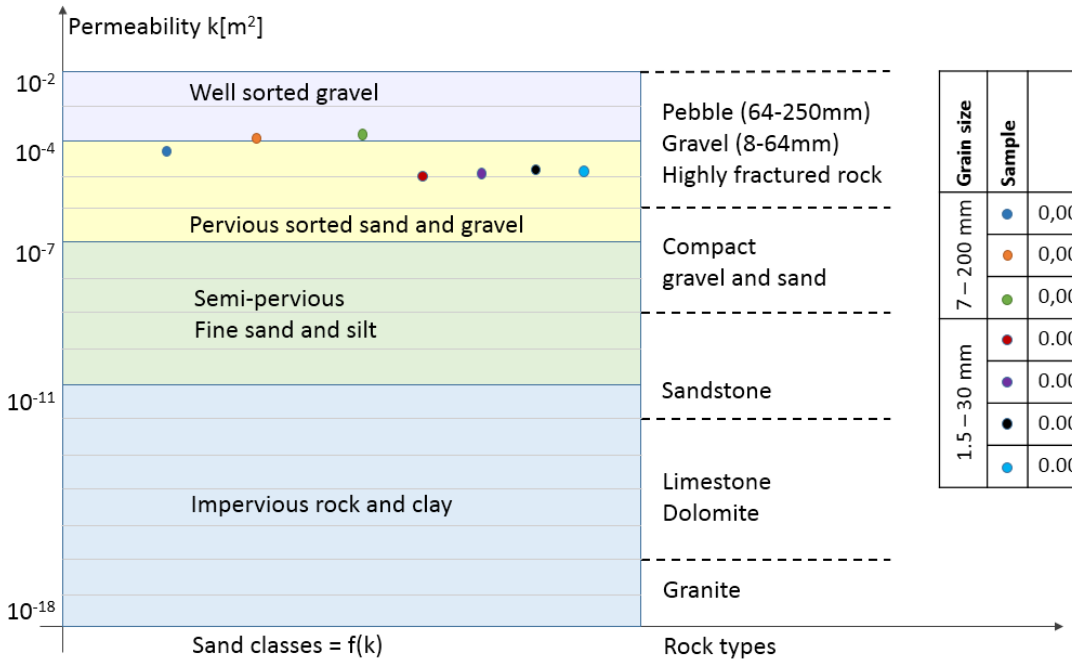


Figure 6.1: Grid study based on the uniform and adaptive refinement strategy. For ten different cases with predefined porosity values (depicted on the x-axis) both the number of total and solely fluid cells in the generated numerical domain are shown (upper figure). Lower figure depicts the simulation run-time of all ten cases, where much shorter times for the adaptive refinement in comparison to the uniform refinement can be observed.

possibility of using several parameters calculated at the micro-scale and generating macro-scale data (i.e. permeability and hydraulic conductivity), which could otherwise be obtained only in a strictly controlled experimental environment. The latter is much more expensive and time-consuming. For this reason, it is extremely important to minimise the effort and costs during the experimental phase and to use such results in the validation process of a numerical approach, presented here. One of the validation sets used is published by Bear [18], showing the correlation between permeability and the soil type. Due to the nature of the soil sample generation process (see Sec. 2.2.1), the vast majority of the samples created are of non-cohesive, sandy material of a different random sand grain size; thus the validation process must be done for highly fractured rock, pebble and gravel material. With the real-life porosity data published in [18], seven different non-cohesive soil samples are modelled, and using the strategy with several scales, as explained in Sec. 4.2.1, their permeability values are calculated. A comparative analysis of the results calculated and the measurements obtained in the laboratory environment is illustrated in Fig. 6.2. It can been seen that all seven randomly generated samples fit into the soil category of gravel, pebble and highly fractured rock. The differences between the samples originate from the fact that the porosity value is set to 45% ±3%, combined with the random insertion of sand particles within each sample. Once the data sets are validated, further sensitivity analysis has to be performed in order to establish the influence of an introduced



Grain size	Sample	k [m²]	Porosity [%]
7 – 200 mm		0,00007643	55
		0,00011358	60
		0,00016050	65
1.5 – 30 mm		0.000016312	55
		0.000022228	55
		0.000026955	60
		0.000024331	60

Figure 6.2: Comparative analysis of the numerically obtained permeability results (depicted as coloured dots in the graph) to the experimentally measured values of permeability, taken from [18], and sorted into different ranges with respect to different material granulation shown next to it. Obtained simulation results as well as granulometric size span and porosity values (which served as basis for the geometry generation) are depicted in the table on the right-hand side. A good general concordance of the measured and the computed data can be observed.

meso-scale on the permeability value calculated and later on used on the coarsest macro-scale. This correlation is tested on four different soil samples, within which the porosity value and two characteristic diameter sizes D_{50} and D_{90} are kept constant. The size of the domain has been kept constant 25 mm × 25 mm × 25 mm for all for cases with the finest mesh resolution $\Delta h = 25/128 \approx 0.2$ mm. The granulometric curves of each soil sample are given in Tab. 6.3, where it can be seen that in test case №1 the large and very fine granulations are dominant; in test case №2 only two sand grain sizes are present; test case №3 has a uniform volume ratio for all diameter classes; and in test case №4, unlike in test case №1, the medium-size granulations prevail over the largest and finest sand particles. This variation has a strong impact on the quantity of particles that should be randomly inserted, thus the same sample cannot be generated twice. Due to the different porous material spatial distributions, the velocity vector fields through the pores differ greatly, as illustrated in Fig. 6.4.
In order to calculate the permeability value for each test case, four different meso-scales have been introduced between the finest micro-scale and the coarsest macro-scale, as shown in Fig. 6.5. The corresponding permeability values at every mesh for each test sample have been calculated and depicted in Fig. 6.6, yielding the conclusion that different soil sample configurations can produce highly distinct permeability values if the volume averaging is performed on an insufficiently large sample size. Nevertheless, if this procedure is conducted over the entire micro-scale sample (see mesh 5, Fig. 6.5), the internal distribution of sand

Domain size	25mm x 25mm x 25mm							
porosity	55%							
d[mm]	case 1	Num.	case 2	Num.	case 3	Num.	case 4	Num.
15	24%	1	17%	1	11%	0	6%	0
10	12%	2	0%	0	11%	1	9%	1
8	7%	2	0%	0	11%	2	12%	3
6	5%	3	0%	0	11%	6	15%	9
5	2%	3	0%	0	11%	11	18%	18
4	5%	10	0%	0	11%	23	21%	43
3	7%	36	83%	414	11%	55	12%	58
2	12%	205	0%	0	11%	186	6%	98
1,5	24%	970	0%	0	11%	442	3%	117

Figure 6.3: Volumetric ratio and resulting total number of spheres packed in a domain of size 25 mm × 25 mm × 25 mm, shown for four different testing cases. Diameter classes, porosity values and refinement rates are kept constant in all four setups during the generation of the numerical domain.

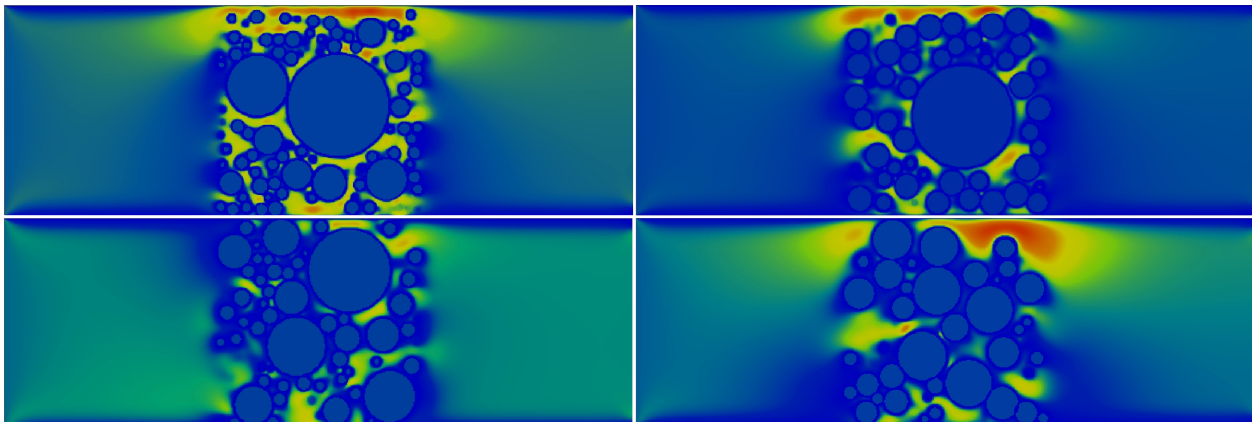


Figure 6.4: Cut through the computational domain of four different physical setups introduced in Tab. 6.6 - case 1 (top left), case 2 (top right), case 3 (bottom left), and case 4 (bottom right). Different spatial distributions of sand grains are to be observed, leading to the variations in the velocity and pressure fields, thus different all meso- and macro-scale parameters.

particles at the micro-scale should not cause large deviations from the expected permeability value at the macro-scale for the set porosity value.

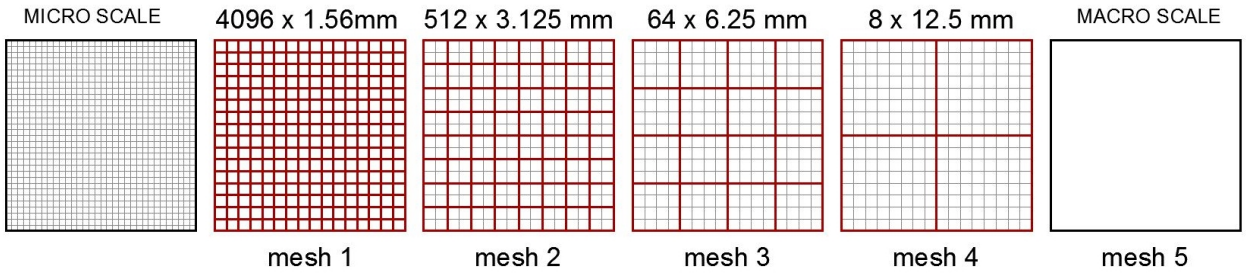


Figure 6.5: Meshes used on the meso-scale in order to compute permeability values $k\left[m^2\right]$ - 4096 different values calculated in the case of Mesh 1 up to one single value obtained for Mesh 5. These computed values are presented in Fig. 6.6.

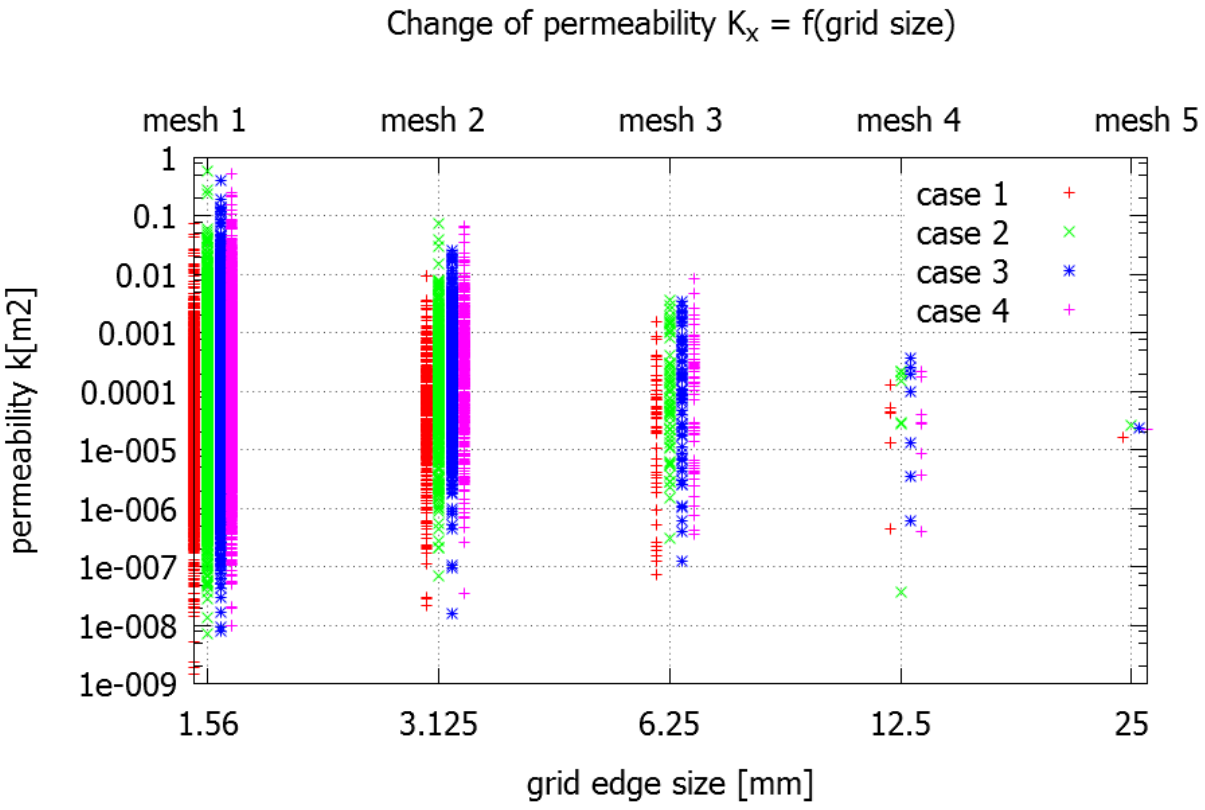


Figure 6.6: Change of permeability values within the domain, depending on the size of meso-scale chosen (see Fig. 6.5). The analysis is performed for four independent physical setups according to Tab. 6.3. A large variation of the permeability values on the finest scale (mesh 1, $\Delta h \approx 0.2$ mm) can be explained with the fact that the cells can be completely filled with the porous medium (permeability very small) or be entirely filled with water, resulting in a permeability value close to one. As can be observed, the resulting permeability values on the coarsest scale (Mesh 5) tend to be very similar, despite very different initial parameters set on the micro-scale.

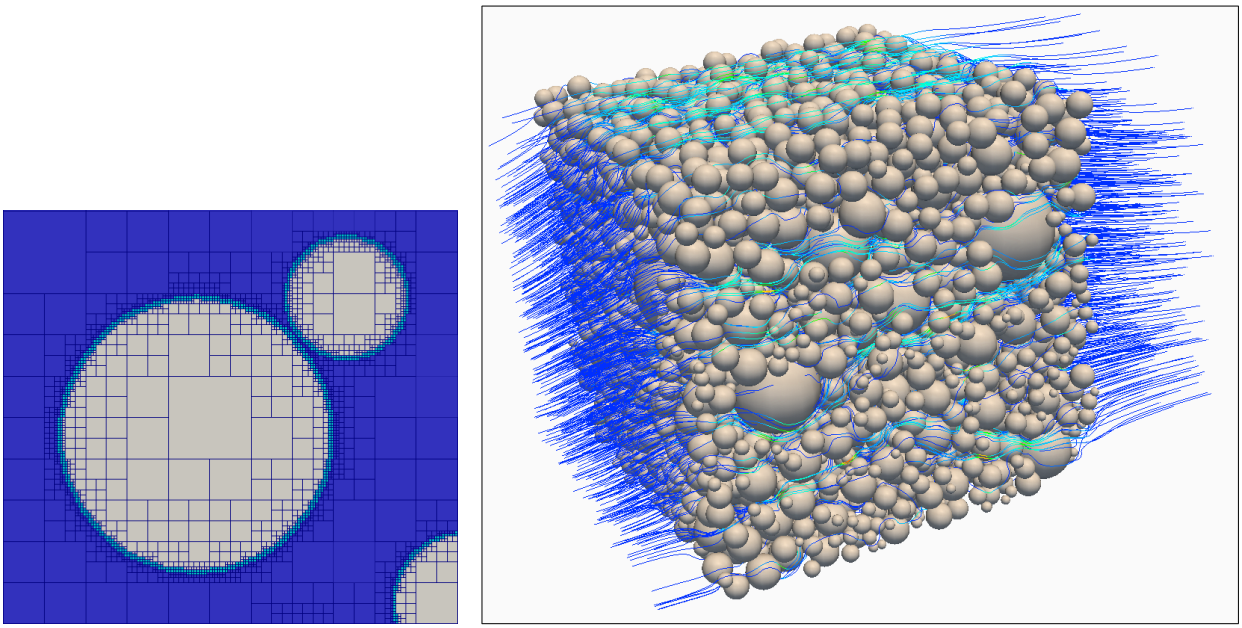


Figure 6.7: 2D cut through the numerical domain generated with an adaptive refinement strategy (left-hand side) and 3D view of the velocity field streamlines through an arbitrarily generated geometry (right-hand side). More details on the domain generation techniques can be found in [51].

Short résumé:

A new strategy, briefly explained in this chapter and published in [105], can be successfully used to produce large number of unique soil samples, given the fixed value of porosity. This value can be obtained in a controlled experimental environment, using destructive methods that are time-consuming and not repeatable. The numerical approach implemented offers a reduction of such experimental work to its minimum, allowing generation of synthetic samples, by imitating this destructive procedure (see Sec. 2.2.1 for more details). Up to hundred different samples have been generated in order to test an influence of different grain distributions on final permeability values at the macro-scale. A general conclusion is that besides the fixed porosity values given, at least one characteristic diameter (e.g. D_{10} or D_{60}) must be set as constant. Given such initial conditions, different, randomly generated samples can be used and a size of a meso-domain $H \geq 2D_{max}$, where D_{max} represents the maximum grain size used, delivers one integrated value of permeability, which is very similar across all the samples (see four tested samples in Figs. 6.3–6.5, for instance). The same conclusion is drawn in a larger simulation setup shown in Fig. 6.7, in the right-hand plot. Further analysis and improvements can be done by inclusion of a modelling methods that accounts for a solid-solid interaction between each two sand grains.

6.2 Shallow water model – Verification and application

Implementation of the shallow water module within the in-house MPFluid Framework is done according to the theoretical principles given in Sec. 3.4. In order to account for the complex terrain geometries, the approach in J. Hou et al. [66] is adopted, allowing the implementation of both first- and second-order spatial and temporal schemes. In this implementation, the goal is not a mere improvement of the shallow water approach by using higher-order spatial schemes, but rather to use lower-order schemes on a simple terrain configuration, while the full 3D Navier–Stokes model should be employed in regions where the complexity of the terrain and flow characteristics is higher. Following this idea, the shallow water module implemented is discretised using a first-order explicit scheme in time (see Sec. 4.1.7) and a first-order Riemann HLLC approach (as described in Sec. 4.1.5) to discretise the advection and pressure equation terms.

A vast amount of scientific work has been done since as long ago as 1970 in order to validate and verify shallow water models, whereby the majority of the validation work, in some way, referred to the experimental data published by J. C. Martin et al. [61] and several laboratory measurements published as a part of the IMPACT initiative (Investigation of Extreme Flood Processes and Uncertainty). In order to avoid unnecessary comparison between different results from various authors that refer to the same simulation cases, the decision is made to select one author who had done extensive work in this field and to compare his results published in [13], [14], [15] with the results obtained in this dissertation.

The main interest of the mentioned author involves numerical simulation based on the 2D shallow water formulation, which he integrated within his FORTRAN framework for HPC simulations of hazardous flooding and tsunami events (NUFSAW2D). Since 2011 he has succeeded in creating a powerful tool, capable of simulating the discontinuities in shallow water events, wet–dry phenomena near solid interfaces, also called moving boundary geometries, as well as highly turbulent flow occurrence using the RANS k–ε numerical model. A variety of schemes have been implemented since then, aimed at higher-order accuracy in space and time, some of which are the Runge–Kuta second-, third- and fourth-order accurate in time, Riemann Roe, HLL and HLLC spatial schemes, and the artificial viscosity Riemann-free numerical treatment for discontinuous flows. Entire sets of schemes are implemented using the finite-volume cell-centred scheme and they are applicable on different mesh settings (e.g. orthogonal, non-orthogonal, triangular, quadrilateral, mixed-mesh settings, etc.). Thorough validation and verification of the implemented and proposed schemes has been done, resulting in very good concurrence between the obtained results and the experimental data previously mentioned, as well as very good conformity with the mathematical considerations and verified data published by J. Hou et al. [66] and Q. Liang et al. [117].

For the purpose of verifying the work presented here, three well-known simulation scenarios are taken and simulated by the chosen author in such a way that precise imitation of the schemes with an identical order of accuracy is performed. In the next part the chosen setups are described in detail, followed by comparative analysis between the results produced in NUFSAW2D and HPC MPFluid.

6.2.1 Scenario 1: Flow around a prismatic obstacle

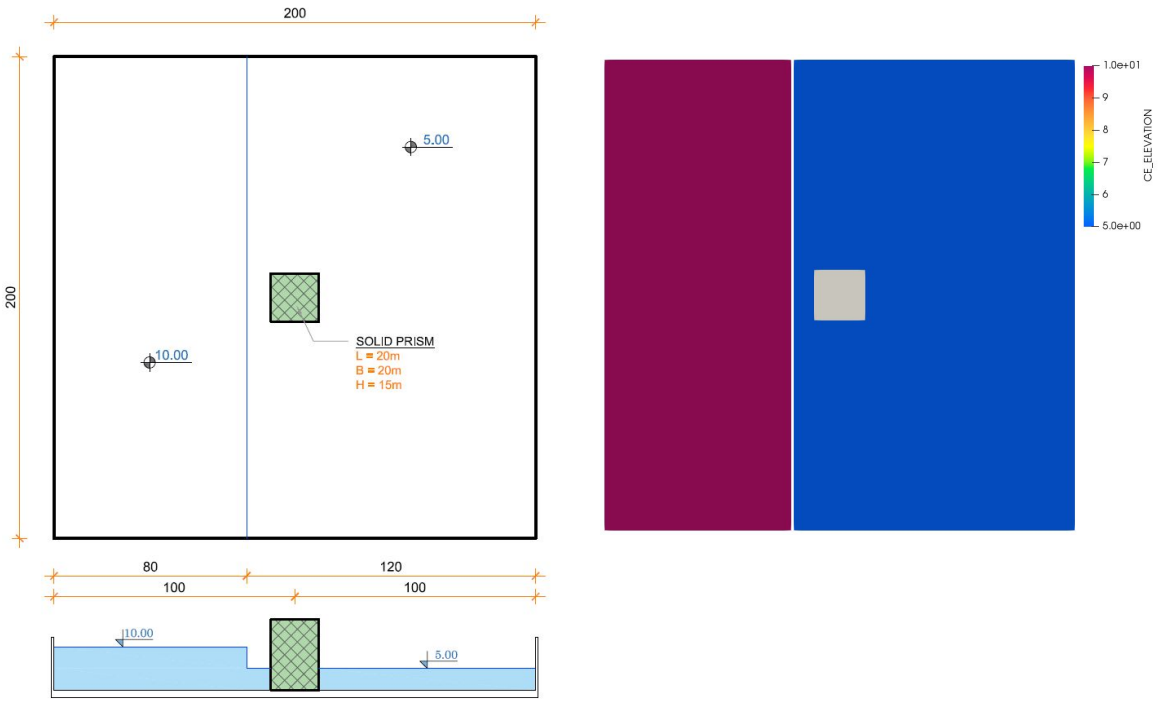


Figure 6.8: Physical and simulation setup: Flow around a prismatic obstacle.

This simulation case is a standard benchmark for verification of the capability of the code to deal with complex wave–structure and wave–wave interactions. A quadratic domain of size 200 m × 200 m is enclosed using solid, frictionless boundaries, as shown in Fig. 6.8. In order to test the wave–structure interaction, a prismatic obstacle of size 20 m × 20 m × 10 m is built in the middle of the domain with no permeable solid walls. For the sake of simplicity, the influence of friction from surface roughness throughout the domain has been ignored. The initial water depths are set to be 10 m and 5 m respectively in the first and second parts of the domain, divided by a thin wall, which should be removed immediately after the simulation starts, in order to create a dam-break effect. The numerical domain is generated using a uniform Cartesian mesh of size 0.5 m resulting in a total of 400 computing cells per direction. The total simulation time of 8 seconds is adopted with the fixed simulation time-step set to $\Delta t = 0.001$ s. Within the MPFluid framework it is possible to choose three different temporal discretisations: a) first-order explicit Euler, b) second-order Runge–Kutta (RK2) and c) third-order Runge–Kutta (RK3).

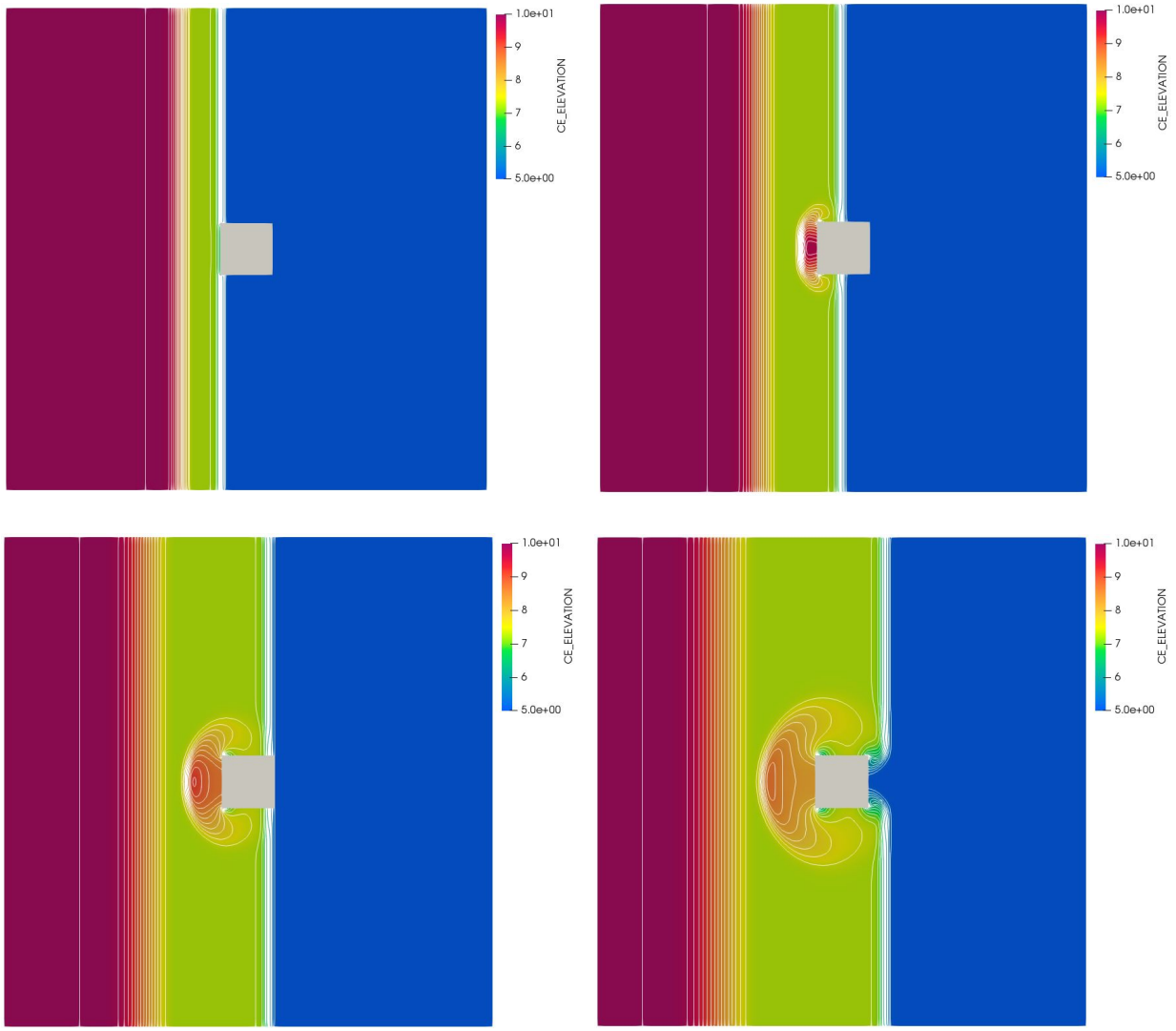


Figure 6.9: Water depth contours after 1 s, 2 s, 3 s and 4 s of the simulation. For $t = 1$ s the first wave-structure contact occurs, $t = 2$ s a backward shock-wave is generated and $t = 4$ s the travelling side waves reach the stagnation point behind the prismatic obstacle.

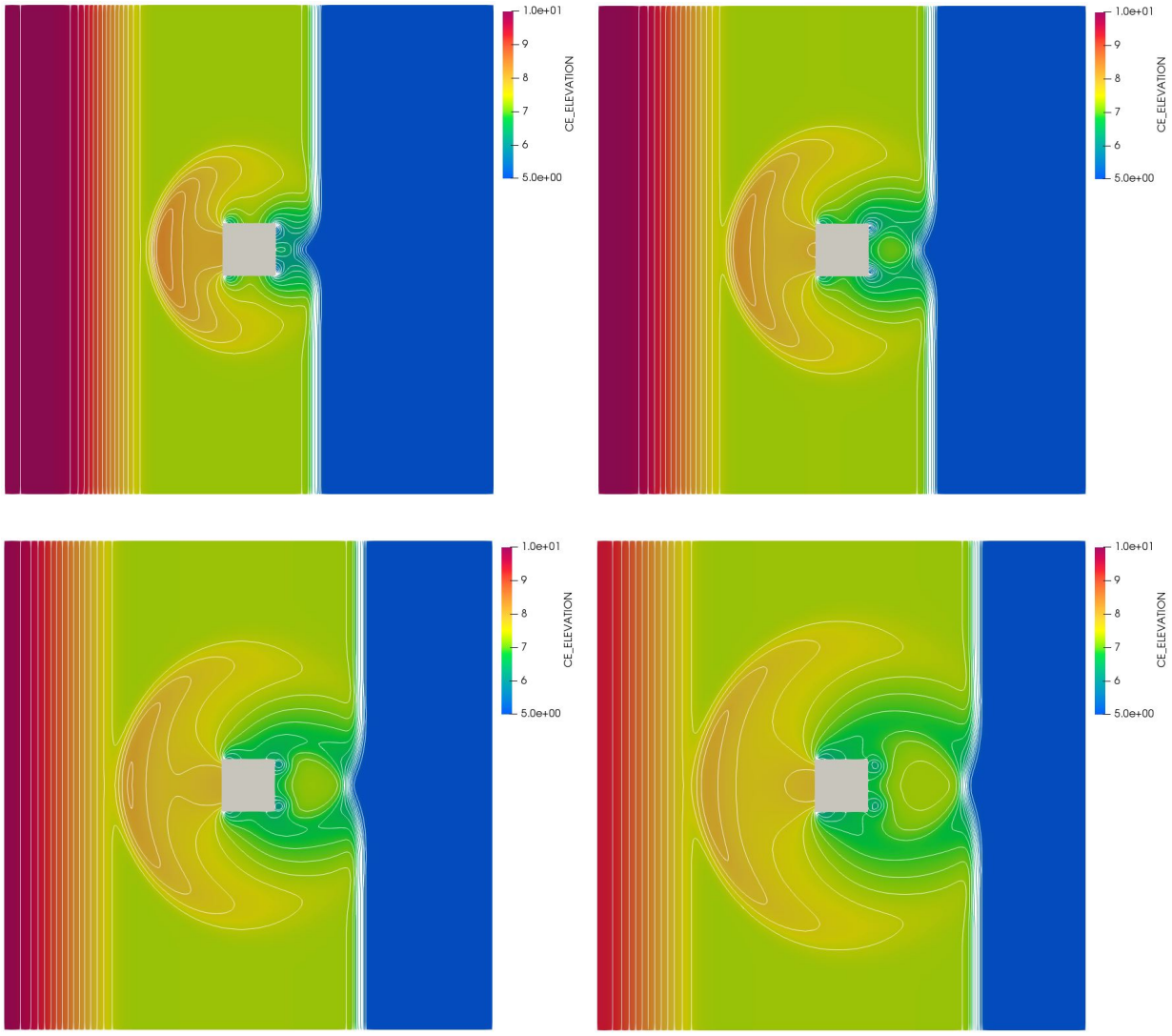


Figure 6.10: Water depth contours after 5 s, 6 s, 7 s and 8 s of the simulation. At $t = 5$ s the creation of the wake, soon after the stagnation point behind the prismatic obstacle is reached, is depicted. At $t = 6$ s, $t = 7$ s and $t = 8$ s a complex longitudinal and transversal travelling of the waves is observed, resulting in further wave–wave interaction.

The Runge–Kutta schemes offer better stability and more flexibility while setting the CFL condition. Nevertheless, for the chosen time-step size, the Euler first-order accurate temporal scheme does not produce any instabilities. The spatial terms of the SWE system of equations are discretised using a first-order accurate Riemann HLLC scheme.

The development of the flow through time can be seen in Figs. 6.9 and 6.10, where the water-depth contours have been depicted in one-second intervals, showing gradually the first contact between the fluid and structure, followed by the creation of the first backwards-moving shockwave that travel from the prismatic obstacle, then reaching the stagnation point behind the obstacle and further wave–wave interaction. The results obtained are in a good concurrence with the results published in [117] and a direct comparison between NUFSAW2D and MPFluid results is shown in Fig. 6.11.

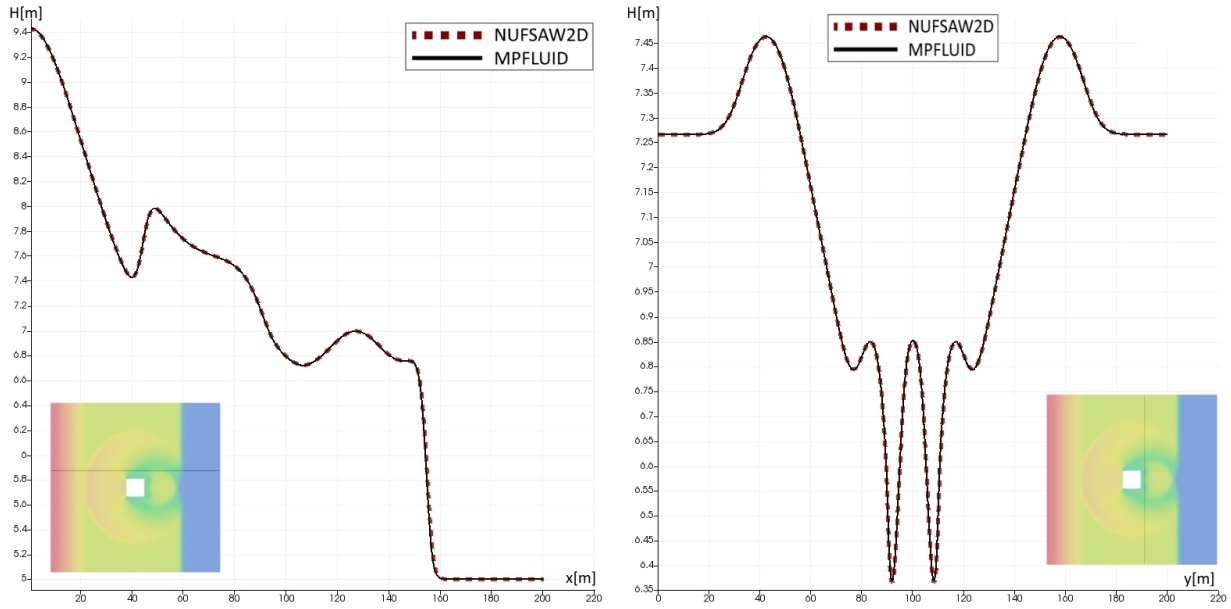


Figure 6.11: Direct comparison of the results obtained in NUFSAW2D and MPFluid frameworks. The first horizontal cross-section (left-hand side figure) is cut at the distance y = 120 m and the second vertical cross-section is made just behind the obstacle at x = 115 m, as shown on both small subfigures. The NUFSAW2D results are depicted with red dotted line, whereas the MPFluid results are represented using solid black line. A very good accordance of the results is achieved.

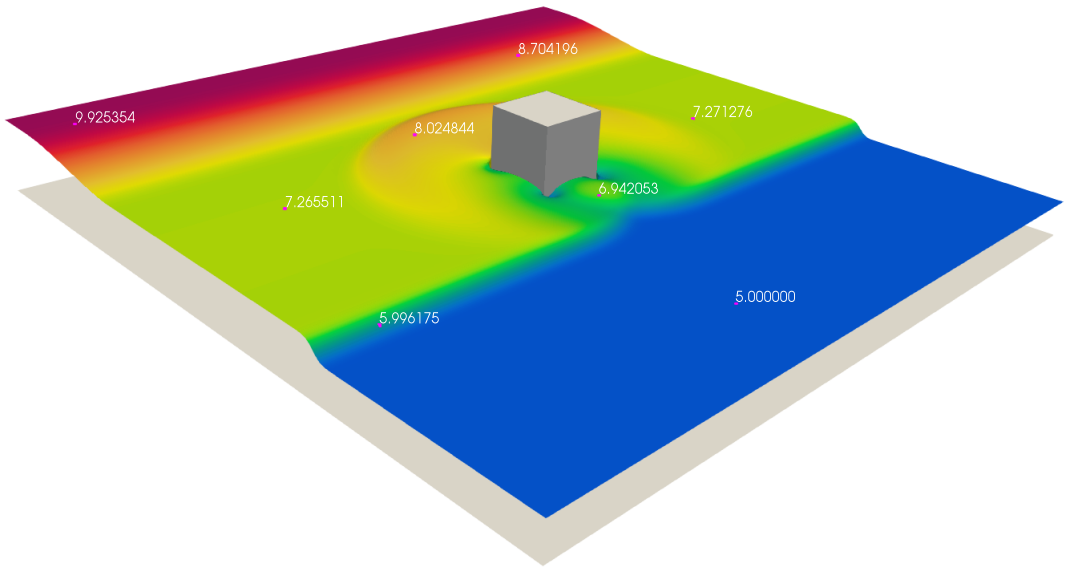


Figure 6.12: 3D representation of the simulation: Flow around a prismatic obstacle at t=6s

6.2.2 Scenario 2: Circular dam break

Another standard benchmark capable of proving accuracy in representing the shock and rarefaction waves of a particular scheme is the case of a circular dam break, introduced by E.R. Toro. As shown in Fig. 6.13, the dry conditions of the flood field are avoided by setting the water at rest over the entire domain (40 m $\times$ 40 m) with a uniform depth of $H_{min} = 0.5$ m. The circular dam occupies a cylindrical shape of radius $R = 2.5$ m and height $H_{max} = 2.5$ m, positioned precisely in the middle of the domain. This form of cylindrical dam requires both the shock and rarefaction waves to travel radially, with no preferred numerical direction, which is successfully reproduced within the MPFluid Framework (see Fig. 6.14).

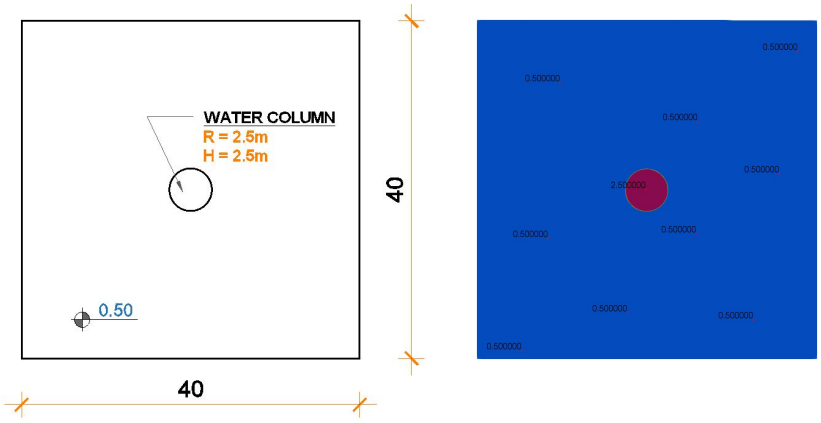


Figure 6.13: Physical and simulation setup: Circular Dam Break.

The numerical setup for this verification test case is made using a uniform Cartesian mesh of size $dx = 0.1$ m, resulting in 400 $\times$ 400 computing cells in total. Both the temporal and spatial schemes are of first-order accuracy, while the non-linear advection term is resolved using the HLLC Riemann solver. As in the previous verification example, the size of the time-step is $dt = 0.001$ s and this is kept constant during the entire simulation ($t_{total} = 5$ s).

In Fig. 6.14 the time sequences of 0.1 s, 0.4 s, 0.8 s, 1.6 s, 3.8 s and 4.7 s are depicted, pointing out the most relevant flow elements throughout the simulation. At $t = 0.1$ s the shockwave is generated, travelling outwards from the centre. At $t = 0.4$ s the rarefaction wave is formed and starts travelling inwards, while the shockwave is still travelling outwards, preserving a sharp front. This can be particularly seen in the 3D representation of the simulated test case in Fig. 6.15. Such a state is maintained until about $t = 1$ s, when the rarefaction wave implodes, followed by the reflection and, soon after, over-expansion. The consequence of this complex process can be seen at $t = 1.6$ s, when the water height drops below the initial water-at-rest state. As a result of the over-expansion, another backwards-moving shockwave is formed, propagating towards the centre, which will also undergo an implosion process, generating a sharp dip in the positive z-direction.
Direct comparison at a randomly chosen cross-section ($y = 20$ m) at two different time-steps, as shown in Fig. 6.16, reveals very good concurrence between the MPFluid and NUFSAW2D frameworks.

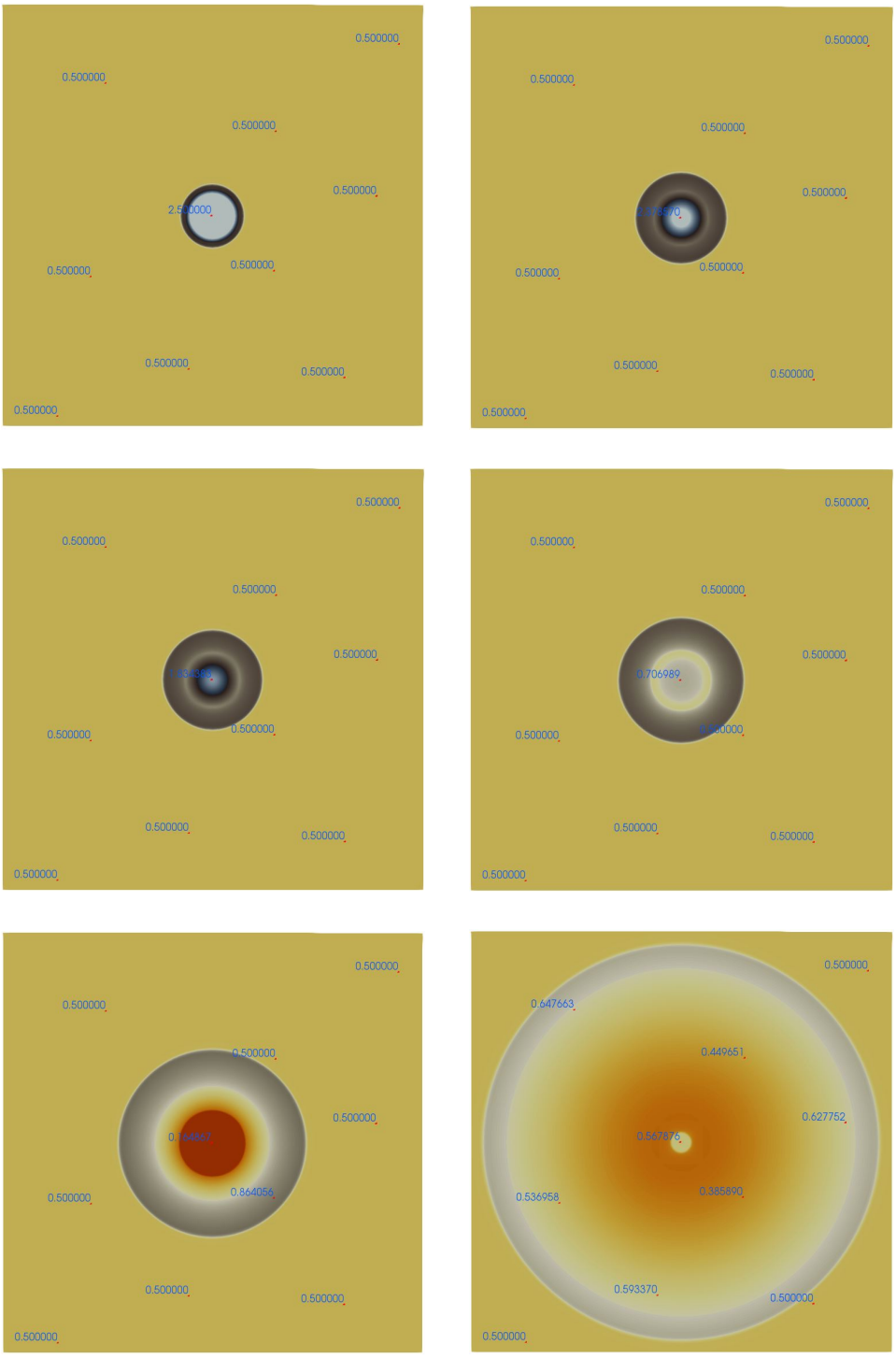


Figure 6.14: Different states of the circular dam break simulation at time sequence of 0.1 s, 0.4 s, 0.8 s, 1.6 s, 3.8 s and 4.7 s capture all important characteristic of the flow including the generation of the shock and rarefaction wave, their outward and inward propagation, and the implosion followed by the over-expansion. This set of processes is repeated until the fluid reaches the steady state rest position.

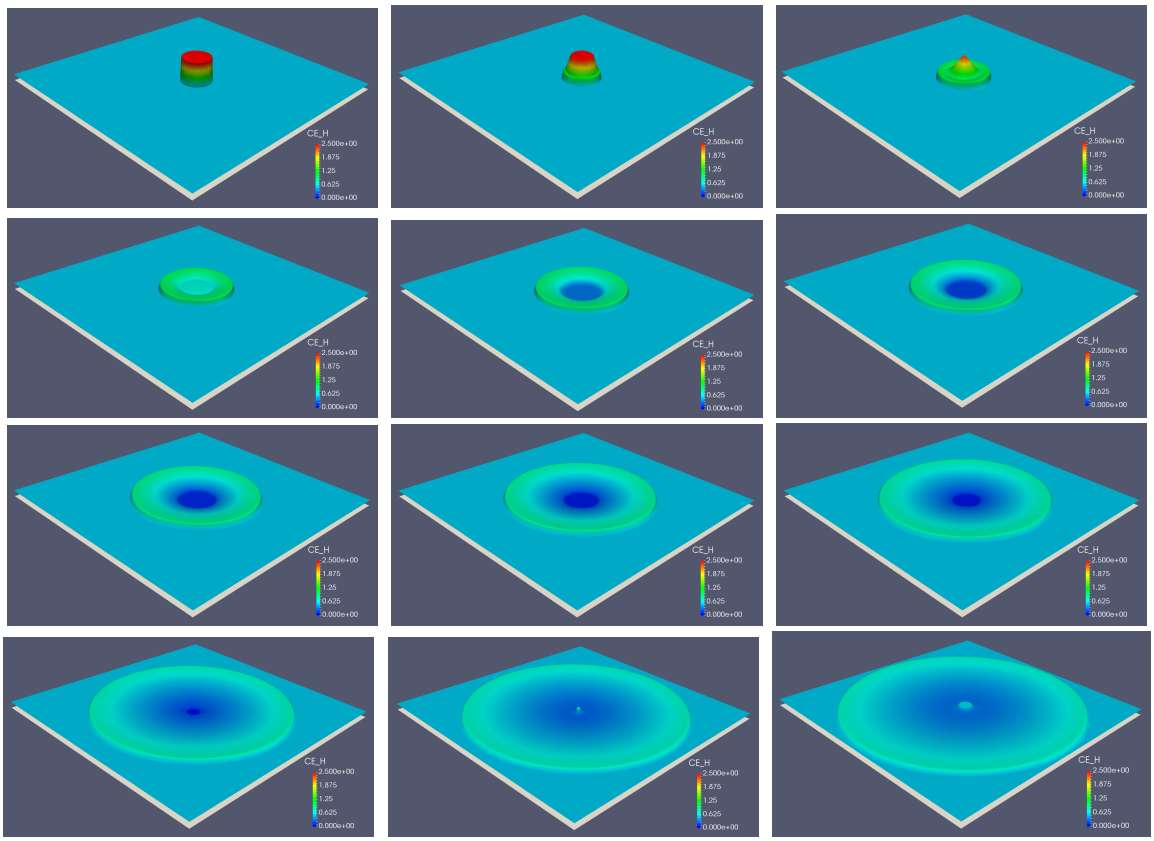


Figure 6.15: 3D representation of the simulated test case at $t = 0$ s, $t = 0.1$ s, $t = 0.4$ s, $t = 0.8$ s, $t = 1.6$ s,$t = 2.2$ s, $t = 3.8$ s, $t = 4.0$ s, $t = 4.2$ s, $t = 4.4$ s, $t = 4.6$ s and $t = 4.7$ s.

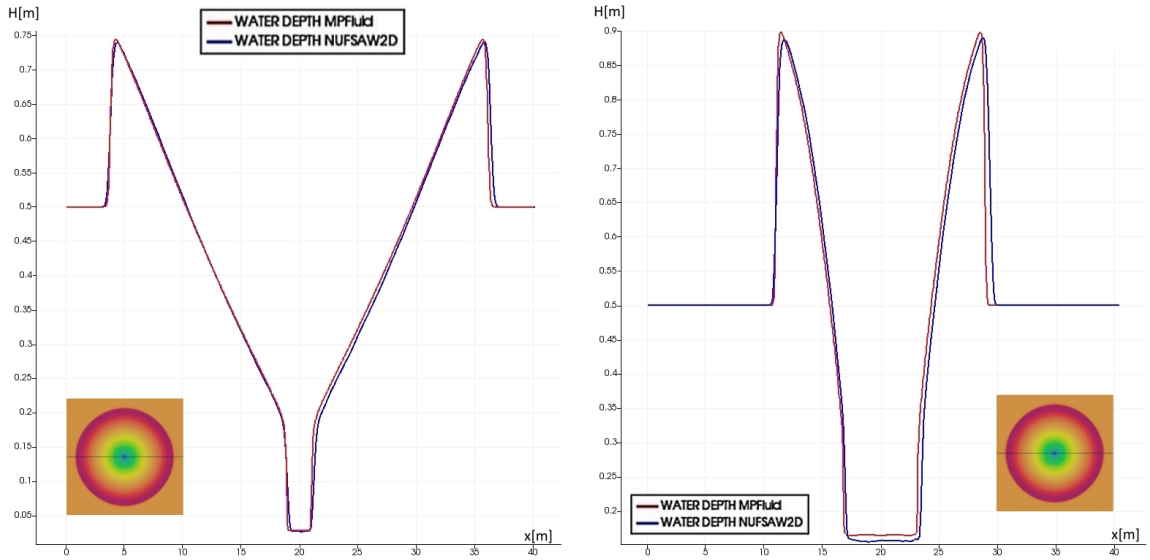


Figure 6.16: Comparison of the water depths obtained in NUFSAW2D and MPFluid frameworks at the cross-section y = 20 m – $t = 4.0$ s (on the right) and $t = 4.4$ s (on the left). The resemblance of the results has been tested in several cross-sections and at least 6–7 randomly sampled time-steps. A very good fit is observed without exception.

6.2.3 Scenario 3: Flooding of an idealised city

The third and final validation and verification scenario tests the ability of the numerical schemes implemented to reproduce the effects of water depth and velocities on an idealised city located in an inundation area during one flooding event. In order to set up one such event, a retention area is designed behind the abutment blocks and separated by a thin membrane from the downstream tailwater section, as depicted in Fig. 6.17. The downstream area is populated with buildings, in this particular case arranged symmetrically. The tailwater cross-section is not left completely flat; instead a complex trapezoidal form (see Fig. 6.18) is initially adopted to test the wetting and drying phenomena, and to force multi-directional wave–wave and wave–structure interaction. The buildings are of the same size in the x and y directions: 30 cm × 30 cm, with a uniform separation of 10 cm. The height of the buildings must be set in such way that no submersion occurs at any point of time. The entire experimental series is conducted in the Hydraulic Laboratory of the Civil and Environmental Engineering Department of the Université Catholique de Louvain, Belgium and the whole study is published by S. Soares-Franzão and Y. Zech [127]. As is described within the article, the downstream area is wetted with a thin layer of water $H_d = 0.011$ m due to the practical limitations of the experimental equipment. Several different experimental setups are performed, in which the buildings are aligned and rotated relative to the preferred flow direction, with the aim of capturing complex flow phenomena, such as a hydraulic jump in front of obstacles, a change of flow regimes in the streets within the city blocks over time, and the influence of transversal waves coming from the trapezoidal banks in the dominant flow direction. The water depth measurements are taken from gauge stations and the surface velocity profile is measured using particle image velocimetry (PIV). The assumption that the surface velocity values approximate the actual velocity profile well must be taken with care, especially in the narrow street regions, where, besides the depth-averaged profile, log-profile and local velocity circulations may occur as well. Nevertheless, in another study by the same author [126], it is discovered that for most of the time the surface velocity profile can be taken as a good measure of the depth-average velocity magnitude.

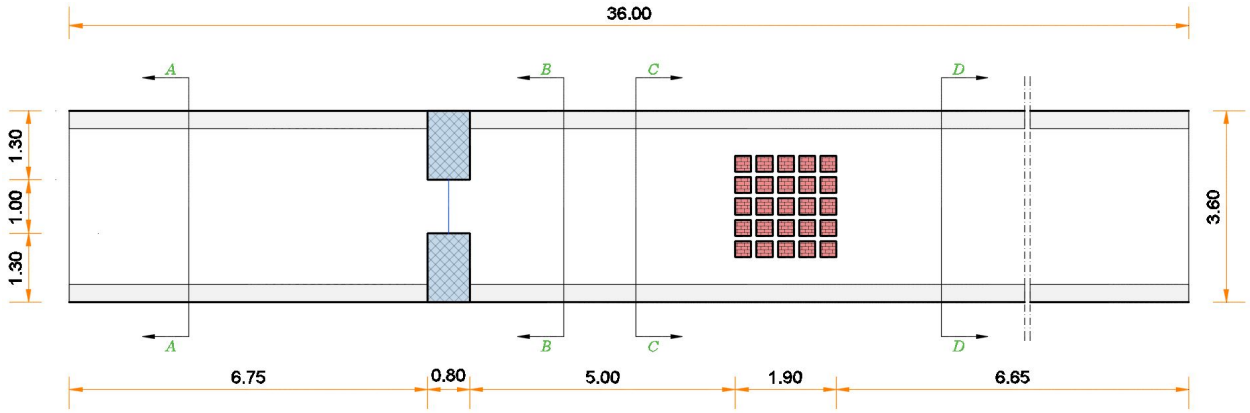


Figure 6.17: Plan view of the physical setup: Flooding of an idealised city. Four annotated characteristic cross-sections are depicted in Fig. 6.18. The initial water elevation in the retention area (left from the abutment blocks) is 0.4 m, whereas the tailwater downstream area is wetted with the minimum amount of water H = 0.011 m.

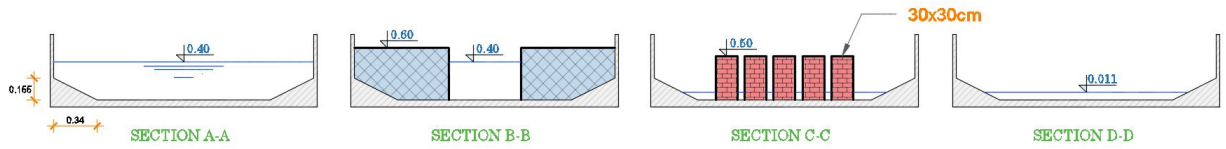


Figure 6.18: Four characteristic cross-sections of the physical setup: Flooding of an idealised city. The buildings and the abutment blocks must be tall enough to prevent submersion at any point of the simulation time.

This experimental data is used by B. M. Ginting [14] to validate the accuracy of the NUFSAW2D Framework, resulting in a very good agreement between the numerically produced and the measured results. Due to the fact that the model implemented in the MPFluid Framework does not account for the influence of friction, the validation of that model is not performed using the gauge data; instead, a new frictionless simulation with all the other parameters identically set is run in the NUFSAW2D Framework and the data obtained is used to verify the MPFluid simulation results.
The physical domain shown in Figs. 6.17 and 6.18 is discretised using a uniform Cartesian mesh of size $\Delta x = 0.01$ m resulting in a large mesh setup with over 1.2 million computing cells (3600 $\times$ 360 in total). This case corresponds to the third experimental case of S. Soares-Franzão and Y. Zech, allowing all the relevant phenomena mentioned to manifest themselves and to be clearly observed. As in the previous two verification cases, only first-order accurate temporal and spatial schemes are used, which, in combination with the small spatial step, leads to a severe limitation of the CFL number. In order to keep the numerical setup stable, the chosen time-step must be drastically reduced to $\Delta t = 10^{-5}$ s. Employment of higher-order temporal schemes such as RK2 and RK3 relax this limitation. The total simulation time is set to $T = 10$ s.

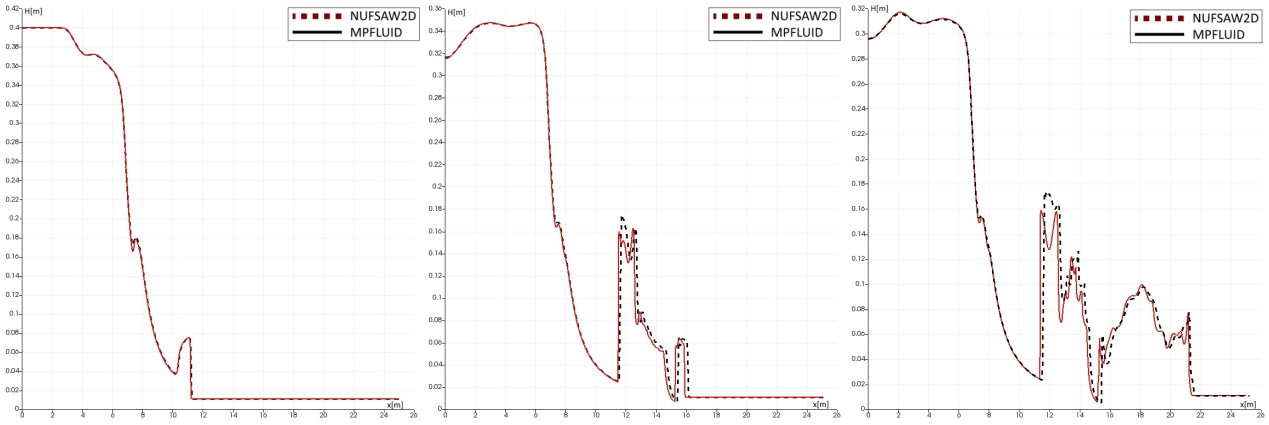


Figure 6.19: Direct comparison of water elevations obtained in the NUFSAW2D and MPFluid frameworks in the cut-plane between the third and fourth row of the buildings, looking from the bottom to the top in Fig. 6.17. The cut-plane is parallel to the x-axis and located at y = 2.0m. Left-most figure depicts the elevation at time $t = 2$ s, the middle one at $t = 5$ s and the right-most one at $t = 8$ s.

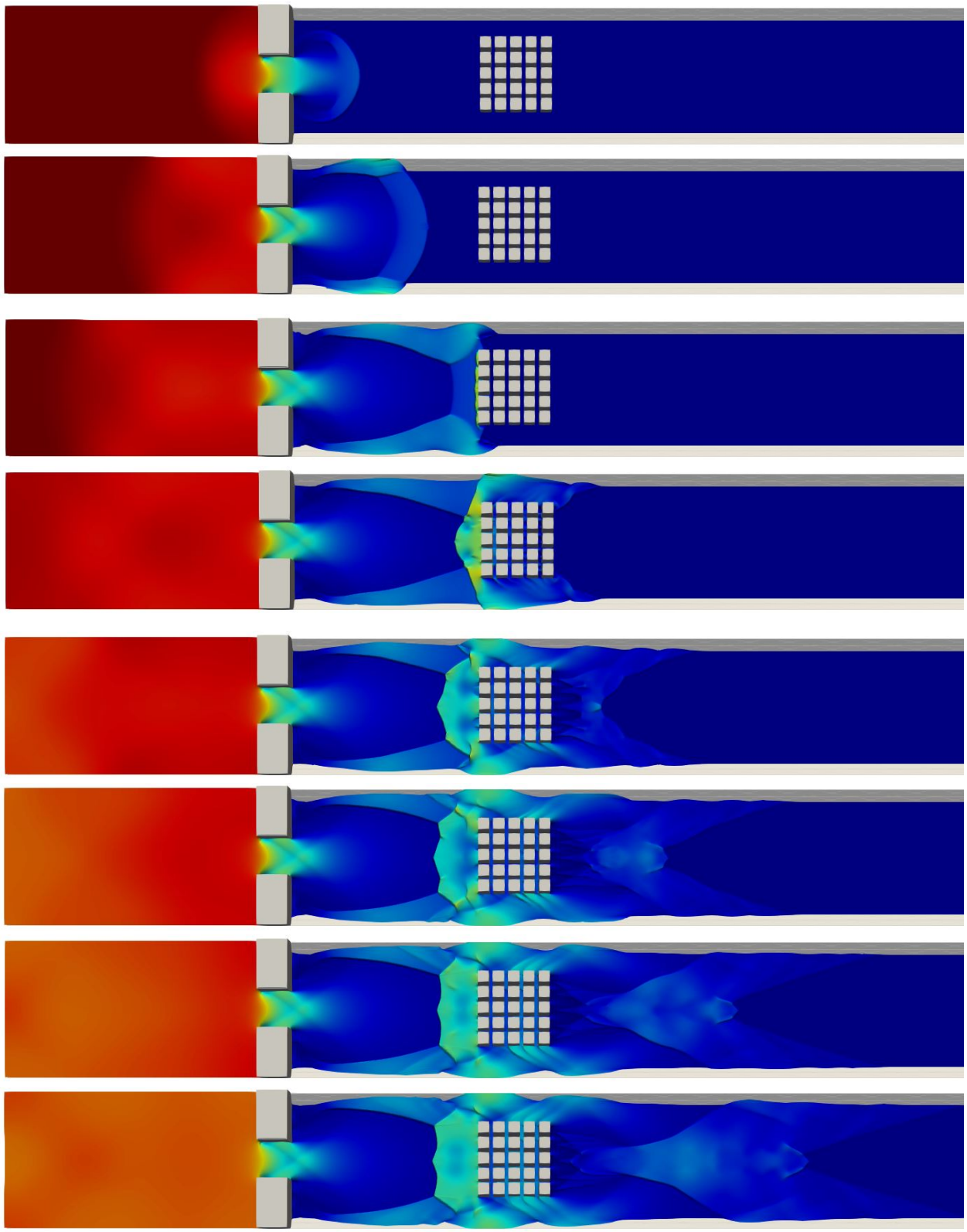


Figure 6.20: 3D representation of the flood through an idealised city for the first 8 seconds of the simulation. All relevant flow phenomena, such as hydraulic jumps, wetting, side reflections and wave interference can be observed in the chosen period of time.

Fig. 6.20 depicts the simulation of a flood event over an idealised city and every sub-figure corresponds to one second further on in the simulation. In the first plot, the generated breach can be seen after the control gate has been removed. This is followed by the widening of the flow and wetting of the dry side banks. Only at $t = 3$ s does the flow reach the city blocks and the first hydraulic jump is to be observed in the frontal area. The water elevation between the city blocks still remains quite low. At $t = 4$ s a backwards-moving shockwave is generated, travelling outwards from the buildings and, at the same time, the level of water between the building rises gradually. Already halfway through the simulation timewise, interactions between the longitudinal and transversal waves are obvious, and due to the progression of the waves going around the entire building complex, a wake on the downstream side is generated. This process is further developed until the end of the simulation, resulting in an overall increase in the water level and a reduction of the upstream hydraulic jump.

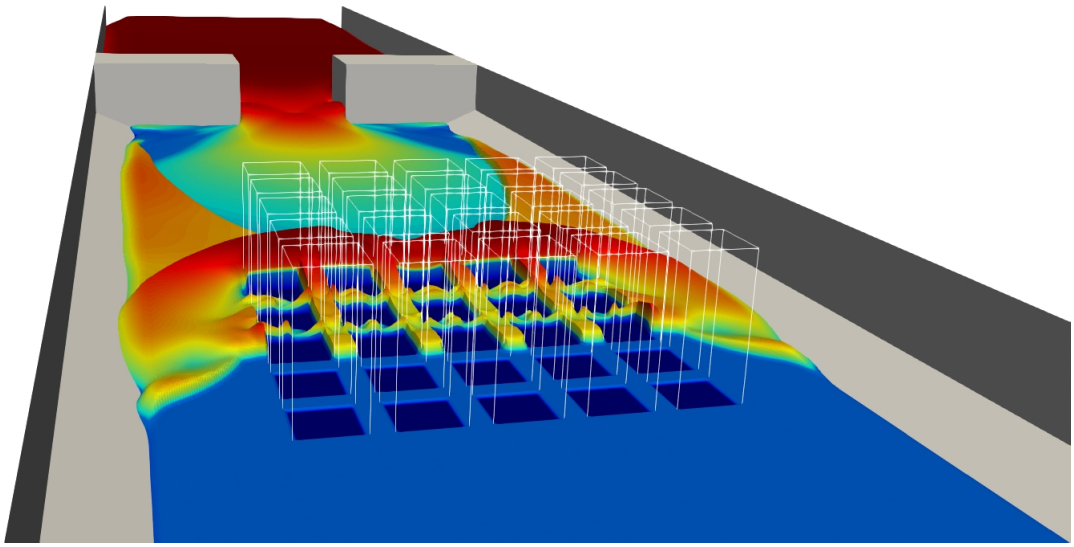


Figure 6.21: 3D representation of the irregular water level increase between the building blocks, due to the complex wave-wave and wave-structure interactions.

6.3 3D Shallow Water application setup on regular grids

The application of the 3D Shallow Water model (3D–SWE), explained in detail in Sec. 3.5 is illustrated within this section, followed by an efficiency study of that same solver coupled with the 2D Shallow Water model (2D–SWE). The coupling strategy used in the example that follows is given in Sec. 4.2.2, emphasising the treatment of the variables exchanged at the coupling interface.

To compare the two-dimensional and three-dimensional Shallow Water models, a simple dam-break scenario of size 1 m $\times$ 0.25 m has been established and performed for a total period of $T = 2$ s, with the time-step size set to $\Delta t = 0.0005$ s. Within the 2D model the calculation of the average water depth H and mean velocity vector in the two horizontal directions is conducted, as well as reconstruction of the level-set indicator and static pressure, based on

the average depth. The 3D model incorporates all the steps conducted in the 2D scenario, adding the influence of the dynamic pressure component, calculated using the Chorin projection method (as explained in 3.5 and 4.2.2). Introduction of a dynamic pressure component leads to the reconstruction of the velocity vector in all three spatial directions.

As can be seen in Fig. 6.22, the topmost and bottommost subplots depict the 2D and 3D models respectively, yielding very good concurrence in the interface advection. Nevertheless, the 2D free surface exhibits smoother curvatures globally, eliminating the local surface deformation present in the 3D model. On the other hand, the 3D model deals with the non-averaged velocities, which introduces a more complex vertical behaviour (see Fig. 6.22, $t = 1.20$ s). The

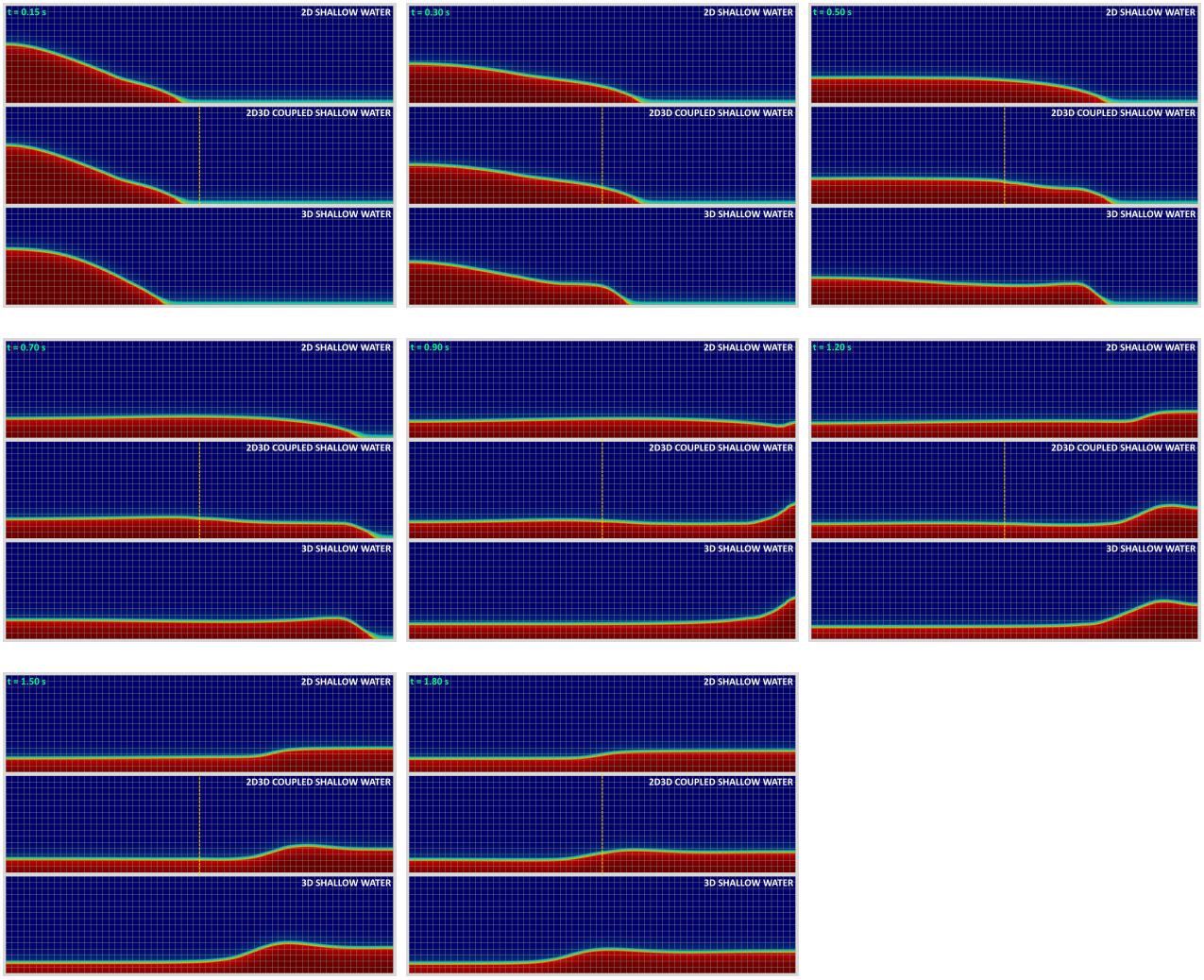


Figure 6.22: Comparative view of the 2D Shallow Water (upper subplot), 2D–3D Coupled Shallow Water (middle subplot) and 3D Shallow Water (bottom subplot). The total simulation runtime is $T = 2$ s and the subsequent figures, illustrated from left to right show the flow development at $t = 0.15$ s, $t = 0.30$ s, $t = 0.50$ s, $t = 0.70$ s, $t = 0.90$ s, $t = 1.20$ s, $t = 1.50$ s and $t = 1.80$ s. All three solvers result in the similar behaviour regarding the level-set interface advection, while the coupled and pure 3D solvers include more complex vertical phenomena due to the inclusion of the dynamic pressure and vertical component of the velocity vector $\mathbf{U}$ (see Fig. 6.23).

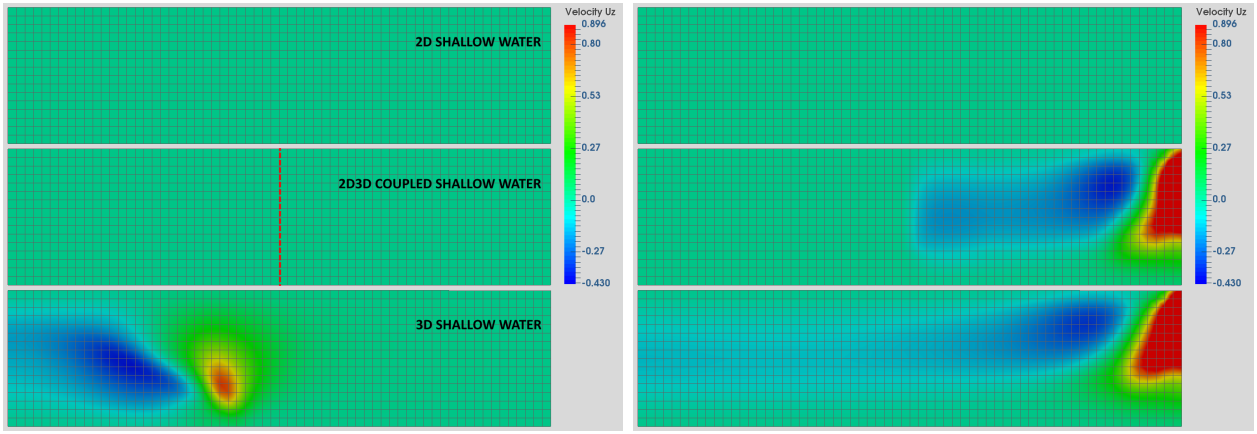


Figure 6.23: Depiction of the vertical component of the velocity vector $\mathbf{U}_z$ in the 2D Shallow Water (2D–SWE, upper subplot), 2D–3D Coupled Shallow Water (2D3D–SWE, middle subplot) and 3D Shallow Water (3D–SWE, bottom subplot) at $t = 0.15$ s and $t = 0.90$ s. The coupling interface between 2D–SWE and 3D–SWE is set in the middle of the domain (red dashed line), having the 2D–SWE model on the left side and 3DSWE on the right side of the domain. At $t = 0.15$ s the propagation wave is entirely located inside the 2D model (see Fig. 6.22), resulting in the $\mathbf{U}_z = 0$. Only after the wave passes through the coupling interface leading to the inclusion of the dynamic pressure component, the vertical component $\mathbf{U}_z$ is computed on the right side of the domain. The 2D–SWE subplot remains empty throughout the simulation time, whereas the 3D–SWE plot computes the vertical velocity component in the whole domain, during the entire simulation time $T = 2$ s.

inertia effects are also stronger within the 3D model, pulling more water to the right once the initial wave reaches the domain's right-hand wall. This phenomenon, as observed, does not influence any further development of the simulation, as the return wave in both cases reaches the domain's left-hand wall simultaneously.

A coupled 2D–3D system combines the simplicity of the 2D model and the accuracy of the 3D model, balancing at the same time the total runtime between the very cheap 2D and extremely expensive 3D computation. As published in [51], 85% of the performance is used for solving the Poisson pressure equation introduced in the 3D model, thus a reduction of the 3D domain size should lead to a significant reduction in the simulation time. In order to test this assumption, a simple rectangular domain is taken and divided into two equal parts (see Fig. 6.23, middle subfigure): in the left part of the domain the 2D–SWE model is applied, leaving the right part of the domain for 3D–SWE model. The level-set indicators reconstructed in the 2D region and calculated in the 3D region match to a great extent, resulting in a smooth transition of the free surface between the 2D and 3D models (see Fig. 6.22, middle subplot). In Fig. 6.23, the vertical component of the velocity vector U_z is depicted and as expected, in the 2D model (topmost subfigure) this data cannot be recovered, the 3D model (bottom-most subfigure) contains complete set of data, while the coupled 2D–3D model contains the proper information only in its right (3D) part. The Poisson pressure equation is solved in both the entire 3D model and 3D region of the 2D–3D coupled model using the Jacobi iterative solver with the precision set to $\epsilon = 10^{-8}$. This introduced an increase in runtime in comparison to the pure 2D model. In order to explore this expected decrease in performance, the time measurement is done

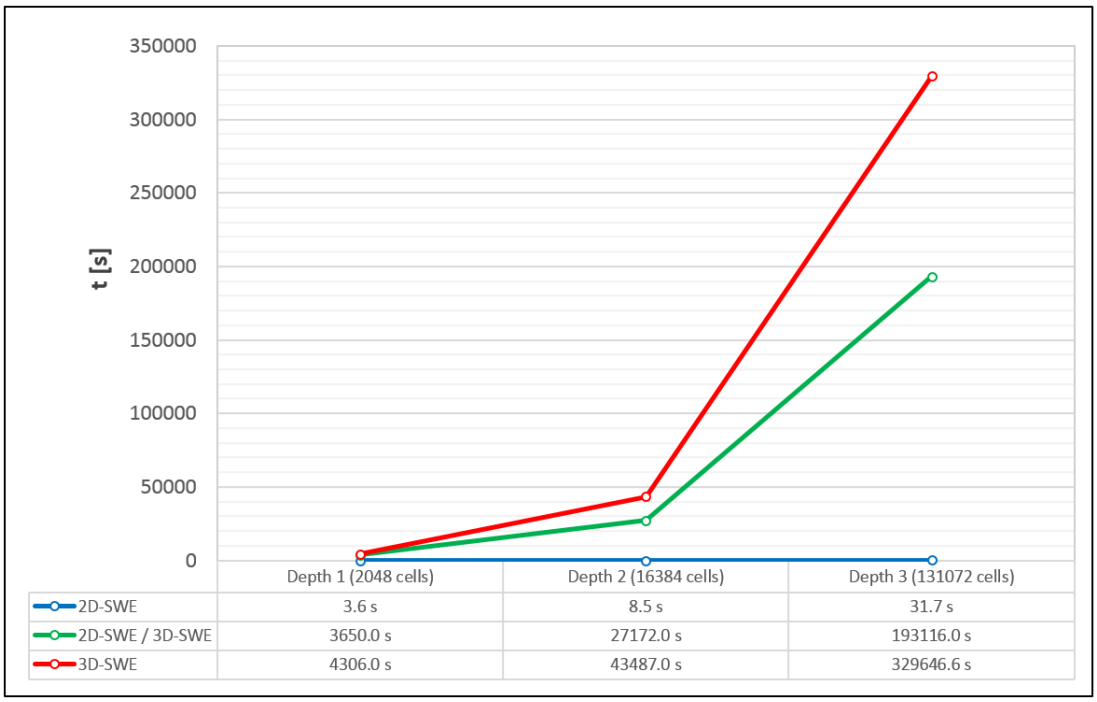


Figure 6.24: The runtime of the shear 3D Shallow Water and coupled 2D–3D Shallow Water simulation compared to the runtime of the pure 2D–SWE model. The simulation setup consists of a dam-break wave initially located on the left side of the domain. Nine simulation scenarios in total are conducted, utilising three different levels of refinement and three different models. As shown in Fig. 6.23 the interface boundary between the 2D and 3D models in the coupled system is set in the middle of the domain, reducing the size of the 3D domain to 50% of the initial size. The total runtime of the coupled system (solid green line) is consequently reduced to 84.8%, 62.5% and 58.6% of the time needed for the pure 3D-SWE scenario (solid red line), respectively for the refinement depths 1, 2 and 3. All nine cases are run on a local machine Acer Aspire R13, 2.3GHz, sequentially (n=2).

for a purely 2D, a purely 3D and a 2D–3D domain. Moreover, three different levels of grid refinement are utilised, resulting in a total of nine simulation setups, where the second grid setup had eight times as many, and the third setup 64 times as many computing cells than the initial grid setup. All the setups are run on a local machine Acer Aspire R13, 2.3GHz, sequentially (n=2). In the initial grid setup (Fig. 6.24, depth 1), due to fewer computing cells, the difference in the 2D–3D and 3D simulation durations is not significant (Wall-time$_{2D3D}$ = 84.8% × Wall-time$_{3D}$). This difference increases with the subsequent grid refinement, as illustrated in Fig. 6.24 (depth 2 and 3), leading to a 37.5% and 41.4% runtime reduction compared to the purely 3D computation.

6.4 3D Navier–Stokes model & Level-set interface reconstruction

6.4.1 Kleefsman's dam-breaking experiment – Validation and verification setup

In order to validate the 3D Navier–Stokes model combined with the Level-set interface reconstruction, Kleefsman's 3D dam-breaking experiment, conducted by the Maritime Research Institute in the Netherlands, has been utilised. The experimental setup imitates green water flow over a ship's deck (i.e. a large amount of water on the deck of the ship as a result of massive waves during large ocean storms) and its impact on the shipping containers being transported. The model is simplified to one single container positioned symmetrically to the ship's deck, keeping the amount of water constant throughout the measurement time. No complex outflow and inflow boundary conditions have been imposed. The deck and the container are 3.22 m × 1 m × 1 m and 0.403 m × 0.161 m × 0.161 m large, respectively, and their exact positions, including the 0.55-metre-high initial wave are illustrated in Fig. 6.27. The release of the water column is assumed to be instantaneous, after which the pressure and water height are measured at four pressure points P_1, P_3, P_5 and P_7, and H_1, H_2, H_3 and H_4, respectively, as depicted in Fig. 6.27. In order to compare the current results, both the validation (K. M. T. Kleefsman et al. [73]) and verification data (V.-T. Nguyen and W.-G. Park [141] and T. Fondelli et al. [133])

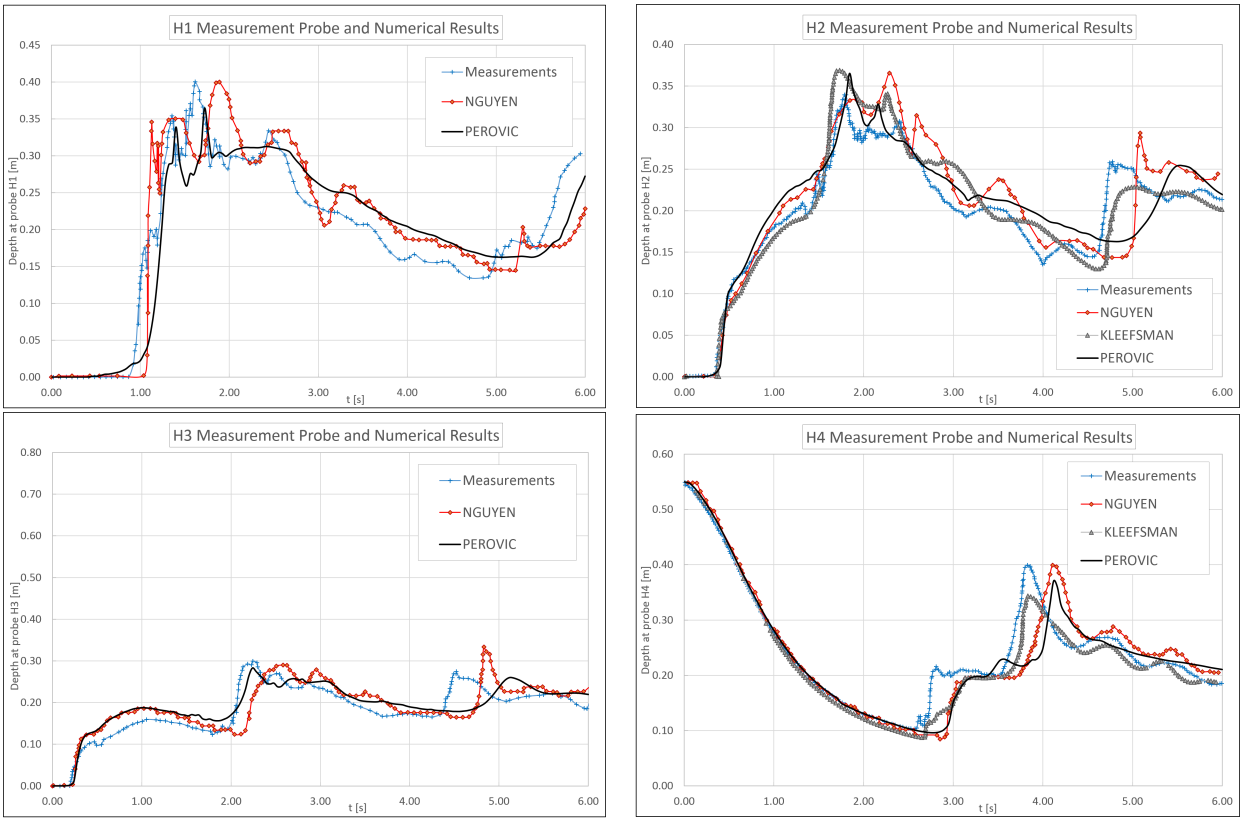


Figure 6.25: Water height measured in four different locations, illustrated in Fig. 6.27, and compared with the additional verification data, published in [141] and [133].

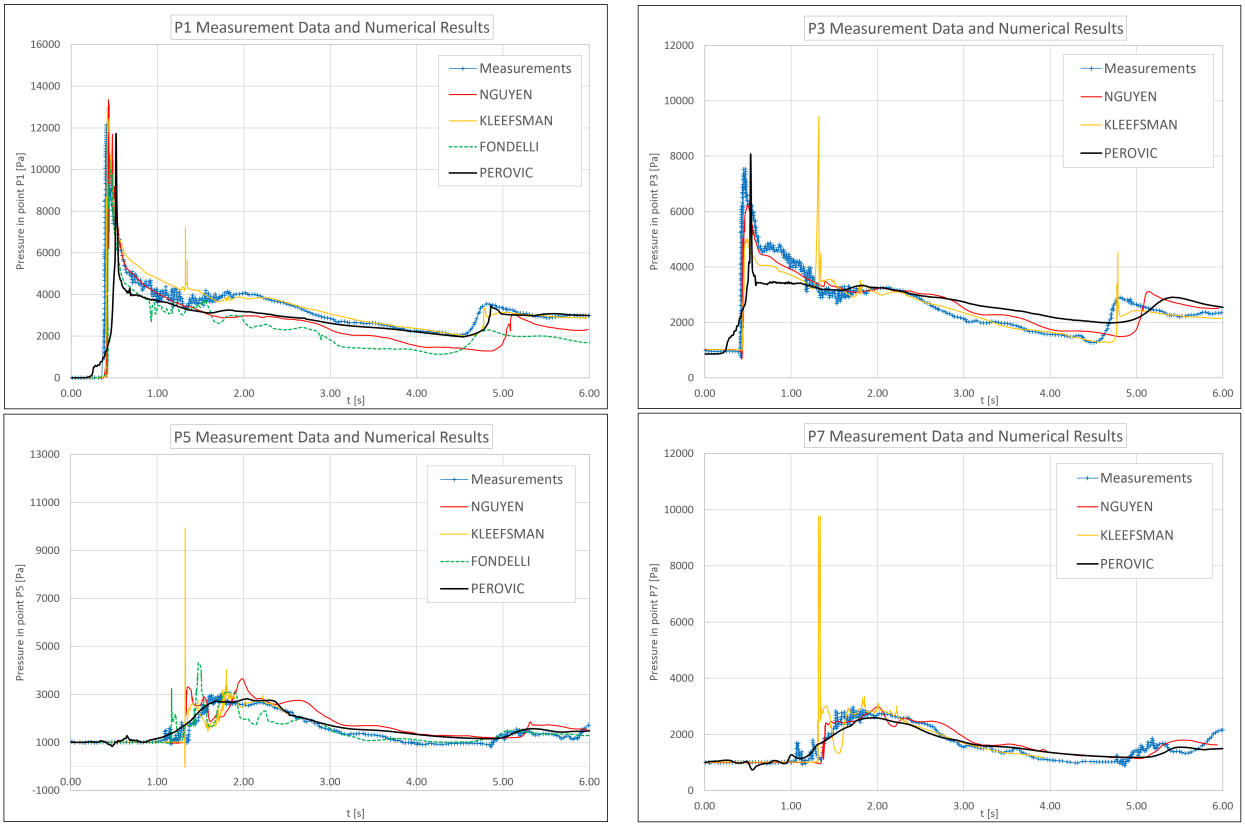


Figure 6.26: Pressure values measured at the surface of the transport container at four different points, and compared with the additional verification data, published in [141] and [133].

have been digitised using WebPlotDigitizer from Rohatgi [120] and the results are presented in Figs. 6.25 and 6.26. All the published verification data used the Volume-of-fluid interface reconstruction technique, thus slight differences in the water-level recordings are to be expected. The only publication found which utilises the level-set reconstruction technique (E. Schillaci et al. [40]) could not be properly digitised due to the black-and-white graphical representation and usage of two very similar line types; nonetheless the visual and quantitative interpretations of the results are in good accordance with the results presented in this work.
A closer look at Fig. 6.26 shows that the initial pressure wave at the experimental point P_1 is underestimated, whereas the further pressure development matches the experimental data. This phenomenon results in a slight phase shift in the measurements of height in the downstream probe H_1 (see Fig. 6.25). Nevertheless, it has been observed that the distribution of pressure over the area where the points P_1 and P_3 are positioned is influenced by many different factors, such as the level-set reinitialisation process and the interface thickness, type of solid boundary cells, mesh arrangement, gradient reconstruction technique chosen, etc. Moreover, in the current case the pressure values are higher in the immediate vicinity of the experimental points than at the measured points themselves. The captured pressure data at points P_5 and P_7, as well as the upstream water depth values at probes H_3 and H_4, is in very good concurrence with both the validation and verification data depicted.

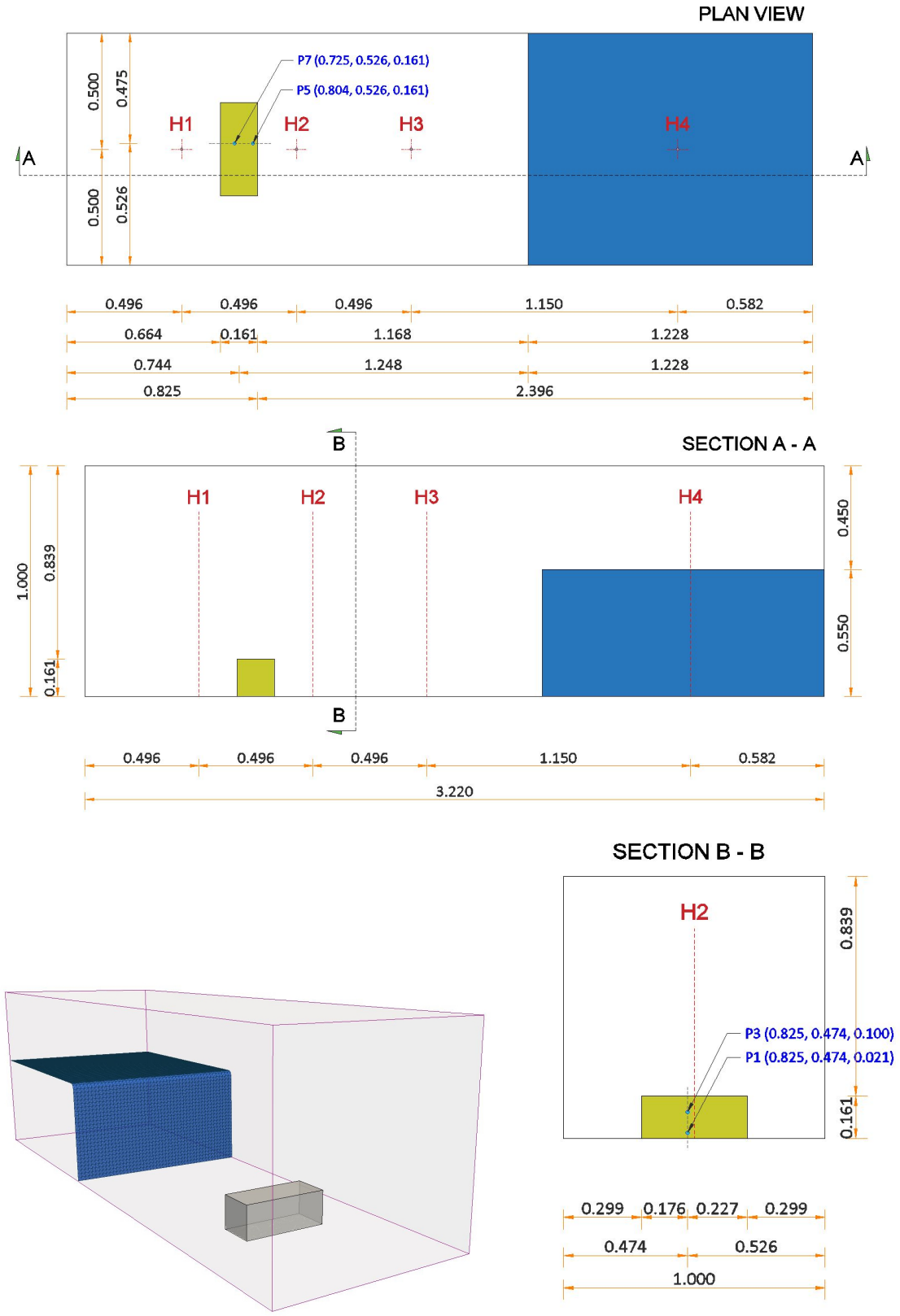


Figure 6.27: Experimental setup of green water flow over a ship's deck. The rectangular obstacle imitates a shipping container on the deck, overflowed by a large wave during an ocean storm.

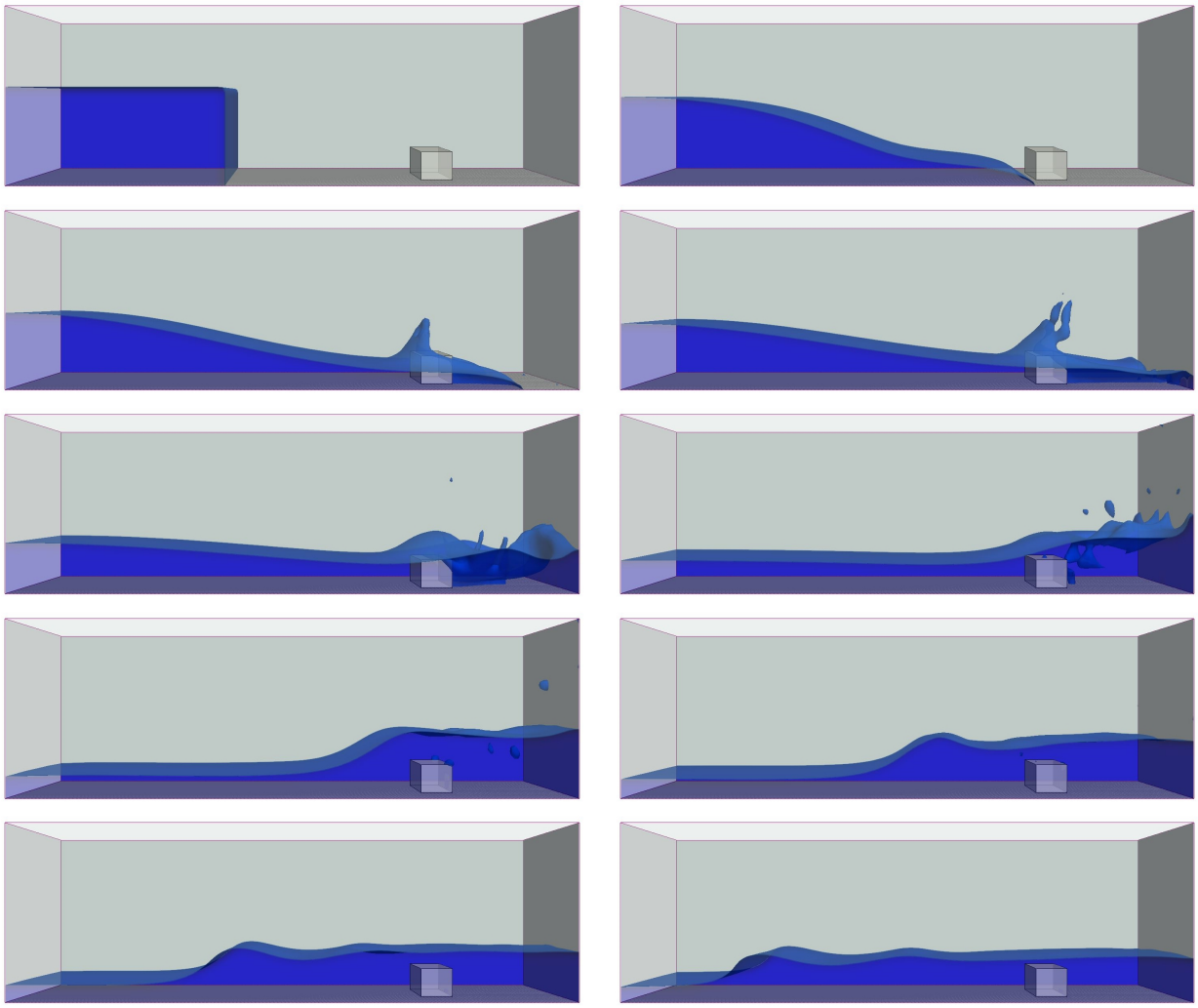


Figure 6.28: Development of the simulation process at time sequence $t = 0.0$ s, $t = 0.4$ s, $t = 0.56$ s, $t = 1.0$ s, $t = 1.4$ s, $t = 1.8$ s, $t = 2.0$ s, $t = 2.4$ s, $t = 2.8$ s and $t = 3.0$ s. The water front reaches the obstacle at t = 0.4s, and shortly afterward an increase in pressure and decrease in the velocity are observed in the frontal area of the obstacle. The flow separates into two main directions, one passing beside the obstacle and reaching the end of the simulation deck, and another going in the vertical direction, losing the kinetic energy and falling down over the obstacle under the influence of the gravitational force. Both flows are reunited at $t = 1.8$ s, going towards the other end of the deck. Pressure and water depths' data obtained are compared against the validation data in Figs. 6.25 and 6.26.

6.4.2 Falling Water Bubble Setup – simulation of multiple fluid bodies

In this section, the Falling Water Bubble problem is analysed, giving insight into several further benefits of the 3D solver: the merging and separation of multiple fluid bodies in the vertical z-direction, simulation of both the air and water phases at the same time and their mutual interference, as well as the influence of the mesh discretisation on the physics of the problem. The simulation domain is of size 6 m × 6 m × 6 m, with two independent water bodies: a water tank (2 m deep) and a spherical bubble of diameter $D_{sp} = 1.5$ m, centred at C = {3, 3, 4.5}, as depicted in Fig. 6.29. The mesh discretisation sensitivity is studied on an artificial test case: $\rho_1 = 1000$ kg/m^3, $\rho_2 = 100$ kg/m^3, and T1 = 40^3, T2 = 80^3 and T3 = 160^3 computing cells. As can be seen in Fig. 6.30, in all three cases the interface thickness is well resolved and remains constant with regard to Δx, throughout the simulation time. Nonetheless, droplet separation tends to happen at a smaller scale, which is why such droplets can be observed only in the finest T3 test case. Case T2 depicts slight changes in the interface shape due to the better resolved pressure from the second fluid (Fig. 6.29, blue-coloured phase), whereas the test case T1 has a smooth average interface representation.

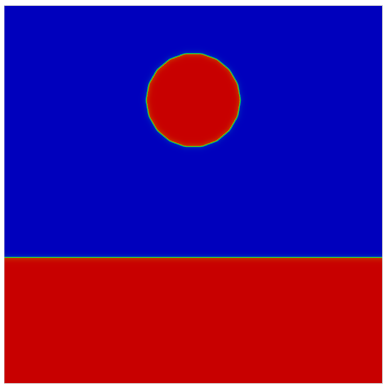


Figure 6.29: Initial state of the simulation setup: Falling Water Bubble case.

Mass conservation in all three cases is satisfactory, as reported in Secs. 3.6.3.3 – 3.6.3.4. All the remaining effects are analysed on a regular water–air test case: $\rho_1 = 1000$ kg/m^3, $\rho_2 = 1.204$ kg/m^3, with the T3 mesh refinement. The results depicted in Fig. 6.31 show that the high velocity value in the z-direction eliminates the influence of surface tension, thus the spherical cross-section deforms on the way to the water tank. Moreover, it can be seen that the deformation is initiated approximately at the separation points, symmetrically. As reported in [35], it is expected that the air located under the sphere creates a minimal depression at the

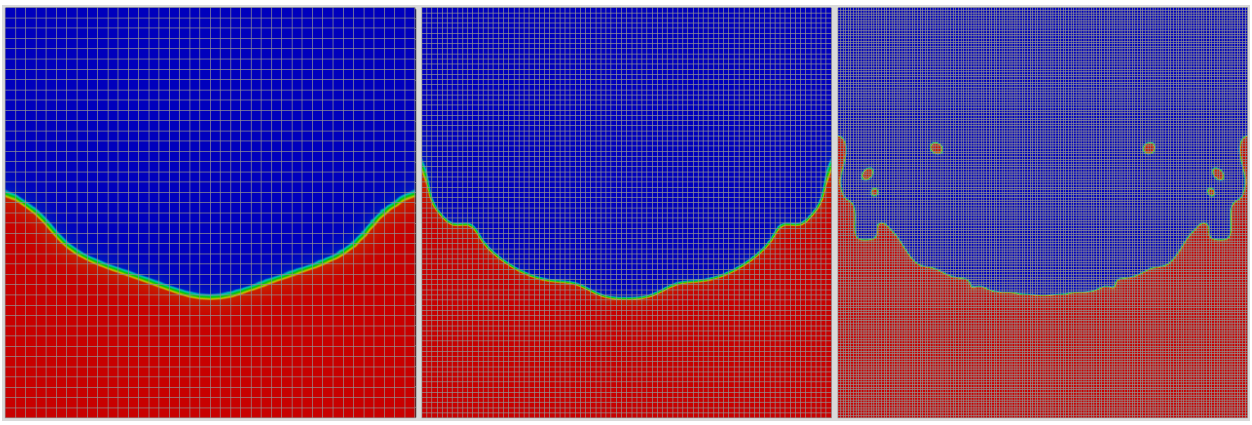


Figure 6.30: Mesh refinement sensitivity analysis tested on the 3 different setups: T1 = 40^3, T2 = 80^3 and T3 = 160^3 computing cells, shown here from left to right, respectively. The total time of the simulation is $T = 20$ s and a result illustrated here occurred at $t = 12$ s.

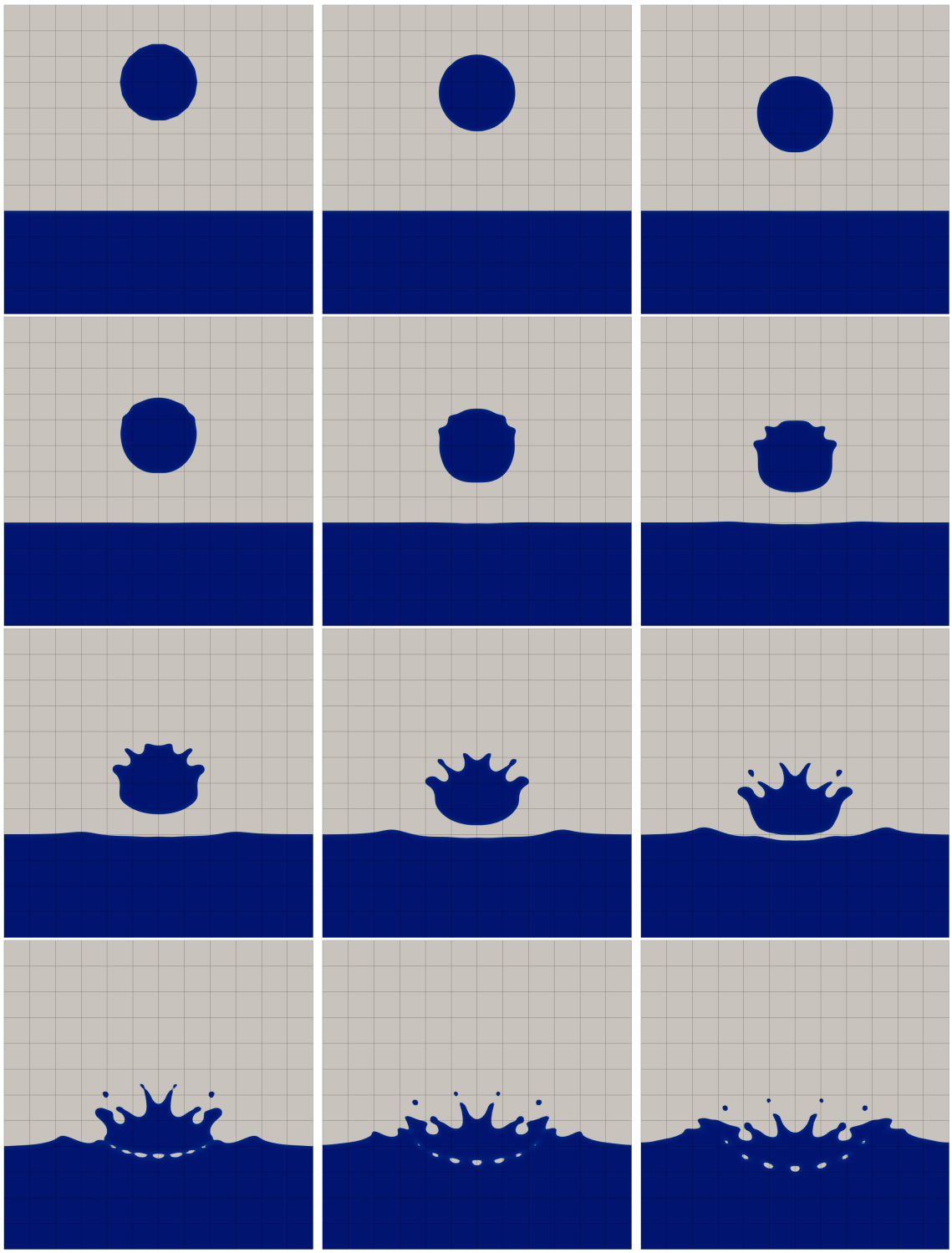


Figure 6.31: Simulation of the water bubble $D_{sp} = 1.5$ m falling into the still water tank of a depth $d = 2$ m. The physical domain has a cubic shape of size $(6\text{m})^3$. Screenshots, ordered from left-to-right in a row-wise manner, illustrate the simulation progress at $t = 0$ s, $t = 2$ s, $t = 3.5$ s, $t = 4$ s, $t = 4.5$ s, $t = 5$ s, $t = 5.5$ s, $t = 6$ s, $t = 6.5$ s, $t = 7$ s, $t = 7.5$ s and $t = 8$ s. A complex separation and merging process can be observed in the last 2 seconds depicted, after which the oscillations and dynamic effects successively diminish.

still water surface, and this can be also observed in Fig. 6.31, at $t = 5$ s, $t = 5.5$ s and $t = 6$ s. At $t = 7$ s, $t = 7.5$ s and $t = 8$ s the separation of the upper droplets happens simultaneously with the capturing of air bubbles into the main body of water, after which propagation waves towards the walls are generated. At this point the wave amplitude drops rapidly, the captured air bubbles will travel toward the interface and rejoin the air phase, and the integral body of water will continue to oscillate, until it reaches the steady-state. The interface remains sharp during the entire time of the simulation.

6.4.3 2D and 3D flows around the 25th April Bridge

After performing the validation of the 2D Saint–Venant Shallow-Water and 3D Navier–Stokes models, this chapter deals with the geometrical requirements of the flow obstacles within one flow scenario, which are sufficient to allow the replacement of the more expensive 3D model with the much faster 2D model, without significant influence on the accuracy of the results obtained. The validations performed are depicted in Secs. 6.2 and 6.4.1, respectively, illustrating some problems where those models can be utilised, as well as pointing out where employment of the simpler model would cause deviations from the solution, leading to a completely different phenomenon, in the worst case. To compare these two models, a restricted geometry representation must be chosen, such that its two-dimensional projection onto the horizontal plane does not cause a significant loss of information. A good example of the geometry required is a multiple-span bridge with quadratic flow fields between the bridge piers and a highly elevated road deck. As published in [19], as long as the water level does not reach the road deck, both the 2D and 3D models can be used, delivering results that are, to a large extent, comparable. Following these statements, within this section, comparison of the 2D and 3D models implemented is performed using an appropriate geometry type, as shown in Fig. 6.32. Further views from the south and east side are given in Fig. 6.33. A two-dimensional projection done onto the horizontal plane is shown in Fig. 6.36, and a detailed visual inspection reveals that all the important geometry information (piers size and position, for instance) have been preserved. The entire domain is 140 m long, 140 m wide and 40 m high, and the lower edge of the road deck is located 14.5 m from the bottom of the valley. The complete geometry description combined with the surrounding terrain can be found in [116].

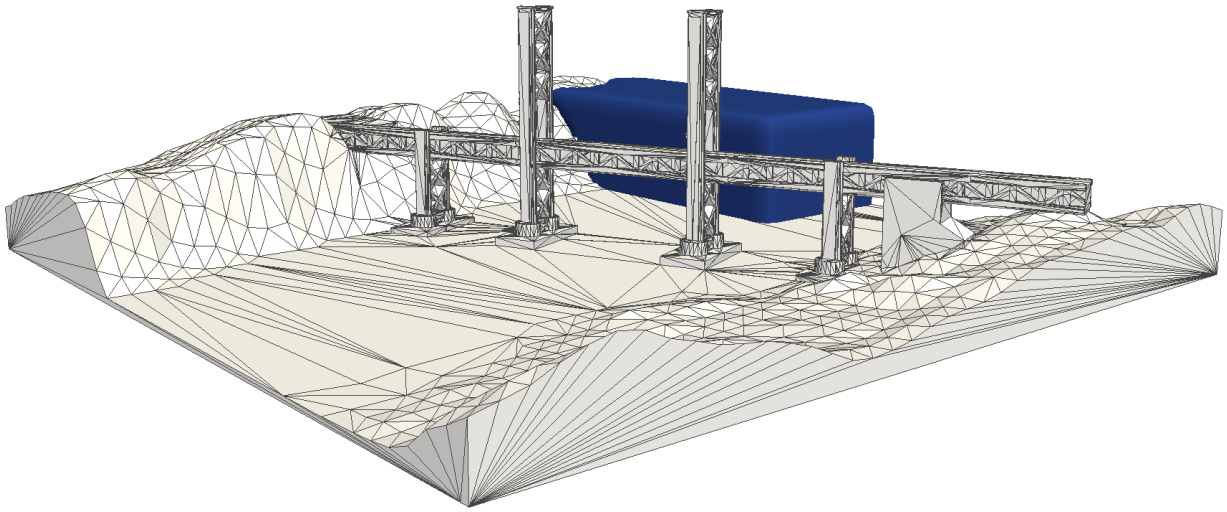


Figure 6.32: Three-dimensional view of the simulation setup 140 × 140 × 40 m large, consisting of complex geometry representation including the bridge construction and the dam-break wave of size 28 × 70 × 25.2 m. The entire STL geometry has a Creative Commons - Attribution Licence and can be obtained at [116]. Side views are given in Fig. 6.33.

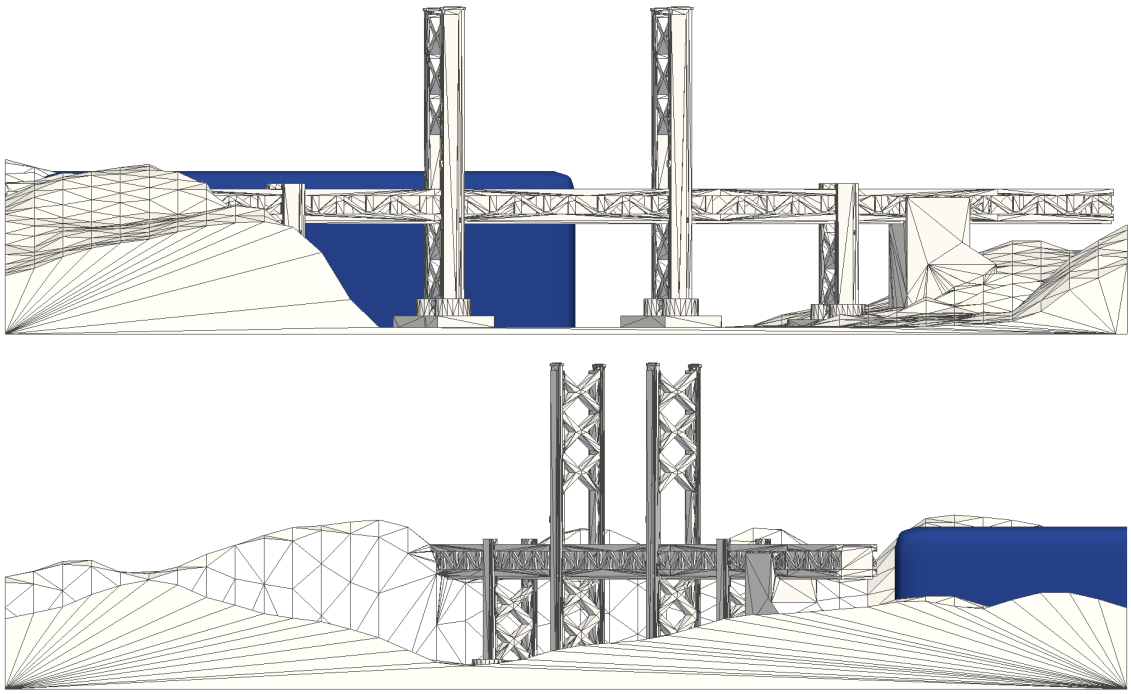


Figure 6.33: Views from the south (upper subplot) and east (bottom subplot) side of the simulation setup, depicted in 3D in Fig. 6.32.

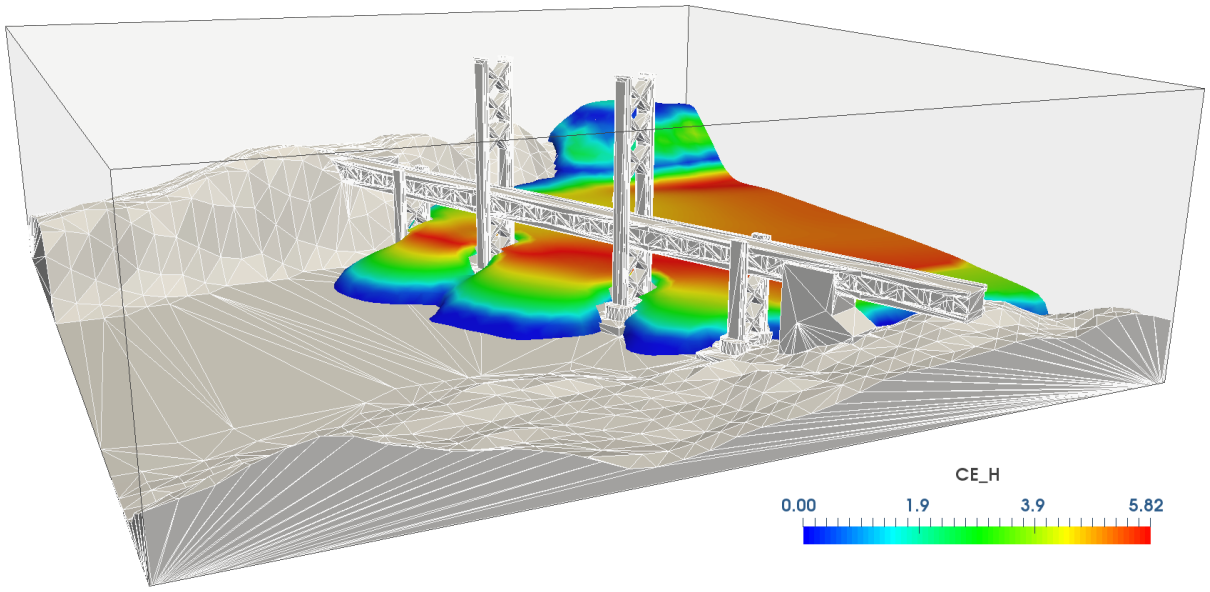


Figure 6.34: Water depth, calculated within the 3D–NSE model, at $t = 8.75$ s. The dam initially breaks into both horizontal directions, reaching the opposite side of the valley, reflecting and traveling towards the bridge openings. Under the bridge, the water level rises locally, due to the complex interaction between the water and solid parts, after which the fluid goes through and unifies with the rest of the water body.

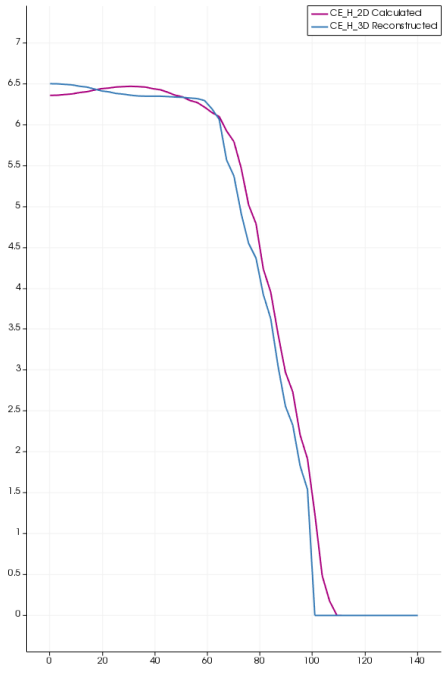


Figure 6.37: Comparison of two water depths, obtained in 2D-SWE model (magenta solid line) and 3D-NSE model (blue solid line).

To simplify the simulation setup, all the outside borders are assumed to be solid walls, with the dam-break wave 28 m long, 70 m wide and 25.2 m high. The uniform structured mesh is chosen with a grid size $\Delta h = 0.85$ m and an initial time-step $\Delta t = 0.01$ s. The total simulation time is set to $T = 50$ s. As depicted in Fig. 6.34, after reaching the opposite side of valley, the flow turns back and goes through the bridge openings, causing a slight local increase in the water level, due to the complex interaction between the water and the solid construction. After the flow passes the bridge piers, the two separate parts of the water merge again into one mass of water, and continue as such. Within the chosen scenario, the road deck cannot be reached, thus both models can be used interchangeably. Nonetheless, it must not be forgotten that the 2D shallow water model

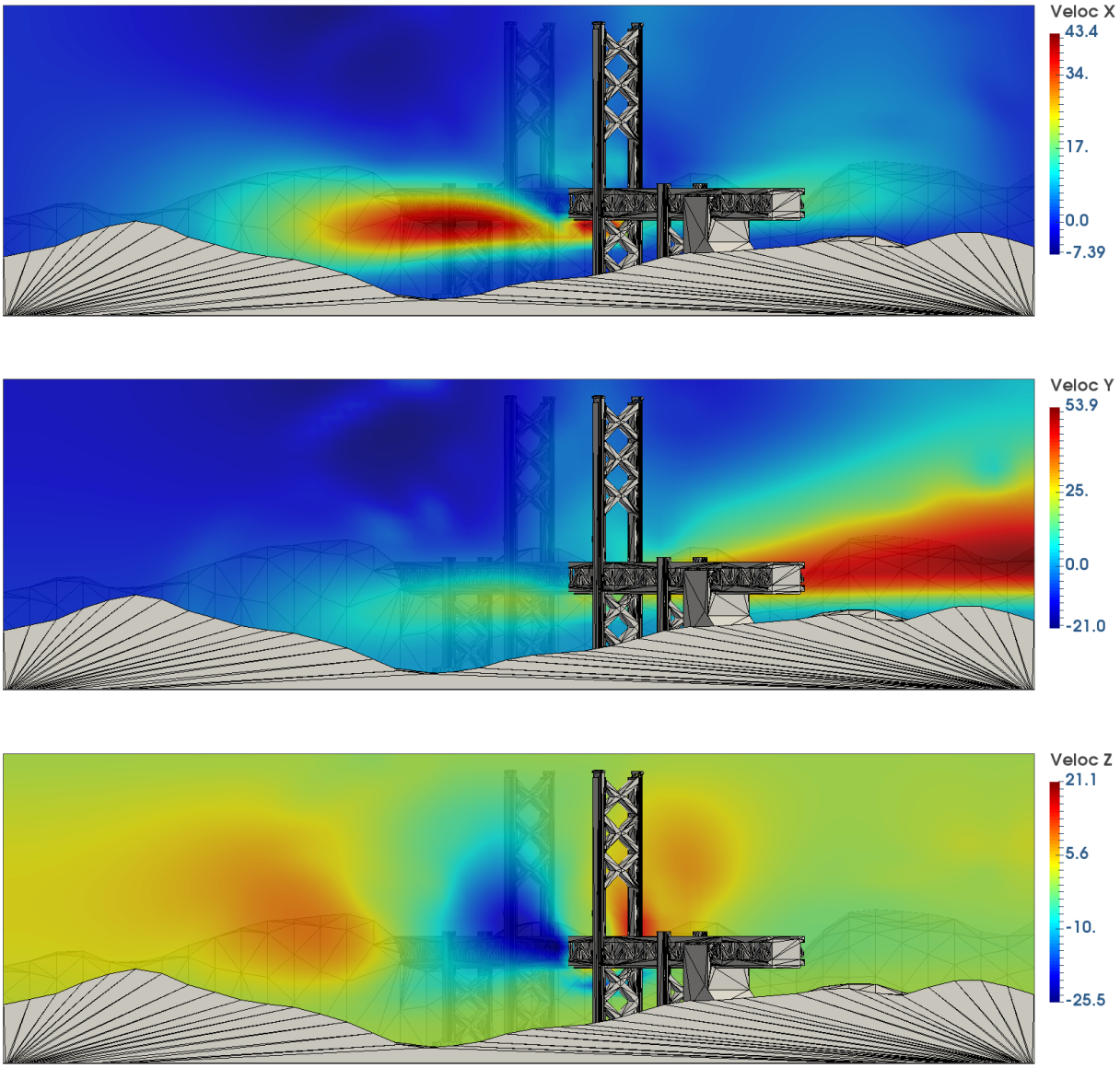


Figure 6.35: Depiction of the calculated velocity vector components in all three Cartesian coordinates, in the 3D–NSE model, in the plane y = 75 m. Having in mind the water level in this particular scenario, it is obvious that these large values happen in the air-phase, due to the high-density ratio. This results in the drastic decrease of the CFL-driven time-step size. The capability of the 3D solver is regardless of the time-step size clearly demonstrated.

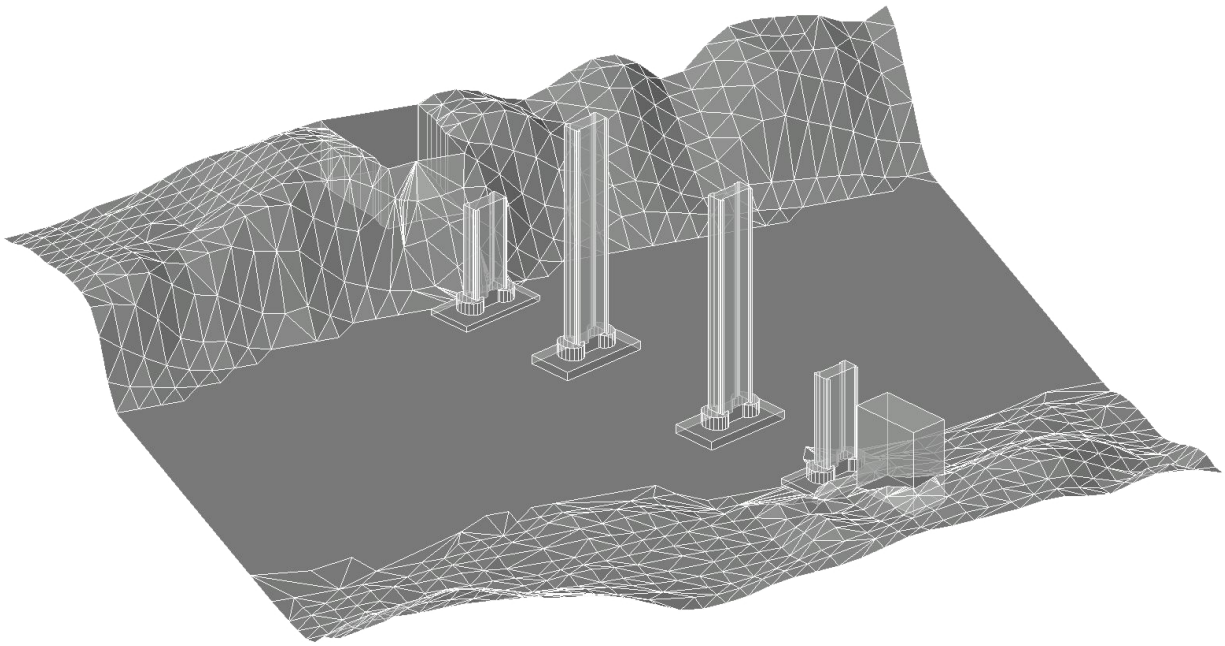


Figure 6.36: 2D projection of the 3D geometry representation, depicted in Fig. 6.32. All exterior dimensions remain the same, as is the case with the size and location of the bridge piers. Carefully comparing the original geometry and geometry generated, it is obvious that all the geometrical details, necessary for the 2D simulation are kept unchanged.

is essentially a single-phase model that takes the velocity values from the water phase in order to calculate the CFL-driven time-step size. On the other hand, the 3D two-phase flow has to take the velocities in both the air and water phase into consideration, and this will significantly reduce the size of the time-step. To illustrate the severity of this issue, the three components of the velocity vector calculated in the 3D model are shown in Fig. 6.35, at $t = 11.25$ s in the cross-section parallel to the x-axis (y = 75 m). While the velocity in the water phase rises to 6–7 m/s, the velocity of the air phase reaches a value almost 10 times as large (see y-component, Fig. 6.35). Besides the solution of the Poisson equation with variable coefficients (see Sec. 5.3) and high-density gradients, this fact will additionally slow down the entire simulation process, thus the possibility of using a 2D model without notable irregularities will be highly beneficial. In Fig. 6.38 three different time instances: $t = 3$ s, $t = 6.25$ s and $t = 10.5$ s of two different models are compared, confirming that the flow dynamics will be quite similar, unless the road deck is reached or overflowed. Additionally, the water levels of both models at $t = 15$ s are compared and depicted in Fig. 6.37, where the solid blue line represents the water level calculated in the 3D model and solid magenta line the corresponding results in the 2D model. It can be seen, despite the difference in the models' complexity, that the concurrence between two water levels measured in the x-axis parallel plane ($y = 45$ m) is very good. Regardless of all the benefits listed, if the dynamic pressure value or velocity distribution over the height plays an important role in the assessment process, there is no workaround to avoid the 3D model can be done. One such complicated scenario is presented in the next section.

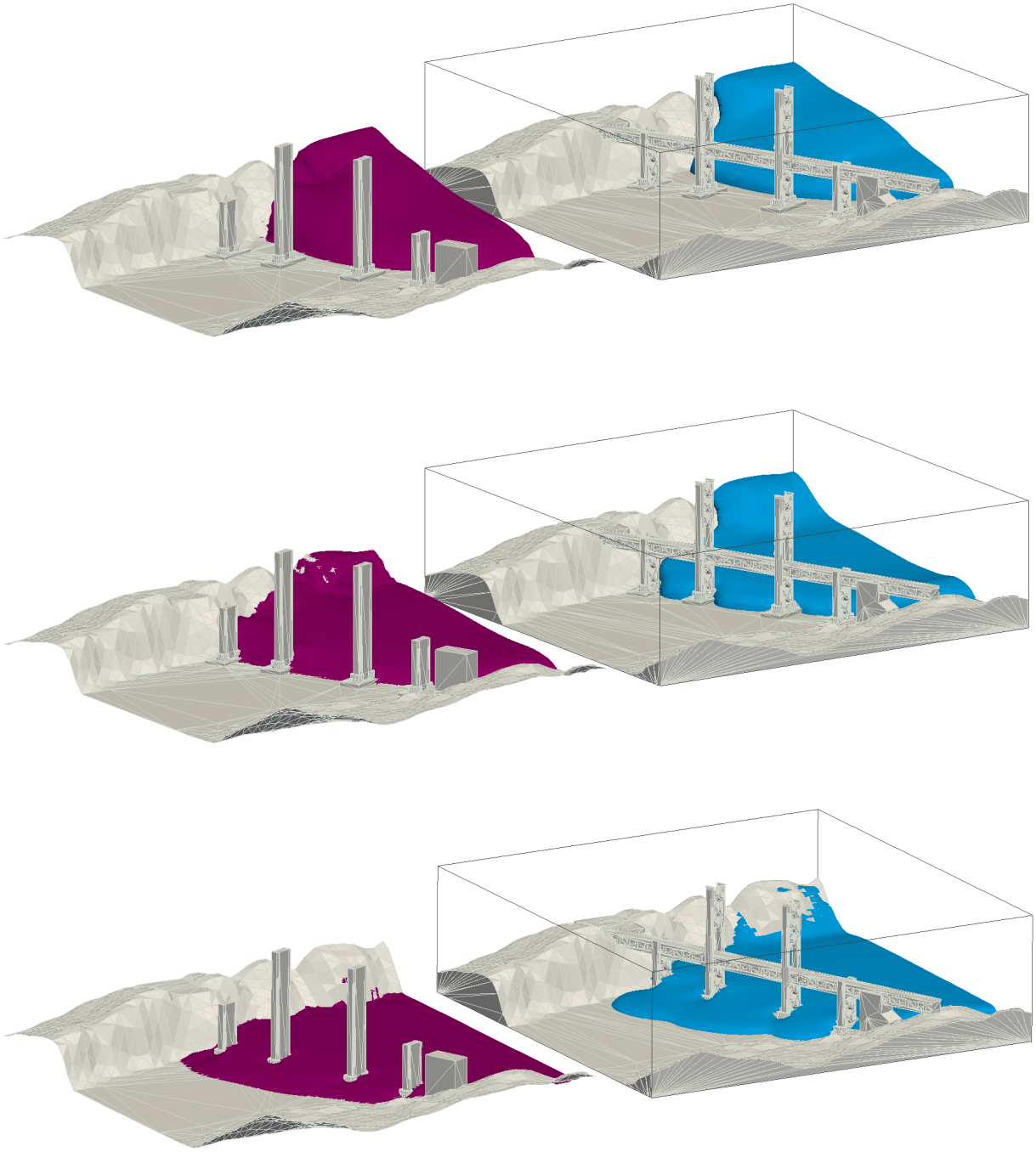
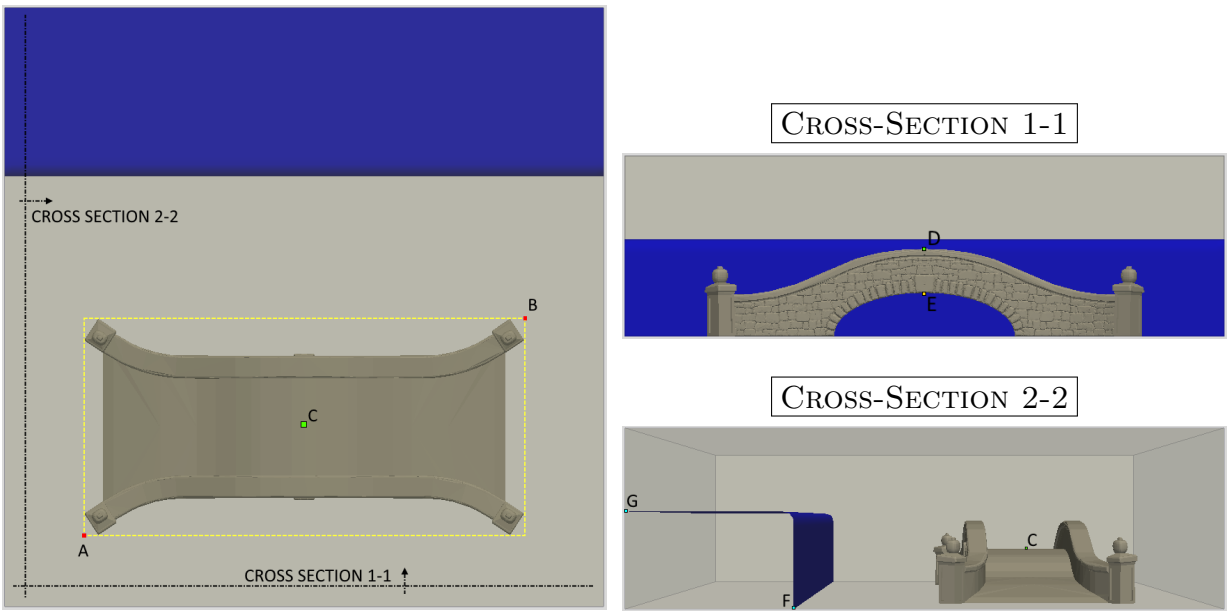


Figure 6.38: Comparison between the 2D–SWE and 3D–NSE models, at three different time instances, $t = 3$ s, $t = 6.25$ s and $t = 10.5$ s. A good concordance between the two water fronts is achieved, which is additionally confirmed in the direct comparison of the water depths, depicted in Fig. 6.37 at $t = 15$ s.

6.4.4 3D flow around a single-span stone bridge

Due to the large scale of river management problems, the vast majority of phenomena in this field can be simulated with the 2D Shallow Water model (see Secs. 3.4 and 6.2), assuming that there is no significant change to any parameter in the vertical z-direction, thus integration over the depth can be done without much loss of accuracy. These principles can be applied to far-field simulations and complex geometric obstacles, as long as these obstacles, averaged over the depth (i.e. projected onto the xy plane), retain their original shape. This excludes every hollow geometry that is to be overflowed in the submerged regime, as well as all flow regimes where flow separation happens. A detailed comparison between one 2D and one 3D model, run on the same geometric profile, where the bridge construction is positioned relatively high to the simulated flow, is given in Sec. 6.4.3. In contrast to the example where the 2D solver demonstrates high efficiency, in this section a geometric profile, which must be simulated in a three-dimensional setup as a partially submerged hollow geometry, will be analysed.



	X	Y	Z
A	13.3	12.0	0.0
B	86.7	48.0	0.0
C	50.0	30.0	8.0
D	50.0	/	14.5

	X	Y	Z
E	50.0	30.0	7.0
F	0.0	75.0	0.0
G	0.0	100.0	15.0

Figure 6.39: The top and side views of the simulation setup: 3D flow around a single-span stone bridge, with the reference coordinates denoted and listed underneath in the tabular view.

In order to create a submerged effect, a single-span stone bridge with a six-metre-high opening is flooded by a 15-metre-high dam-break wave. This does not match any realistic scenario, but the goal of this simulation is to analyse an extreme event where all three types of flows – flow around an obstacle, overflowing of the bridge construction, and flow through a hollow area – may occur at the same time. The numerical domain is of size 100 m × 100 m × 30 m,

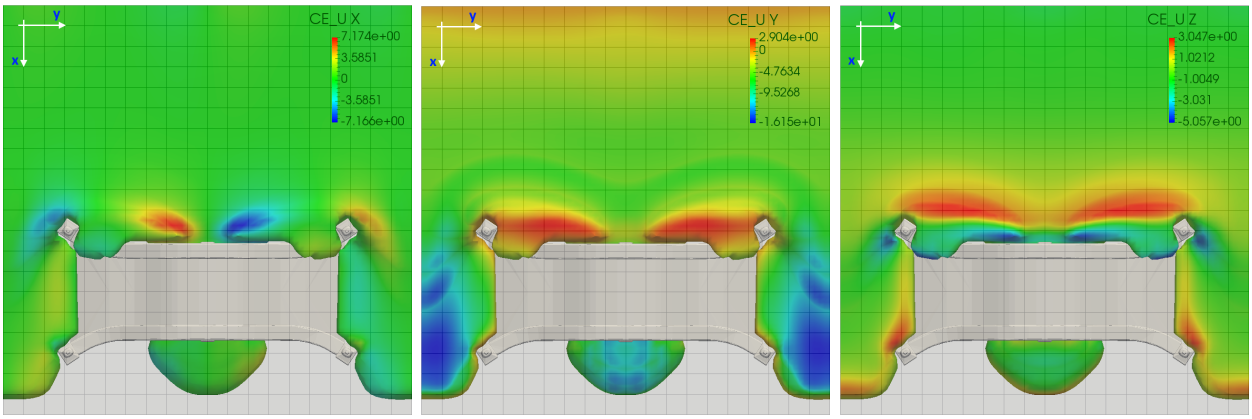


Figure 6.40: Velocity profiles in all three orthogonal coordinate directions, shown from left to right and observed from the top (no slicing performed) - the velocity values shown are captured at the water-air surface. The backward flow is predominantly visible in the middle plot, as the main flow, before reaching the obstacle, occurs in the y-direction. The flows around, over and under the geometric obstacle can also be clearly identified.

with the spatial discretisation $\Delta x = 1.25$ m and the temporal step determined according to the requirements listed in Sec. 4.1.7. All the geometric details of the simulation setup are shown graphically in Fig. 6.39. In order to keep the dynamic effects as long as possible, all the external boundaries are defined as solid walls, preventing the water from leaving the domain. The total time of the simulation is $T_{total} = 35$ s, but it can be seen in Fig. 6.41 that already after 12 s (last subplot), no significant dynamic phenomena can be observed. At $t = 7$ s all three flow types mentioned above can be identified. The overflowing of the bridge construction is linked to the flow separation at the bridge parapet, resulting in one small part flowing over the bridge and merging with the rest of the body of water, while the main part will hit the stone parapet and turn to the opposite direction (backwards-moving flow). This effect is illustrated in Fig. 6.40.

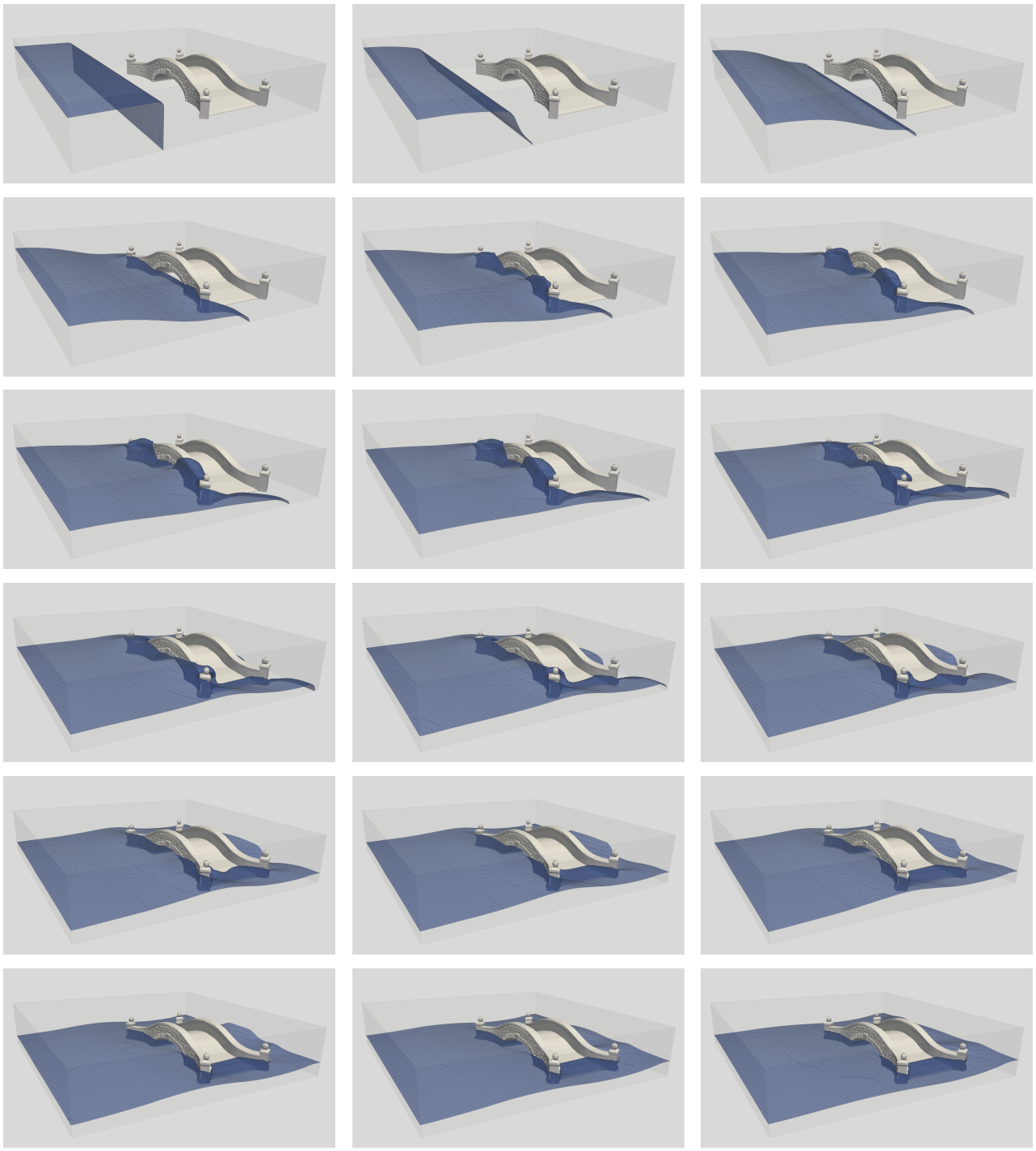


Figure 6.41: Simulation of the three-dimensional dam-break problem in the combination with the complex geometric obstacle. Three different types of motion can be seen here: A) flow around the obstacle, B) flow under the obstacle, and C) flow over the obstacle (overflow). The screenshots illustrate the simulation progress at $t = 0$ s, $t = 1$ s, $t = 2$ s, $t = 3$ s, $t = 3.5$ s, $t = 4$ s, $t = 4.5$ s, $t = 5$ s, $t = 6$ s, $t = 6.5$ s, $t = 7$ s, $t = 7.5$ s, $t = 8$ s, $t = 8.5$ s, $t = 9$ s, $t = 10$ s, $t = 11$ s and $t = 12$ s, from left-to-right in a row-wise manner. The Stone bridge STL geometry has a Creative Commons - Attribution Licence (free-to-use and adapt), and is obtained from [97].

6.5 Coupled 2D shallow water and 3D Navier–Stokes models

The bidirectional coupling between the 2D shallow water and 3D Navier–Stokes models is described in detail in Sec. 4.2.3. In this chapter, two additional tests will be conducted before the real-world case of a flow under a complex bridge structure is performed. The two tests executed reveal the capability of the coupled model to transfer a shockwave over the coupling interface in both directions and the influence of the coupling border on the transient flow setup.

6.5.1 Test No. 1: Shockwave transfer over the coupling interface

Taking into account all the parameters that have to be exchanged at the coupling interface between the 2D-SWE and 3D-NSE models (see details in Sec. 4.2.3), good matching between the elevation surface E_{2D} and the contour level-set surface with a value $LS = 0.5$ is to be expected. In this section only the passing of the shockwave through the coupling interface is examined, excluding any potential influence of the complex geometry located in the three-dimensional part of the domain. As illustrated in Fig. 6.42, within the domain of size $L = 1.5$ m, $B = 1.0$ m and $H = 0.25$ m, the dominant part is marked as 2D, leaving a small area of size $\mathrm{L}_{3D} = 0.5$ m, $\mathrm{B}_{3D} = 0.5$ m and $\mathrm{H}_{3D} = 0.25$ m to be initialised as the 3D model. The 3D part is, on purpose, completely surrounded by the 2D part, in order to test the ability of the shockwave to move from the 2D into the 3D region, and vice versa. The initial wave 0.35 m × 0.60 m × 0.25 m in size is located at the north-west boundary of the domain, falling freely towards the 3D region.

At the very beginning, the 3D region remains empty (see Fig. 6.42, top-left subplot), waiting for the wave to enter. Only afterwards is clear interaction between the 2D surface (transparent

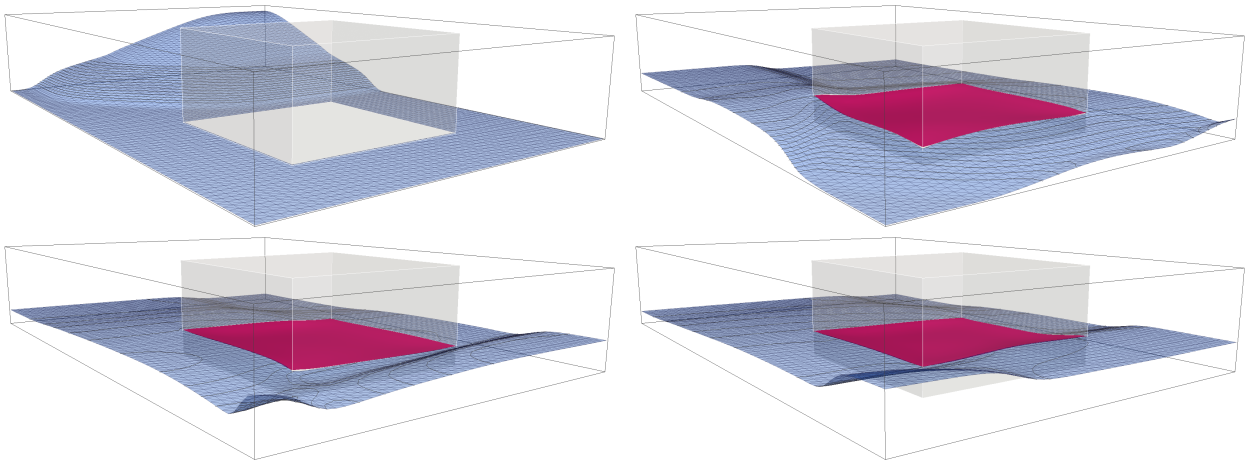


Figure 6.42: Four stages of the shock-wave transport over the coupling interface between the 2D shallow water model and 3D Navier–Stokes model. Top-left plot shows a transversal traveling of the wave before it enters the 3D region. Top-right plot depicts a local rise of the level-set surface (solid magenta) due to the arrival of the wave, bottom-left plot illustrates the wave passing through the 3D region and moving from 3D to 2D region, and bottom-right plot represents a reflection at the wall boundary, causing the wave to travel back.

blue surface) and the 3D surface (solid magenta surface) obvious. The wave travels transversely towards the longer, opposite side of the domain, reflecting and heading towards the 3D region. For this reason, the wave hits the corner of the 3D region, causing the level-set surface to rise locally (see Fig. 6.42, top-right plot). Soon afterwards, the wave travels through and leaves the 3D region on the other side (Fig. 6.42, bottom-left plot), reaching the solid boundary of the domain, and then travels back, causing another corner of the 3D sub-domain to adapt (Fig. 6.42, bottom-right plot).

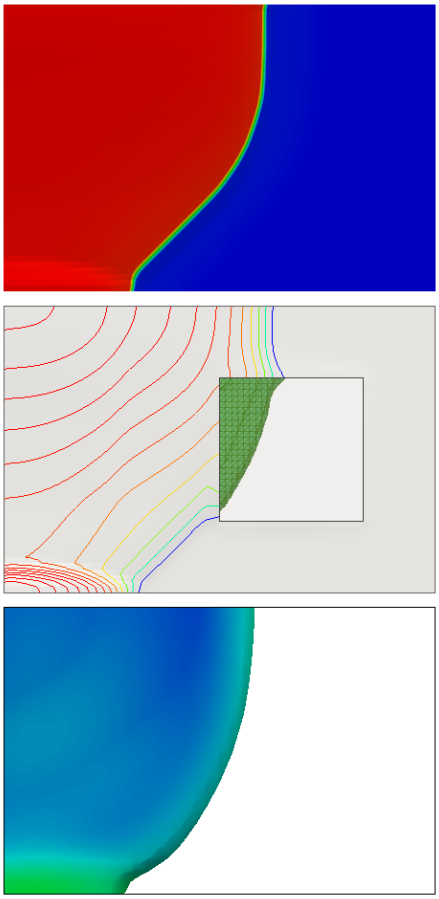


Figure 6.43: Comparison of the wave travelling within the 2D–SWE model, 2D–3D coupled and 3D–NSE models.

During all four stages good agreement in both Cartesian directions can be observed, leaving no place for the accumulation of mass around the artificially introduced coupling interface.
The physics of the problem solved is kept intact, no matter what model is employed. As shown in the left of Fig. 6.43, the wave can be located at the almost same position within the 2D–SWE model (uppermost plot), the coupled 2DSWE–3DNSE model (middle plot) and the 3D–NSE model (bottommost plot). Slight differences observed in the curvature in the coupled model originate from the level-set reinitialisation process, where the thickness of the level-set value, reconstructed from the depth H in the 2D region and passed as a boundary value for the 3D region, cannot be set correctly, if the water depth is measured close to zero (e.g. $H \leq 10^{-5}$). In the further stages of the simulation this phenomenon is not present, as seen in Fig. 6.42, leading to a good approximation of both surfaces around the coupling interface.

6.5.2 Test No. 2: Influence of the coupling interface on the transient flow setup

In the previous section the transfer of the shockwave over the interface is examined, with the exception of some more complicated cases, such as transient steady-state flow in one dimension or flow with an additional flow obstacle located in the three-dimensional domain. These two phenomena are to be dealt with in this chapter. As shown in Fig. 6.44, the transient setup is established within a domain of size 150 cm $\times$ 100 cm $\times$ 100 cm, where only the central part,

represented as a quasi-grid, is set to be a 3D domain. The constant inflow boundary condition is applied on the left-hand side with the constant, required elevation value $H_{ELEV} = 50$ cm. The average velocity field is calculated using the method of characteristics (see [66]), which is mathematically formulated as follows:

$$U_{BC} + 2\sqrt{gh_{BC}} = U_I + 2\sqrt{gh_I} \tag{6.1}$$

where U represents the mean velocity vector in the two principal directions, g is the Earth's gravitational acceleration and h is the depth scalar field. The indices BC and I correspond to the boundary ghost-layer cell and the neighbouring cell inside the domain, respectively.

Within the 2D domain, as in the previous test case shown in Sec. 6.5.1, a first-order Euler explicit scheme is used, whereas in the 3D domain first- and second-order schemes are used interchangeably. Namely, the Navier–Stokes momentum equations are solved using a first-order scheme, combined with the interface advection equation, solved using both first- and second-order schemes. The results depicted in Fig. 6.44 use a second-order accurate interface-advection scheme. The same analysis is done also for the first-order interface advection scheme, resulting in a large deviation of the expected 3D water front, thus this setup is discarded in the further considerations.

Fig. 6.44 reveals a slight local deformation introduced at the sharp corners of the 3D domain, which remains persistent throughout the entire simulation time. As soon as the wave passes the opposite side of the domain, the local deformation is carried out with the flow towards the outflow boundary region (east side of the domain) and rapidly mitigated. In order to illustrate the size of this local disturbance, the whole simulation setup is depicted in full 3D Cartesian representation in Figs. 6.45 and 6.46 (see left-hand side plot).

The second issue examined within this section is an influence of an additional flow obstacle, set in the 3D domain, as shown in Figs. 6.45 and 6.46, on the right-hand side. The same simulation setup, as previously introduced, is used here on purpose, comparing side by side the significant simulation instances, in order to emphasise the disturbance caused by the insertion of a rigid body. The dominant influence in the vertical z-direction is visible especially in the downstream part, behind the obstacle, where the depth calculated within the 3D model is somewhat lower than the depth calculated using the 2D–SWE model alone. A steep transition between the two water depths can be observed at the coupling interface, although the smooth connection is preserved. To prevent this kind of behaviour, a slightly larger 3D domain should be chosen, such that all significant three-dimensional effects are fully resolved before they reach the 2D–3D interface. How large this domain should be is strongly influenced by the phenomena examined; nevertheless, it is good practice to choose three to five lengths of the inserted obstacle in every direction.

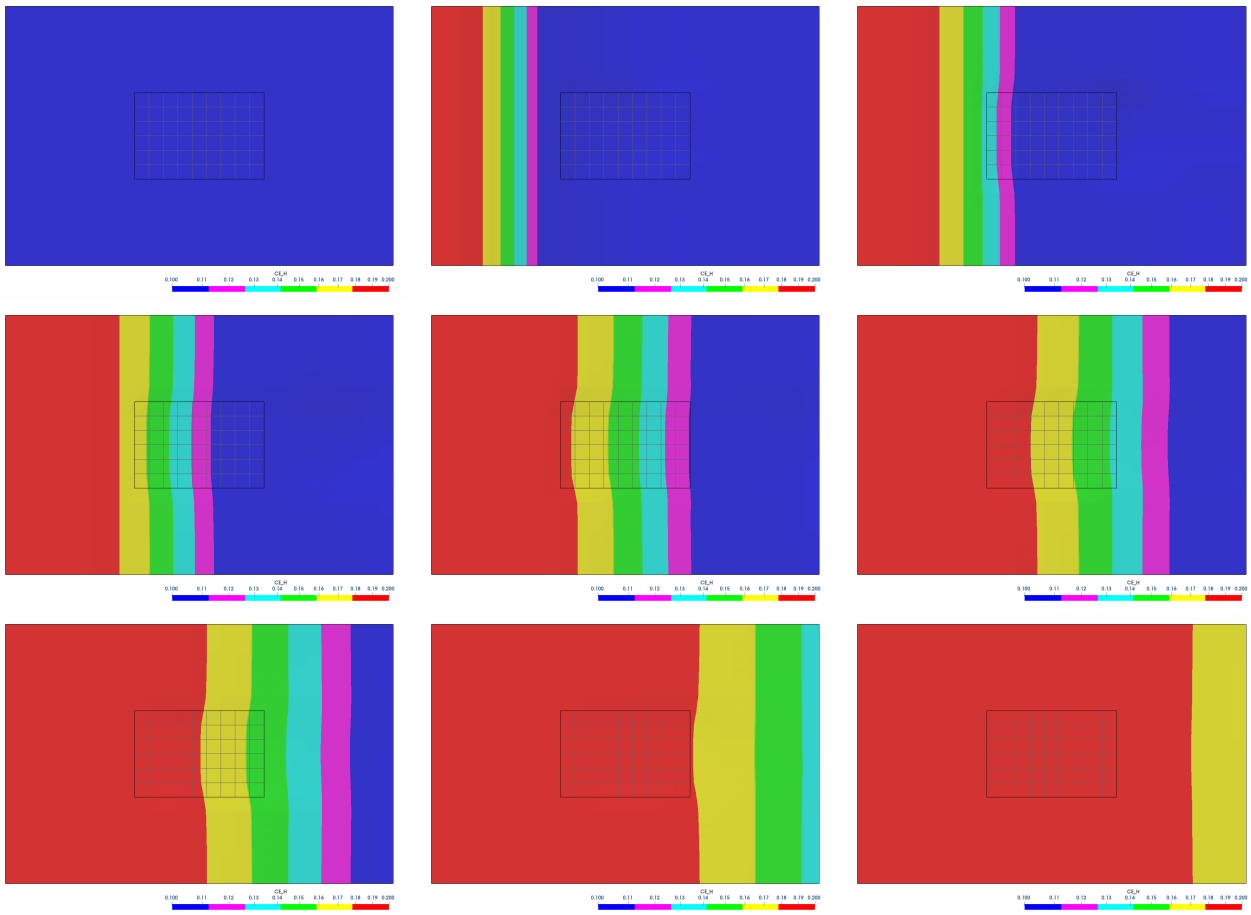


Figure 6.44: Transient flow through the empty 2D–3D domain, entering on the left (west side) and leaving the domain on the right (east side). The simulation sequences are shown in the correct order from left-to-right at $t = 0$ s, $t = 0.2$ s, $t = 0.4$ s, $t = 0.6$ s, $t = 0.8$ s, $t = 1.0$ s, $t = 1.2$ s, $t = 1.4$ s, $t = 1.6$ s. The slight local deformation is triggered at the sharp corners of the 3D domain, nevertheless, it remains constant and diminishes as soon as the wave reaches the opposite side of the domain.

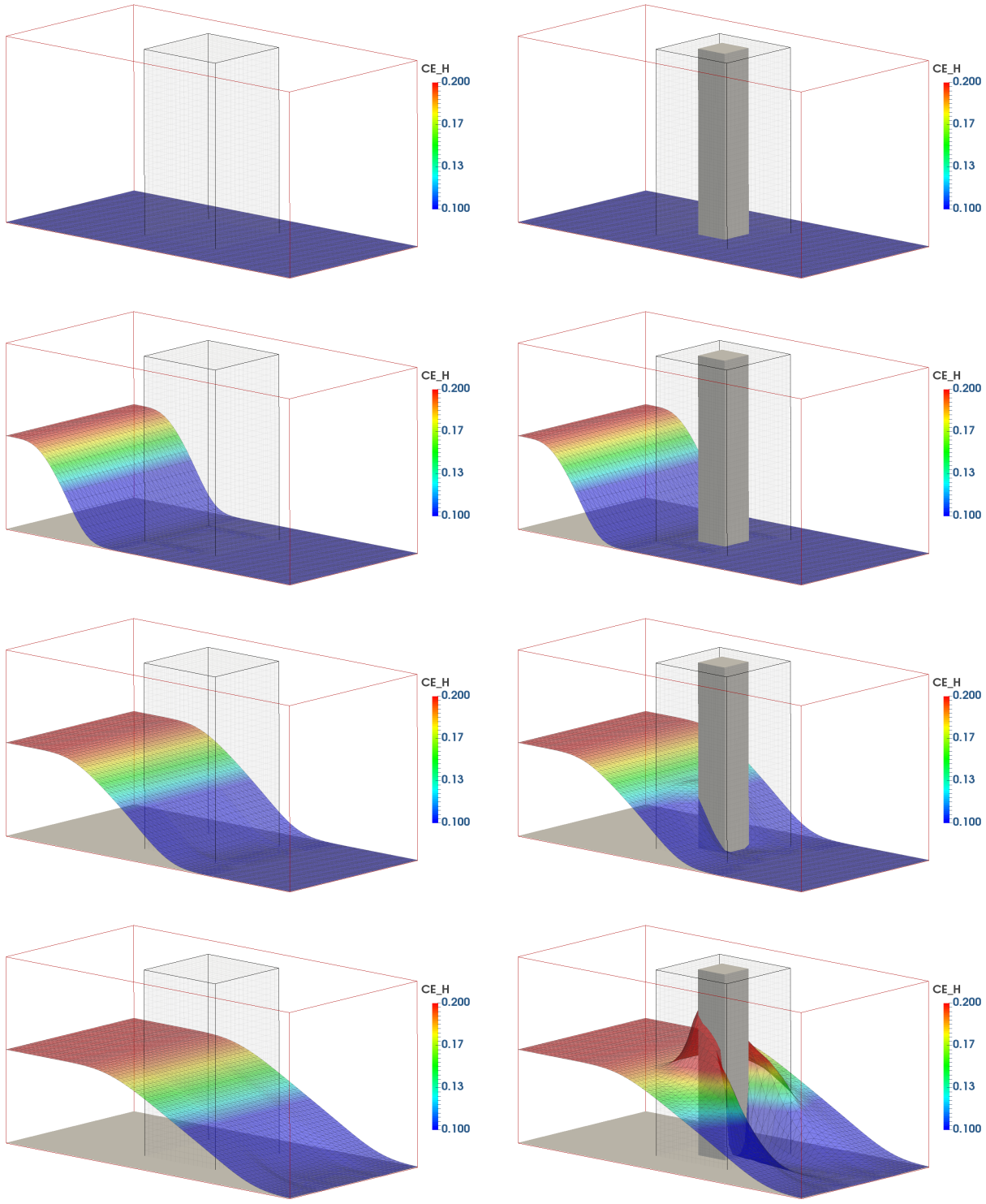


Figure 6.45: Full 3D representation of the simulation setup shown in Fig. 6.44, both with and without an obstacle inserted into the 3D domain. The simulation sequences are captured at $t = 0$ s, $t = 0.2$ s, $t = 0.6$ s, $t = 0.8$ s.

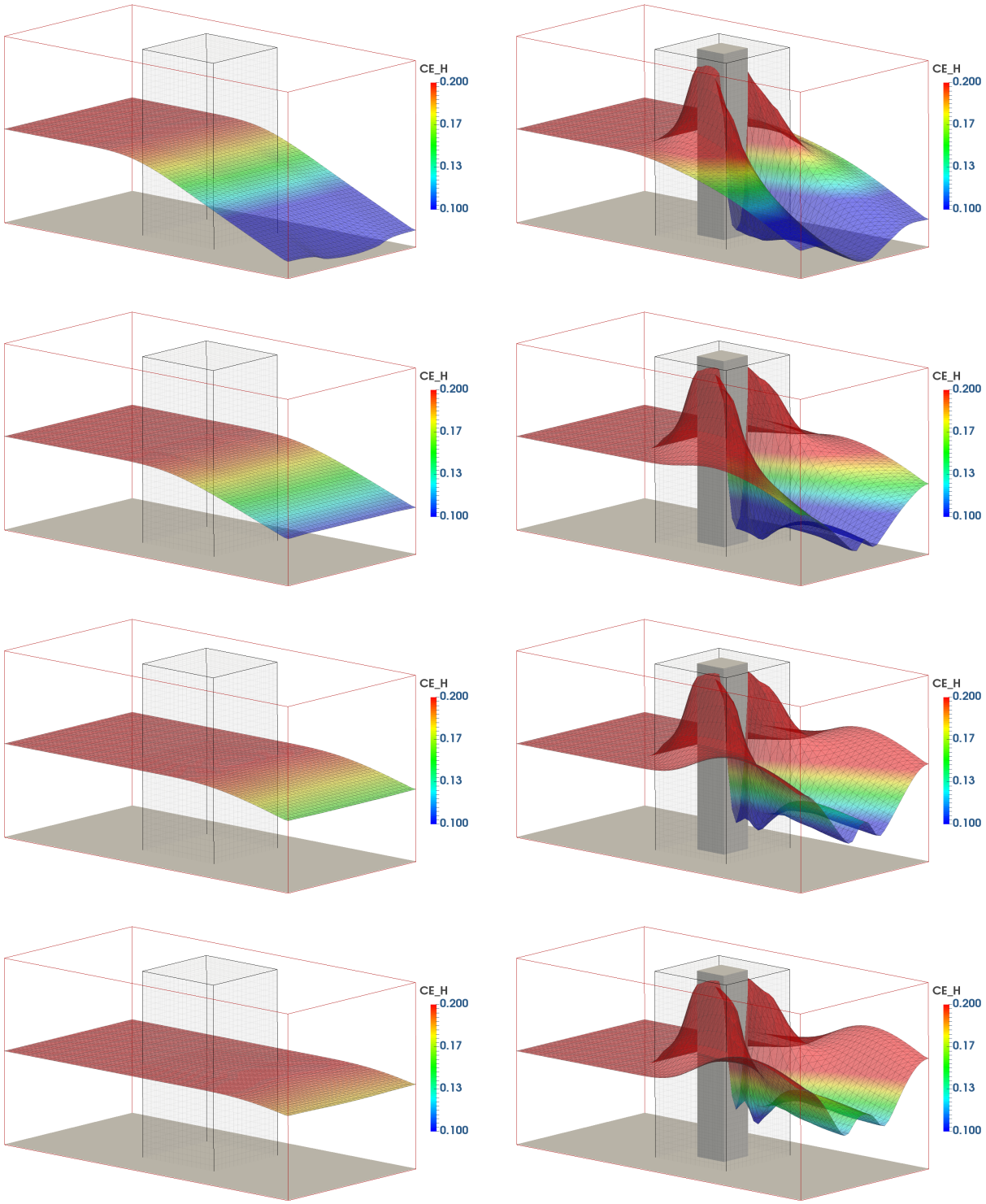


Figure 6.46: Full 3D representation of the simulation setup shown in Fig. 6.44, both with and without an obstacle inserted into the 3D domain. The simulation sequences are captured at $t = 1.0$ s, $t = 1.2$ s, $t = 1.4$ s, $t = 1.6$ s.

6.5.3 2D–3D flow through a hollow obstacle

To illustrate the benefits of coupling of two separate models of different complexities, a physical domain must be chosen, such that in one part of the domain the 2D–SWE model can represent all the flow behaviours correctly, whereas in the rest of the domain the 3D–NSE model must be employed in order to resolve complex flow phenomena, such as simultaneous flow under and over the structure, separation and merging of the fluid entities, etc. One such scenario is depicted in Fig. 6.47, where it can be seen that the flat area far away from the complex hollow obstacle can be calculated using a simple model, thus leaving the sub-domain close to the obstacle to be calculated using the full 3D–NSE model. As stated in Sec. 6.5.2, the 3D sub-domain must be three to five times as large as a characteristic size of the inserted obstacle, in order to be able to represent all the three-dimensional effects in case of the transient flow setup. In this particular case, the space behind the obstacle is slightly larger than the space in front of it, as depicted in Fig. 6.48, leaving just enough space behind for the merging of the waves coming under the obstacle and the water coming through the openings within the obstacle itself. After the merging has been completed, the flow calms down and leaves the 3D sub-domain.

The complexity of the geometry chosen is important, as it has been proven in Sec. 6.4.3 that any less complicated geometry can be successfully approximated with the 2D–SWE model, leading to the sufficiently accurate results for general purpose analysis. For that reason, a hollow obstacle has been chosen here, with 6 openings of different sizes and elevations, which allowed a precise control of amount of water redirected through each of them. The geometrical characteristics of the obstacle can be observed in the side views, shown in Fig. 6.49.
The entire domain is 240 cm long and 60 cm wide. It consists of two 2D regions, separated with one central 3D region. the first, upstream 2D region is 60 cm long, as illustrated in Fig. 6.48; the second, downstream 2D region is 90 cm long, and the central 3D part is of size 90 cm $\times$ 60

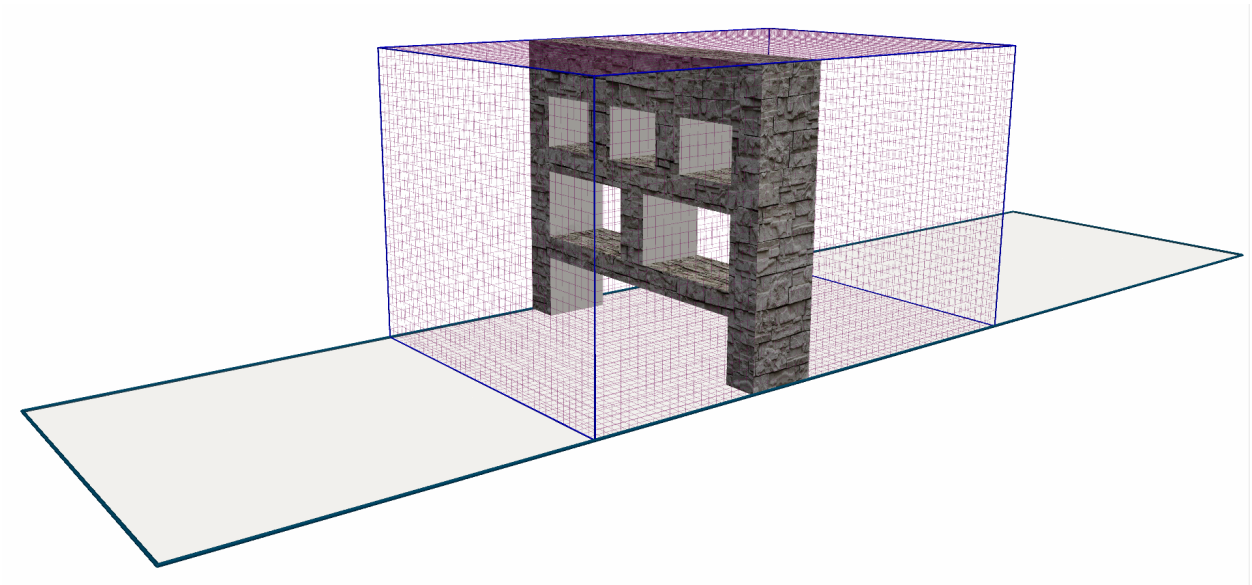


Figure 6.47: 3D view of the stone obstacle, set in a 2D–3D domain. The upstream 2D region is 60 cm long and both the central 3D domain and the downstream 2D region are each 90 cm long. The width and height of the domain are $B = 60$ cm and $H = 50$ cm, respectively.

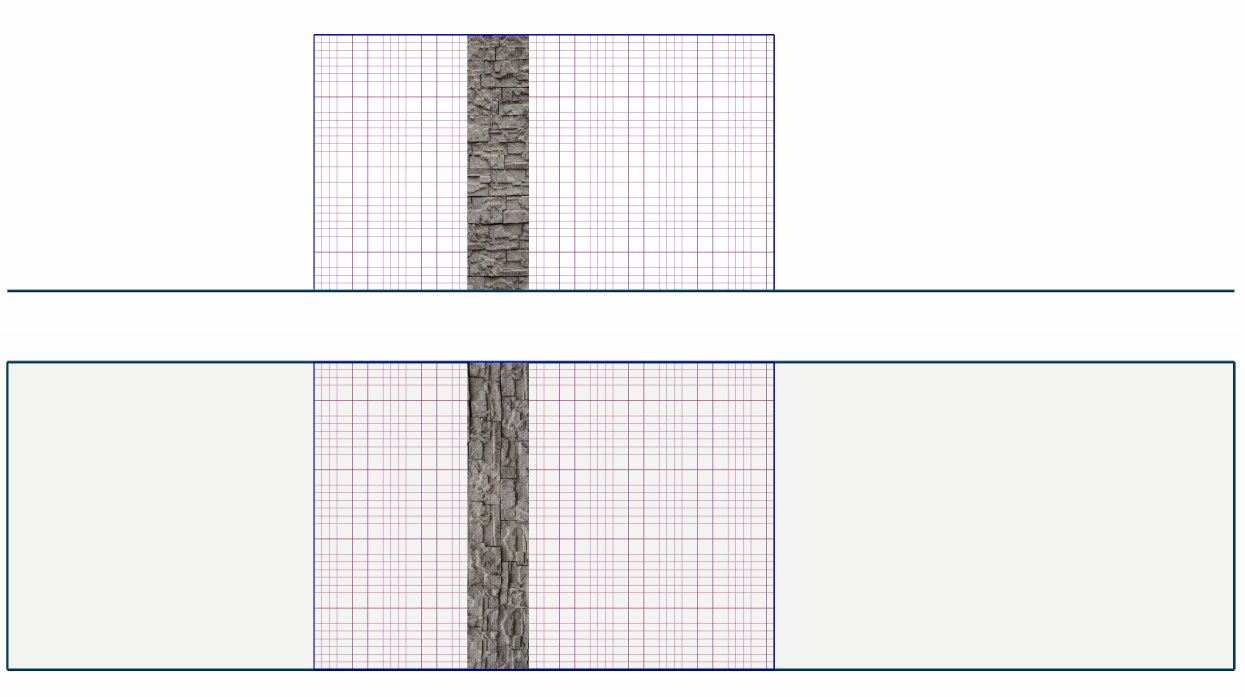


Figure 6.48: Side and top views of the 2D–3D numerical domain, with a stone obstacle positioned 30 cm away from the upstream 2D–3D interface.

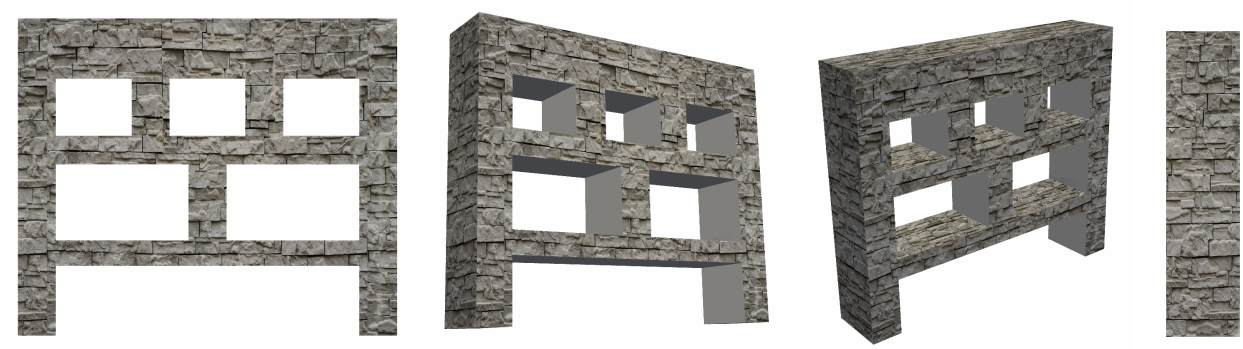


Figure 6.49: The front and side views of a hollow obstacle inserted in the 3D region. The total width and height of the obstacle are $B_{ob} = 60$ cm and $H_{ob} = 50$ cm, respectively.

cm $\times$ 50 cm. The height of the 3D region is chosen based on the total height of the obstacle $H_{ob} = 50$ cm, in order to create a back-flow from the obstacle.

In order to avoid the accumulation of water within the domain, the outflow boundary condition is assigned to the east border, allowing the water to leave the domain without any disruption. All the other boundaries are declared as solid walls which have a solid-slip boundary condition applied. The width of the obstacle is equal to the width of the domain $B_{ob} = 60$ cm, preventing the flow from going around the obstacle, as is the case with the single-span stone bridge shown in Sec. 6.4.4. The dominant flow direction is perpendicular to the obstacle orientation and, if the setup was run long enough, the fluid would escape the domain completely. The total

time of the simulation is set to $T = 10$ s, and the initial time-step size is $\Delta t = 0.001$ s. All later time instances are calculated in accordance with the Courant–Friedrichs–Lewy condition (see Sec. 4.1.7). The computing cells' size must be set to $\Delta x \leq 2$ cm if the behaviour through the hollow parts of the obstacle is to be reproduced in an acceptable way. A resolution larger than that leads to a rather large deviation from the real physical model. This phenomenon is even more obvious if the openings of the obstacle do not conform to the three principal Cartesian directions. Moreover, further grid refinement contributes to a better approximation of the flow when it comes to small-scale phenomena, reducing the influence of the octree mesh representation that is used instead of a geometry-conforming mesh.

As can been seen in Fig. 6.50, at the very beginning ($t = 0$ s) the complete domain is considered to be dry, with a constant-height wave at the left, the west side of the domain $H_{wave} = 110$ cm. Such a large wave comparing to the size of the domain generates a high velocity field in the dominant flow direction, thus the wave reaches the 3D region in less than 0.1 seconds. This kind of behaviour, according to the Reynolds number, can already be seen as turbulent flow; nevertheless, within the MPFluid framework the turbulent flow analysis is implemented only within the single-phase flow module. For that reason, this simulation setup will be treated and observed as a laminar flow occurrence. As shown in Figs. 6.53–6.52, the flow is strictly x-direction dominant while residing in the 2D region, whereas the first 3D effects are to be seen when the water front gets closer to the obstacle. A significant movement in the perpendicular y-direction starts to be obvious at $t = 2$ s, resulting in the sharp redirection of the flow through the lower positioned openings. At the same time, one part of the fluid experiences a large vertical acceleration, creating an upstream back-flow wave, as shown in Fig. 6.50, in the middle plot. As time passes, the upstream water depth starts to rise, gradually filling all the void space with water. Due to the existing of multiple openings within the obstacle, the separation of the flow has been enforced, leading to one portion of the flow going under the obstacle and the second portion flowing through the small openings. Once the flow has crossed the construction, each separate stream has to fall downwards due to the prescribed gravity acceleration, forcing the upper flows to merge with the portion of the flow going under the obstacle, as depicted in Figs. 6.51–6.52. This complex event is accompanied by air pockets captured within the fluid phase, which travel along with the flow for a short period of time, trying to escape into the atmosphere. At the very end, the locally disturbed flow calms down and heads towards the interface between the 3D and 2D regions. As illustrated in Fig. 6.52, middle and bottom figures), as soon as the flow re-enters the 2D domain, fewer dynamics can be observed and the flow is directed towards the eastward boundary with the outflow boundary condition posed.

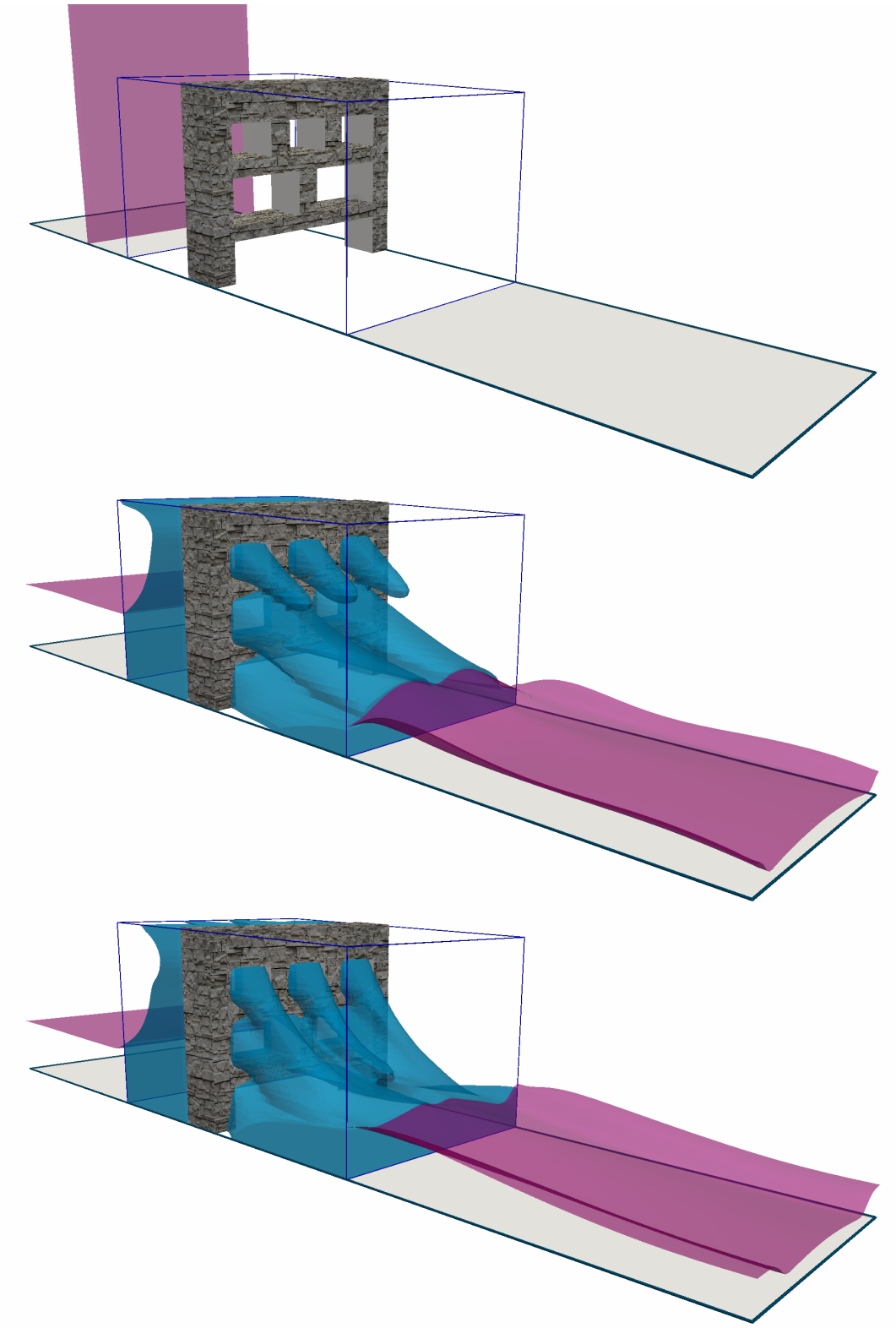


Figure 6.50: Development of the simulation process at time sequence $t = 0.0$ s, $t = 0.475$ s and $t = 0.60$ s, observed from the downstream side. The upstream behaviour of the same simulation is shown in Figs. 6.53–6.55.

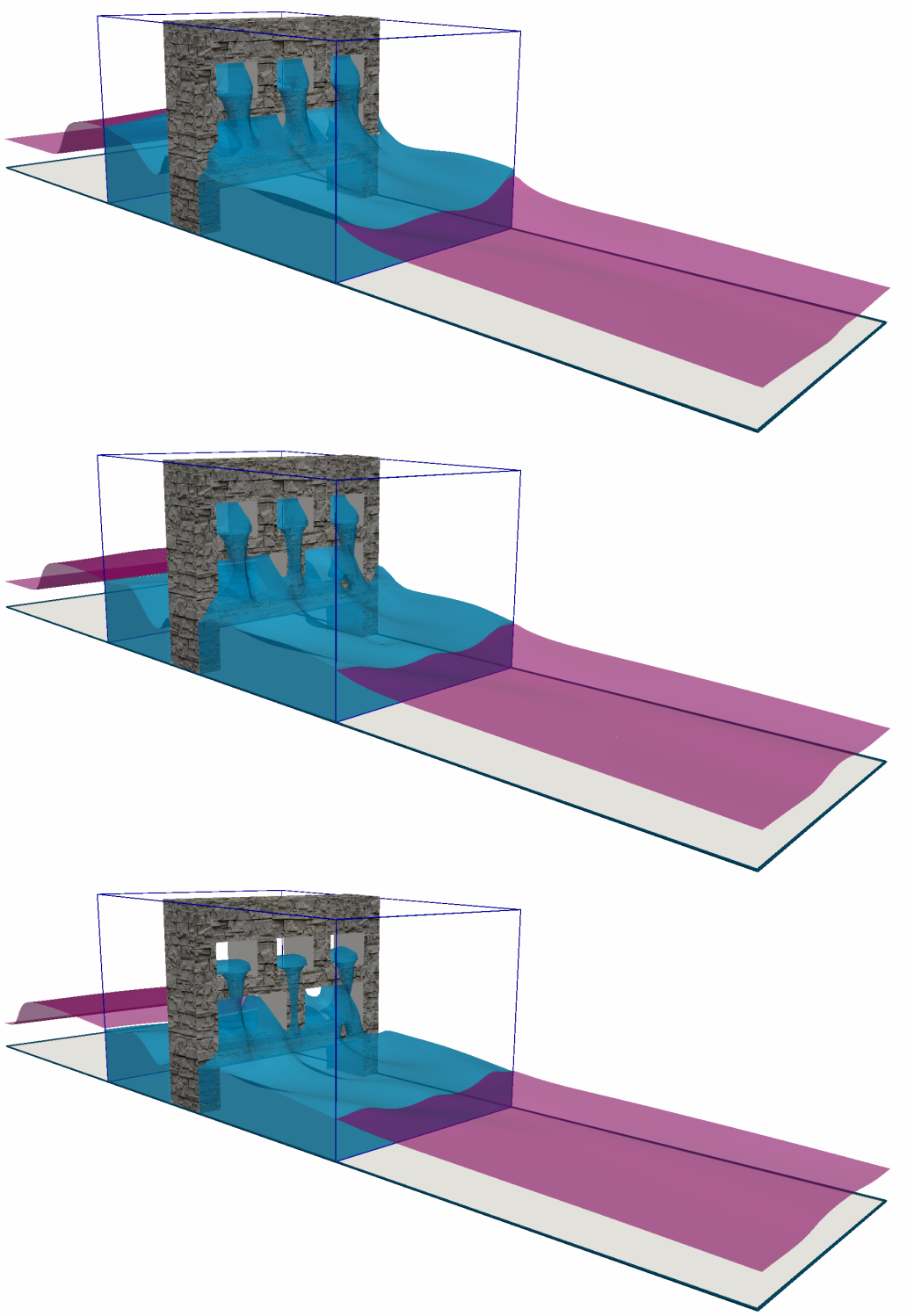


Figure 6.51: Development of the simulation process at time sequence $t = 0.95$ s, $t = 1.05$ s and $t = 1.15$ s, observed from the downstream side. The upstream behaviour of the same simulation is shown in Figs. 6.53–6.55.

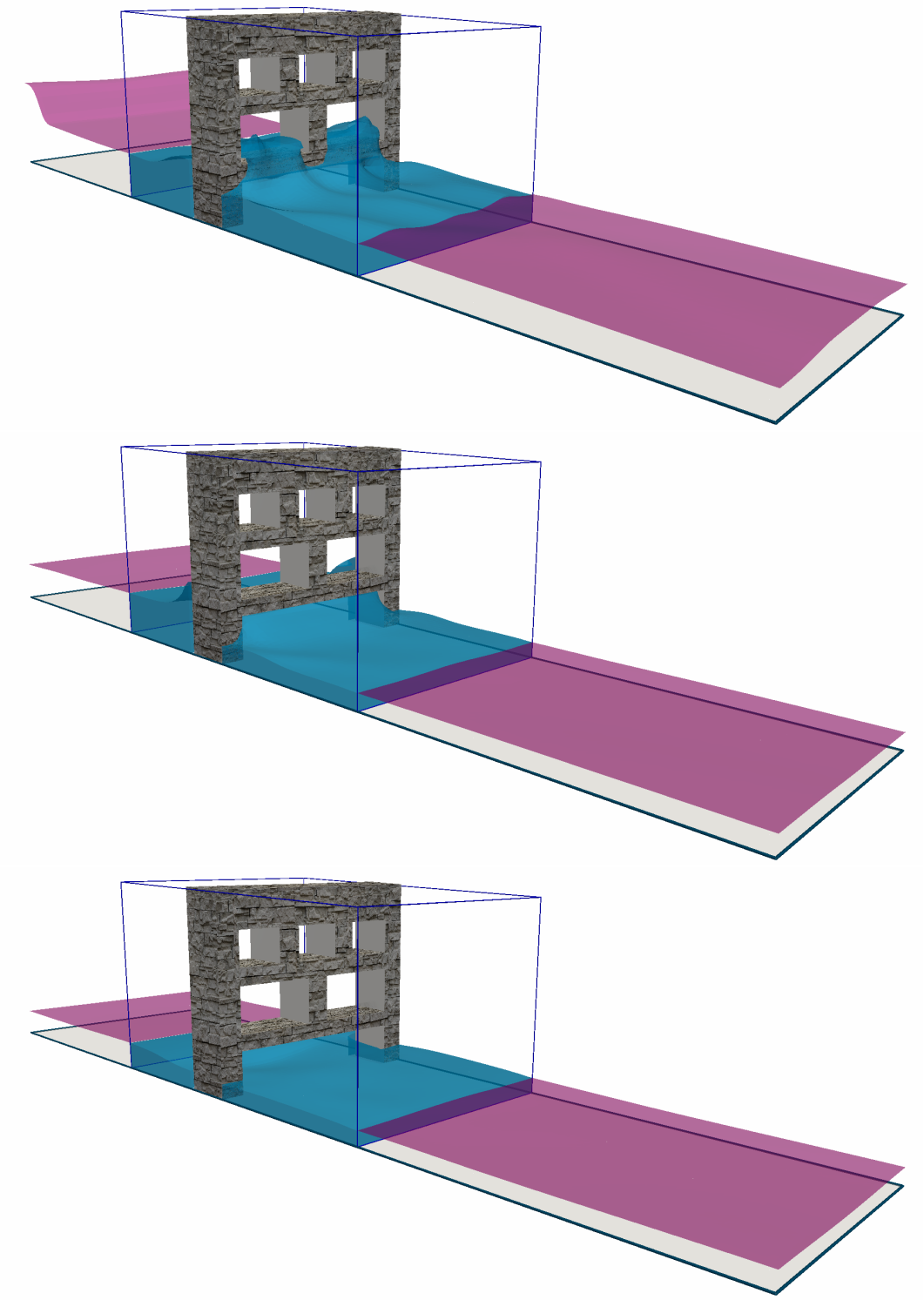


Figure 6.52: Development of the simulation process at time sequence $t = 1.3$, $t = 2.125$ s and $t = 3.5$ s, observed from the downstream side. The upstream behaviour of the same simulation is shown in Figs. 6.53–6.55.

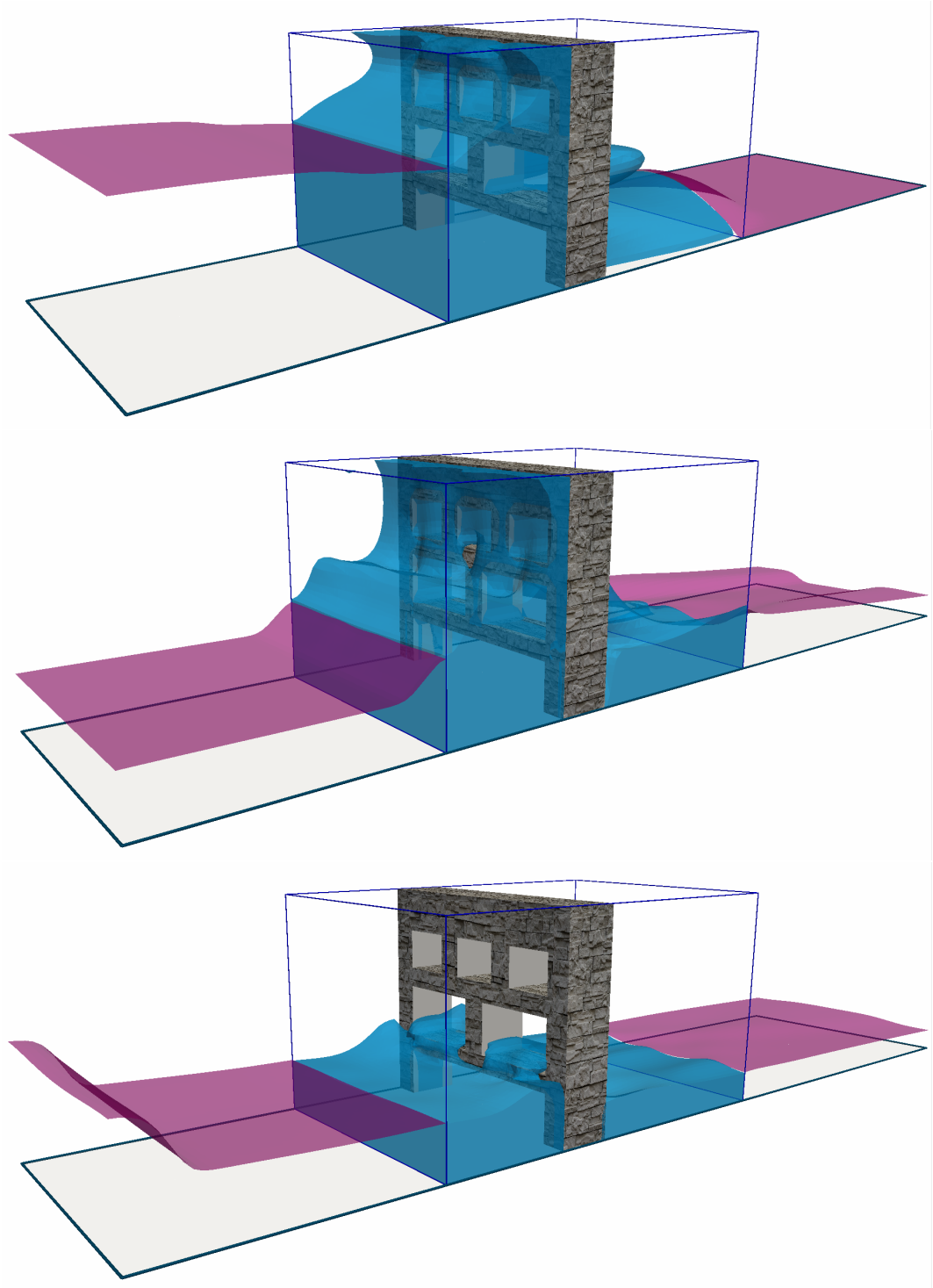


Figure 6.53: Development of the simulation process at time sequence $t = 0.3$ s, $t = 0.75$ s and $t = 1.30$ s, observed from the upstream side. The downstream behaviour of the same simulation is shown in Figs. 6.50–6.52.

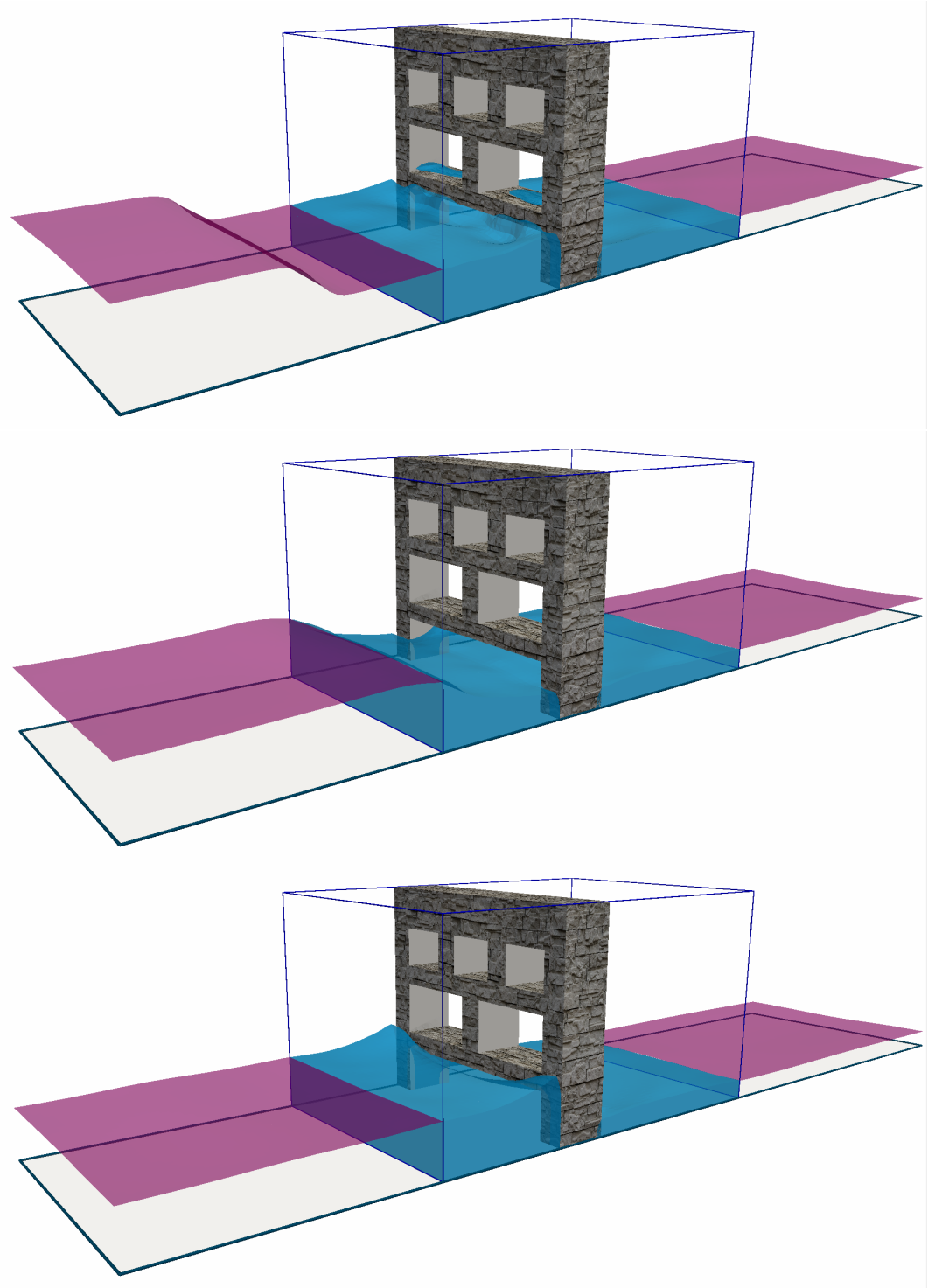


Figure 6.54: Development of the simulation process at time sequence $t = 1.50$ s, $t = 1.75$ s and $t = 2.0$ s, observed from the upstream side. The downstream behaviour of the same simulation is shown in Figs. 6.50–6.52.

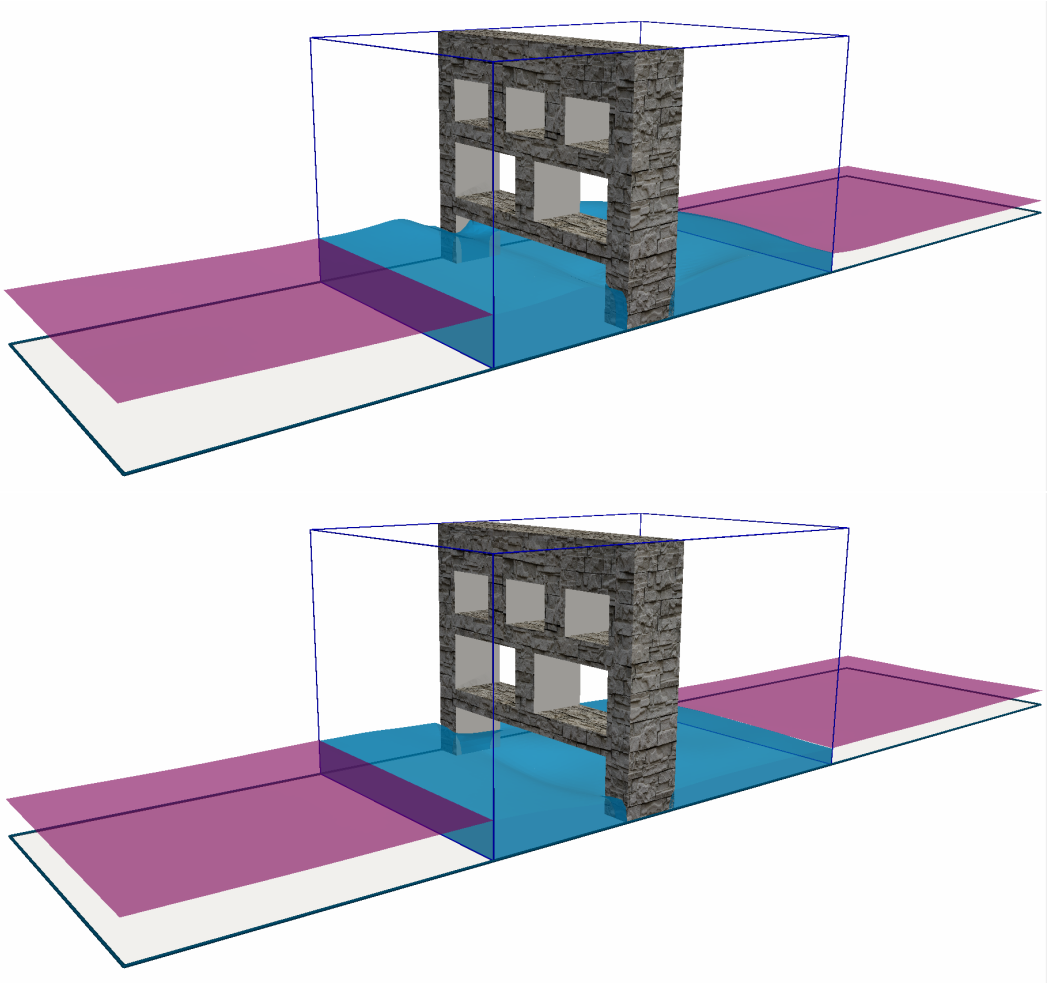


Figure 6.55: Development of the simulation process at time sequence $t = 2.4$ s and $t = 3.5$ s, observed from the upstream side. The downstream behaviour of the same simulation is shown in Figs. 6.50–6.52.

Short résumé of all methods used:

Within this chapter the simplest model used is the 2D–SWE model that is also the least time consuming and capable of dealing with flows on large flat inundation areas without complex geometry representation (Sec. 6.2). The 3D–SWE model is built on the basis of the 2D-SWE model and is moderately expensive due to the necessity to solve a pressure Poisson equation. This model can retrieve the information about the pressure and vertical velocity component U_z, which is not available in 2D–SWE. The distribution of the horizontal velocity components over the height is not constant anymore and the geometry representation is as complex as in the 2D–SWE approach (Sec. 6.3). The most complex model implemented is the 3D–NSE model combined with the level-set interface capturing method. It is fairly expensive; nevertheless, it can deal with a very complex geometry representation and resolve all kinds of different fluid phenomena.(Sec. 6.4) Using a rather simple geometric obstacle, the 2D–SWE is compared with the 3D–NSE (Sec. 6.4.3) and delivers reliable results, thus it is recommended to use the complex model only there, where the application of the 2D–SWE model is not possible, without compromising the quality of the results expected. The 3D–SWE model can be used, if no other, more capable three-dimensional model is available; otherwise, it is proven to be rather expensive (42% less expensive than 3D–NSE on a rather simple geometry representation) with a limited number of benefits comparing to the simple 2D–SWE and a wide range of disadvantages in comparison to the 3D–NSE model.

Chapter 7

Conclusions and further considerations

Rapid development of high-performance computing strategies and their usage in a wide variety of engineering fields, have brought tremendous advancements in the solution of many problems that could not be solved using only traditional analytical methods. Within this work, these strategies have been predominantly utilised in the field of multi-phase fluid flow simulations, applied on five models of different complexities. As described in Sec. 5.1, the framework, generated in [51] and adopted for further development, supports the solution of elliptic partial differential equations that describe single-phase fluid flows, taking advantage of Chorin's projection method and Poisson equations solved on a block-structured grid setup. The code is validated using standard single-phase flow benchmarks, such as the Kármán vortex street, driven cavity and Rayleigh–Bénard convection. Starting from that point, this thesis covers the development of three different multi-phase models: a two-dimensional shallow water model (2D–SWE), a three-dimensional shallow water model (3D–SWE) and three-dimensional Navier–Stokes model (3D–NSE), as well as two different coupled models combining 2D–SWE and 3D–SWE, and 2D–SWE and 3D–NSE, respectively, in order to benefit from the efficiency of the simple 2D–SWE model and the accuracy of the more complex 3D model. The 2D–SWE model has a low memory footprint, requiring n^2 (n represents a number of cells per direction) computing cells with only 5 unknowns to maintain throughout the time: 1) height of the water; 2) two components of the mean velocity; and 3) two components of the flux, saved in the cell-centred position. No additional interface reconstruction method is required, thus using a Riemann HLLC solver for the advection term results in a very good mass conservation. The 3D–SWE model, based on the 2D–SWE, solves the pressure Poisson equation in order to reconstruct the dynamic pressure value and all three components of the velocity vector field; therefore, adding the dynamic and static pressure components, density and three three-dimensional velocity values results in 11 variables that must be saved per cell throughout the time. As no additional interface reconstruction method is used, this method is, as implemented in MPFluid framework, only about 20% cheaper than the fully resolved 3D–NSE model. Combining 50% of 2D–SWE and 50% of 3D–SWE model resulted in the runtime reduction of about 42%. The 3D–NSE model is the most expensive model implemented, as it has to solve the pressure Poisson equation and capture the interface between two fluids using an external level-set method. In total 15 variables per cell are tracked throughout the simulation, and if Runge–Kutta second or third order is chosen, this number is further increased.

Furthermore, the Poisson equation solved for single-phase flows is extended to the variable-

coefficient Poisson equation that can be utilised for both single- and multi-phase flows, also giving a solid foundation for the further inclusion of compressible flow phenomena.

In order to accurately solve the discontinuities and shockwave events within multi-phase flows with large gradients in the transport properties, an approximate HLLC Riemann solver is implemented for both the 2D and 3D models, allowing direct computation of the volume fluxes at the faces of the computing cells. These schemes are applied to the advection terms of the respective equations and, although more expensive, solving four different fluxes per cell face than some other spatial schemes (e.g. central difference – no face flux solved, and upwind schemes – one flux value per face calculated), they contributed to a better mass conservation within the coupled models, at the 2D–3D interface border.

To enhance the accuracy in the smooth regions and preserve the stability in the discontinuous regions, first and second order upwind schemes are used in combination with several flux-limiting functions (e.g. MinMod, Sweby, Superbee, etc.) when it comes to the advection of the interface border between two fluids. These methods are of second-order accuracy, requiring a wide 5-elements stencil in one direction, thus in order to use them in the MPFluid framework two layers of ghost cells must be exchanged. This increases a memory footprint, as $2n^2$ cells must be exchanged per grid side. Nevertheless, all other methods tested that has not included such limiting functions are proven to be either too diffusive (e.g. upwind scheme), leading to the smeared solution, or too sharp (CD scheme), resulting in a numerical instability.

Explicit time-stepping methods are frequently used throughout this work, allowing a simple temporal discretisation that avoids the simultaneous solution of a system of linear equations. Yet, in order to remain stable, the time-step size is limited according to the CFL number and several additional conditions, separately included to satisfy the stability of each particular term of the system. One of these additional conditions is a consideration of the Riemann wave speeds while calculating a maximum velocity magnitude, that reduced the size of the time-stepping for about 25%–30%, comparing to the regular CFL condition. Comparing Euler explicit (EE), Adams–Bashforth (AB), Runge–Kutta first, second and third order (RK1, RK2 and RK3), the largest memory footprint with three additional temporary parameters saved is the AB scheme; the RK3 requires two additional variables, and the RK2 requires one additional variable in comparison to the EE scheme. The runtime of the EE, AB and RK1 schemes is almost identical (less than 2% difference). The RK2 is about 10%–15% more expensive, whereas the RK3 is about 25%-35% more expensive, the way it is implemented in MPFluid framework. The RK2 method delivers the most stable results in all simulation cases performed.

The validation and verification results attest to good concordance between the simulated scenarios and the experimental and numerical benchmarks, such as pure advection tests, solenoidal velocity field interface transport, flow around a prismatic obstacle, circular dam break, flooding of an idealised city and Kleefsman's dam-breaking experiment. Furthermore, the complexity of the flow behaviour is shown using several very complex geometric representations, followed by a detailed analysis.

Taking into consideration all the new models and procedures added, it can be concluded, that the MPFluid Framework, which is capable of executing a massive parallel single-flow simulation pipeline, is now enriched with another very complex branch of simulation processes,

with their own multi-flow simulation pipelines, suitable for execution on high-performance parallel computer architecture. Having those two separate kernels is a very good basis for further implementation and improvements, among which the most important ones are listed below:

Implicit time treatment: Within this work, several multi-step temporal explicit schemes are developed, notably improving the stability of the entire set of schemes chosen. Yet, the more complex the flow that is tackled, the worse the CFL limitations that are imposed on the whole system. Including implicit treatment will significantly reduce the time to solution, allowing a larger time-step size to be applied. Upon implementation, a thorough analysis should be conducted, comparing the overall performance with the currently available methods.

Turbulence: Two turbulence models, $k-\epsilon$ and Baldwin–Lomax, are implemented within the single-flow simulation pipeline, and a detailed analysis has been published by Lin [81] in his master's thesis. Following the work by him and several other authors, cited throughout this current work, a necessary efficiency analysis has already been carried out for multi-phase flows; nevertheless, implementation of the two most frequently used RANS models, $k-\epsilon$ and $k-\omega$ is yet to be completed. This would constitute a large improvement to the analysis of the local disturbance of a flow around an obstacle, and give us a more realistic picture of air–water interaction at the water surface interface, due to the different energy dissipation quantities triggered at the different length scales.

Level-set re-distance function vs. multigrid convergence: The re-distance function of the hyperbolic profile of the Level-set indicator, also called the reinitialisation function throughout the thesis, influences the convergence rate of the Poisson pressure equation, solved in the corrector step of the Chorin projection method. Not only does this function introduce tangential diffusion, which leads to deformation of the initial body shape, it also compresses the density field together with the level-set indicator field. Thus this compression is directly accounted for within the variable-coefficient pressure Poisson equation. It has been observed throughout this work that, if the reinitialisation is performed more than suggested in the standard literature, a stiffer solution is produced and the amount of iterations necessary for the Poisson solver to converge is increased up to four times. Such behaviour is observed in the case of both the Jacobi iterative solver and the multigrid approach. This issue must be analysed more in detail and the proper correlation parameters must be established in order to be able to fully control the simulation process. Furthermore, this study can include a wide variety of reinitialisation functions available on the market, to help to explain the general usability of those functions in combination with pressure-correction methods.

Water interface adaptive refinement: Adaptive refinement, available in the MPFluid Framework, is primarily developed to address an insufficiently small cell size close to solid obstacles, in order to resolve proper behaviour in the points with singularities, within the domain. This work is improved by means of an adaptive refinement based on the gradient of the transport property within the single-phase-flow kernel. Implementing the latter strategy on the interface tracking method (i.e. level-set or volume-of-fluid) would significantly reduce the risk of having an under-resolved normal vector calculation and the appearance of spurious currents at the interface between two or more fluids.

Interactive visualisation: The MPFluid Framework is developed in such a way that enables both online and offline visualisation strategies. The former is well documented in [119] and [104], and tested for single-phase flows; nevertheless, with the development of the multi-phase-flow branch, none of the functionality is affected, thus a straightforward visualisation of the newly developed kernel can be produced. The second, offline visualisation strategy is based on HDF5 technology and the entire work is published in [42] and [21]. Slight adjustment of the HDF5 kernel is necessary, in order to be compatible with the multi-phase-flow extension.

Sub-cycling: Having an adaptive refinement setting, in combination with calculation of the global time step based on CFL requirements, causes a significant reduction in the efficiency of the simulation. In this case, the small time-step value calculated for the finest mesh representation must be also used for the coarse mesh representation. Implementing a sub-cycling technique would allow a coarser mesh representation to be performed with a larger time-step size, leaving the remaining small time-step values only for a particular, well-refined part of the domain, yielding better overall temporal performance.

Sub-cycling – Coupled models: As two different coupled models are developed in the scope of this work, based on the 2D shallow water, 3D shallow water and 3D Navier–Stokes models, and these independent models have different complexities and time-step requirements, a model-based sub-cycling routine could be further developed, resulting in better temporal efficiency. At the moment, both coupled models use a globally defined time-step size, although there are many indications that the time-step size of the simple 2D model can be up to ten times as large as the time-step used within the more complex 3D model.

Decentralised organisational structure: Performing a numerical simulation of large real-world phenomena involves a high-performance parallel computer architecture, where the numerical domain is generated, divided into sub-domains and distributed to the available processing units, and the workload balancing is executed continuously, in order to enhance the efficiency of a single computing resource, while keeping the communication costs as low as possible. In order to achieve the latter, either the topology of the problem is shared among all computing units, which significantly increases the memory footprint, or else dedicated servers are used to facilitate the communication process. This centralised approach is, despite all the benefits, recognised as a bottleneck in the system, and this becomes much more significant with an increase in the number of computing units involved. According to Ertl et al. [43], a decentralised organisational structure may overcome this shortcoming, limiting the communication threads to direct hierarchical and geometrical neighbours. The change of communication strategy is already implemented on top of the MPFluid centralised approach; however, having this decentralised system would create a good foundation for further highly beneficial solver modifications (i.e. development of asynchronous iterative matrix solvers and convergence detection algorithms, see [44]).

Immersed boundaries or/and smart octree: Following the octree hierarchical approach implemented, solid objects within the flow have to be depicted using a step-wise representation, which leads to large modelling errors, especially when those objects are very complex and not aligned with the principal coordinate directions. In order to reduce these errors, either an immersed boundary representation has to be implemented, allowing higher-order treatment of

the boundaries (see [99] and [78], for instance), or a smart octree approach has to be considered, within which standard division into recursive octants is followed by movement of the voxel nodes onto the object's surface, in order to create boundary-conforming sub-cells, as published by Kudela et al. [77].

Fluid–structure interaction: Finally, a logical continuation, concerning overall development of the entire MPFluid Framework, would be the implementation of a structural kernel that would enable coupling of the multi-phase-flow kernel with six-degrees-of-freedom solid obstacles. This would allow us to consider even more complex and very realistic physical cases that can be found anywhere in nature. In order to proceed with this final step, it is highly probable that the data structure currently available will have to undergo quite serious modification, in order to represent all the necessary parameters of importance, while not losing an efficient memory management approach and validated high-performance parallel capability.

Bibliography

[1] A. A. Johnson and T. E. Tezduyar. **3D Simulation of fluid-particle interactions with the number of particles reaching 100**. Computer Methods in Applied Mechanics and Engineering, 145(3-4):301–321, 1997. doi: https://doi.org/10.1016/S0045-7825(96)01223-6.

[2] A. E. Bryson. **An Experimental Investigation of Transonic Flow Past Two-Dimensional Wedge and Circular-Arc Sections Using a Mach-Zehnder Interferometer**. NACA Technical Notes, Vol. 2560:1–97, 1951. doi: {https://resolver.caltech.edu/CaltechAUTHORS:20170728-105329495}.

[3] A. Harten, P. D. Lax, and B. V. Leer. **On Upstream Differencing and Godunov-Type Schemes for Hyperbolic Conservation Laws**. SIAM Review, Vol. 25(1):35–61, 1983. doi: https://doi.org/10.1137/1025002.

[4] A. Haselbacher and O. V. Vasilyev. **Commutative discrete filtering on unstructured grids based on least-squares techniques**. Journal of Computational Physics, Vol. 187(1):197–211, 2003. doi: https://doi.org/10.1016/S0021-9991(03)00095-0.

[5] A. J. Chorin. **The numerical solution of the Navier-Stokes equations for an incompressible fluid**. Bulletin of the American Mathematical Society, Vol. 73(6):928–931, 1967. doi: https://projecteuclid.org/euclid.bams/1183529112.

[6] A. Kurganov and D. Levy. **Central-Upwind Schemes for the Saint-Venant System**. International Journal for Numerical Methods in Fluids, Vol. 36(3):397–425, 2002. doi: https://doi.org/10.1051/m2an:2002019.

[7] A. Pathak and M. Raessi. **A 3D, fully Eulerian, VOF-based solver to study the interaction between two fluids and moving rigid bodies using the fictitious domain method**. Journal of Computational Physics, Vol. 311:87–113, 2016. doi: https://doi.org/10.1016/j.jcp.2016.01.025.

[8] A. Salama and P. J. Van Geel. **Flow and Solute Transport in Saturated Porous Media: 1. The Continuum Hypothesis**. Journal of Porous Media, 11(4):421–441, 2008. doi: 10.1615/JPorMedia.v11.i5.10.

[9] Grégoire Allaire. **Introduction to homogenization theory**, 2010. URL http://www.cmap.polytechnique.fr/~allaire/homog/lect1.pdf.

[10] O. Antepara. **Adaptive mesh refinement method for CFD applications**. PhD thesis, Universitat Politecnica de Catalunya, 2019.

[11] B. Einfeldt, C. D. Munz, P. L. Roe, and B. Sjögreen. **On Godunov-type methods near low densities**. SIAM Journal on Scientific Computing, Vol. 92(2):273–295, 1991. doi: https://doi.org/10.1016/0021-9991(91)90211-3.

[12] B. Lalanne, L. R. Villegas, S. Tanguy, and F. Risso. **On the computation of viscous terms for incompressible two-phase flows with Level Set/Ghost Fluid Method**. Journal of Computational Physics, 301:289–307, 2015. doi: https://doi.org/10.1016/j.jcp.2015.08.036.

[13] B. M. Ginting. **A two-dimensional artificial viscosity technique for modelling discontinuity in shallow water flows**. Applied Mathematical Modelling, Vol. 45: 653–683, 2017. doi: https://doi.org/10.1016/j.apm.2017.01.013.

[14] B. M. Ginting. **Central-upwind scheme for 2D turbulent shallow flows using high-resolution meshes with scalable wall functions**. Computers & Fluids, Vol. 179:394–421, 2017. doi: https://doi.org/10.1016/j.compfluid.2018.11.014.

[15] B. M. Ginting, R. P. Mundani, and E. Rank. **Parallel Simulations of Shallow Water Solvers for Modelling Overland Flows**. Proc. in 13th International Conference on Hydrodynamics, Vol. 3:788–799, 2018. doi: https://doi.org/10.29007/wdn8.

[16] B. van Leer. **Towards the ultimate conservative difference scheme. II. Monotonicity and conservation combined in a second-order scheme**. Journal of Computational Physics, Vol. 14(4):361–370, 1974. doi: https://doi.org/10.1016/0021-9991(74)90019-9.

[17] M. Bader. **Space Filling Curves: An Introduction with Applications in Scientific Computing**. Springer, 2013. ISBN 978-3-642-31046-1.

[18] J. Bear. **Dynamics of Fluids in Porous Media**. Dover Publications Inc, 1989. ISBN 9780486656755 / 0486656756.

[19] P. K. Bhola. **Dynamic flood inundation forecasting in real-time including associated uncertainties for operational flood risk management**. PhD thesis, Technische Universität München, 2019.

[20] C. Ai, S. Jin, and B. Lv. **A new fully non-hydrostatic 3D free surface flow model for waterwave motions**. Internationnal Journal for Numerical Methods in Fluids, Vol. 66:1354–1377, 2011. doi: https://doi.org/10.1002/fld.2317.

[21] C. Ertl, J. Frisch, and R. P. Mundani. **Design and optimisation of an efficient HDF5 I/O Kernel for massive parallel fluid flow simulations**. Concurrency and Computation: Practice and Experience, 29(24):1–8, 2017. doi: https://doi.org/10.1002/cpe.4165.

[22] C. Hirsch. **Numerical Computation of Internal and External Flows**. Butterworth-Heinemann, 2007. ISBN 9780750665940.

[23] C. K. Sollitt and R. H. Cross. **Wave Transmission through Permeable Breakwaters**. 13th International Conference on Coastal Engineering, pages 1827–1846, 1972. URL https://ascelibrary.org/doi/abs/10.1061/9780872620490.106.

[24] C. M. Rhie and W. L. Chow. **Numerical study of the turbulent flow past an airfoil with trailing edge separation**. AIAA Journal, Vol. 21(11):1525–1532, 1983. doi: https://doi.org/10.2514/3.8284.

[25] C. W. Hirt and B. D. Nichols. **Volume of fluid (VOF) method for the dynamics of free boundaries**. Journal of Computational Physics, Vol. 39(1):201–225, 1981. doi: https://doi.org/10.1016/0021-9991(81)90145-5.

[26] C.-W. Shu and S. Osher. **Efficient Implementation of Essentially Non-Oscillatory Shock Capturing Schemes**. Journal of Computational Physics, Vol. 77(2):439–471, 1988. doi: https://doi.org/10.1016/0021-9991(88)90177-5.

[27] A. Chiapolino and R. Saurel. **Models and methods for two-layer shallow water flows**. Numerical methods and modeling of multiphase flows, 371:1043–1066, 2018. doi: https://doi.org/10.1016/j.jcp.2018.05.034.

[28] D. A. Zimmerman, G. De Marsily, C. A. Gotway, M. G. Marietta, C. L. Axness, R. L. Beauheim, R. L. Bras, J. Carrera, G. Dagan, P. B. Davies, D. P. Gallegos, A. Galli, J. Góez-Hernández, P. Grindrod, A. L. Gutjahr, P. K. Kitanidis, A. M. Lavenue, D. McLaughlin, S. P. Neuman, B. S. Ravenne, and Y. Rubin. **A comparison of seven geostatistically based inverse approaches to estimate transmissivities for modeling advective transport by groundwater flow**. Water Resources Research, 34(6):1373–1413, 1998. doi: https://doi.org/10.1029/98WR00003.

[29] D. Greaves. **A quadtree adaptive method for simulating fluid flows with moving interfaces**. Jornal of Computational Physics, 194(1):35–56, 2003. doi: https://doi.org/10.1016/j.jcp.2003.08.018.

[30] D. Kuzmin. **On the design of general-purpose flux limiters for finite element schemes. I. Scalar convection**. Journal of Computational Physics, Vol. 219(2):513–531, 2006. doi: {https://doi.org/10.1016/j.jcp.2006.03.034}.

[31] D. L. Sun and W. Q. Tao. **A coupled volume-of-fluid and level set (VOSET) method for computing incompressible two-phase flows**. International Journal of Heat and Mass Transfer, Vol. 53(4):645–655, 2009. doi: https://doi.org/10.1016/j.ijheatmasstransfer.2009.10.030.

[32] D. Shepard. **A two-dimensional interpolation function for irregularly-spaced data**. Proceedings of the 1968 ACM National Conference, Vol. 187(1):517–524, 1968. doi: https://doi.org/10.1145/800186.810616.

[33] E. Dick, K. Riemslagh, and J. Vierendeels. **Multigrid Methods VI: Lecture Notes in Computational Science and Engineering**. Springer, 2000. ISBN 978-3-5406-7157-2.

[34] E. F. Toro, M. Spruce, and W. Speares. **Restoration of the contact surface in the HLL-Riemann solver**. Shock Waves, Vol. 4(25):25–34, 1994. doi: https://doi.org/10.1007/BF01414629.

[35] E. Olsson and G. Kreiss. **A conservative level set method for two phase flow**. Journal of Computational Physics, Vol. 210:225–246, 2004. doi: https://doi.org/10.1016/j.jcp.2005.04.007.

[36] E. Olsson, G. Kreiss, and S. Zahedi. **A conservative level set method for two phase flow II**. Journal of Computational Physics, Vol. 225(1):785–807, 2006. doi: https://doi.org/10.1016/j.jcp.2006.12.027.

[37] E. R. Toro. **Shock-Capturing Methods for Free-Surface Shallow Flows**. John Wiley & Sons, Inc., 2001. ISBN 978-0-471-98766-6.

[38] E. R. Toro. **Riemann Solvers and Numerical Methods for Fluid Dynamics**. Springer-Verlag Berlin Heidelberg, 2009. ISBN 978-3-540-25202-3.

[39] E. Schillaci, N. Balcázar, L. Jofre, O. Lehmkuhl, and J. Castro. **A free surface model for the numerical simulation of oscillating water column systems**. 6th Conference on Computational Fluid Dynamics, pages 1–12, 2014. doi: https://doi.org/10.1016/j.compfluid.2016.09.014.

[40] E. Schillaci, L. Jofre, N. Balcázar, O. Lehmkuhl, and A. Oliva. **A Level-Set Aided Single-Phase Model for the Numerical Simulation of Free-Surface Flow on Unstructured Meshes**. 70th Conference of the ATI Engineering Association, 140: 97–110, 2016. doi: https://doi.org/10.1016/j.compfluid.2016.09.014.

[41] C. Ericson. **Real-Time Collision Detection**. CRC Press, 2004. ISBN 1558607323.

[42] C. Ertl. **Scalable Parallel I/O using HDF5 for HPC Fluid Flow Simulations**. Master's thesis, Technische Universität München, June 2015.

[43] C. Ertl, R.-P. Mundani, and E. Rank. **Ensuring domain consistency in an adaptive framework with distributed topology for fluid flow simulations**. CoRR, abs/1807.00312, 2018. URL http://arxiv.org/abs/1807.00312.

[44] C. Ertl, R.-P. Mundani, and E. Rank. **Ensuring domain consistency in an adaptive framework with distributed topology for fluid flow simulations**. CoRR, abs/1807.00312, 2018. URL http://arxiv.org/abs/1807.00312.

[45] Esri. **ArcGIS inverse distance weighting implementation**, 2015. URL https://pro.arcgis.com/en/pro-app/latest/help/analysis/geostatistical-analyst/how-inverse-distance-weighted-interpolation-works.htm.

[46] F. Mintgen and M. Manhart. **A bi-directional coupling of 2D shallow water and 3D Reynolds-averaged Navier–Stokes models**. Journal of Hydraulic Research, Vol. 56(6):771–785, 2017. doi: {doi:10.1080/00221686.2017.14199891}.

[47] F. N. Felten and T. S. Lund. **Kinetic energy conservation issues associated with the collocated mesh scheme for incompressible flow**. Journal of Computational Physics, Vol. 215(2):465–484, 2006. doi: https://doi.org/10.1016/j.jcp.2005.11.009.

[48] F. Nicoud. **Conservative High-Order Finite-Difference Schemes for Low-Mach Number Flows**. Journal of Computational Physics, Vol. 158(1):71–97, 2000. doi: {https://doi.org/10.1006/jcph.1999.6408}.

[49] FloodEvac. **Vulnerability of Transportation Structures Warning and Evacuation in Case of Major Inland Flooding**, 2015. URL http://www.floodevac.org/.

[50] A. C. Frank. **Organisationsprinzipien zur Integration von geometrischer Modellierung, numerischer Simulation und Visualisierung**. PhD thesis, Technische Universität München, 2000.

[51] J. Frisch. **Towards Massive Parallel Fluid Flow Simulations**. PhD thesis, Technische Universität München, 2014.

[52] G. A. Narsilio, O. Buzzi, S. Fityus, T. S. Yun, and D. W. Smith. **Upscaling of Navier-Stokes equations in porous media: Theoretical, numerical and experimental approach**. Computers and Geotechnics, Vol. 36(7):1200–1206, 2009. doi: https://doi.org/10.1016/j.compgeo.2009.05.006.

[53] G. Guderley and H. Yoshihara. **Two-Dimensional Unsymmetric Flow Patterns at Mach Number 1**. Journal of the Aeronautical Sciences, Vol. 20(11):757–768, 1953. doi: {https://doi.org/10.2514/8.2831}.

[54] G. Tryggvason, B. Bunner, A. Esmaeeli, D. Juric, N. Al-Rawahi, W. Tauber, J. Han, S. Nas, and Y.-J. Jan. **A Front Tracking Method for the Computations of Multiphase Flow**. Jornal of Computational Physics, 169:708–759, 2001. doi: https://doi.org/10.1006/jcph.2001.6726.

[55] G. Tryggvason, B. Bunner, A. Esmaeeli, D. Juric, N. Al-Rawahi, W. Tauber, J. Han, S. Nas, and Y.-J. Jan. **A Front-Tracking Method for the Computations of Multiphase Flow**. Journal of Computational Physics, Vol. 169(2):708–759, 2001. doi: https://doi.org/10.1006/jcph.2001.6726.

[56] H. H. Hu, N. A. Patankar, and M. Y. Zhu. **Direct Numerical Simulations of Fluid-Solid Systems Using the Arbitrary Lagrangian-Eulerian Technique**. Jornal of Computational Physics, 169(2):427–462, 2001. doi: https://doi.org/10.1006/jcph.2000.6592.

[57] H. Zeng, L. Grbcic, I. Lucin, and L. Kranjcevic. **Mesh Creation for Realistic Terrain Cases for Shallowfoam - 2D OpenFOAM Solver**. Proceedings of the 29th DAAAM International Symposium, Vol. 20(4):1065–1070, 2018. URL 10.2507/29th.daaam.proceedings.152.

[58] H. Zeng, L. Kranjcevic, and M. Manhart. **An Approach to Couple Shallow Water and Navier-Stokes Suited for Upstream-Travelling Shock Waves Based on**

Openfoam Framework – Shallowinterfoam. Proceedings of the 38th IAHR World Congress, pages 1–9, 2019. doi: {https://doi.org/10.2514/8.2831}.

[59] N. M. Hunter, P. D. Bates, s. Neelz, G. Pender, I. Villanueva, N. G. Wright, D. Liang, R. A. Falconer, B. Lin, S. Waller, A. J. Crossley, and D. C. Mason. **Benchmarking 2D hydraulic models for urban flooding**. Proceedings of the ICE – Water Management, 161:13–30, 2008. doi: https://doi.org/10.1680/wama.2008.161.1.13.

[60] J. B. Bell and D. L. Marcus. **A Second-Order Projection Method for Variable-Density Flows**. Journal of Computational Physics, Vol. 101(2):334–348, 1992. doi: https://doi.org/10.1016/0021-9991(92)90011-M.

[61] J. C. Martin, W. J. Moyce, W. G. Penney, A. T. Price, and C. K. Thornhill. **Part V. An experimental study of the collapse of fluid columns on a rigid horizontal plane, in a medium of lower, but comparable, density**. Journal of Computational Physics, Vol. 244(882):134–198, 1952. doi: https://doi.org/10.1098/rsta.1952.0007.

[62] J. G. Zhou and P. K. Stansby. **An arbitrary Lagrangian-Eulerian sigma (ALES) model with non-hydrostatic pressure for shallow water flows**. Internationnal Journal for Numerical Methods in Fluids, Vol. 178(1–2):199–214, 1999. doi: https://doi.org/10.1016/S0045-7825(99)00014-6.

[63] J. G. Zhou, D. M. Causon, C. G. Mingham, and D. M. Ingram. **The Surface Gradient Method for the Treatment of Source Terms in the Shallow–Water Equations**. Journal of Computational Physics, Vol. 186(1):1–25, 2001. doi: https://doi.org/10.1006/jcph.2000.6670.

[64] J. Glimm, J. W. Grove, and X. L. Liand W. Ohand D. H. Sharp. **A Critical Analysis of Rayleigh-Taylor Growth Rates**. Jornal of Computational Physics, 169(2):652–677, 2001. doi: https://doi.org/10.1006/jcph.2000.6590.

[65] J. H. Ferziger and M. Perić. **Computational Methods for Fluid Dynamics**. Springer Berlin Heidelber, 2001. ISBN 3540420746 / 9783540420743.

[66] J. Hou, F. Simons, M. Mahgoub, and R. Hinkelmann. **A robust well-balanced model on unstructured grids for shallow water flows with wetting and drying over complex topography**. Computer Methods in Applied Mechanics and Engineering, Vol. 257:126–149, 2013. doi: https://doi.org/10.1016/j.cma.2013.01.015.

[67] J. Mencinger and I. Zun. **On the finite volume discretization of discontinuous body force field on collocated grid: Application to VOF method**. Journal of Computational Physics, Vol. 221:524–538, 2007. doi: https://doi.org/10.1016/j.jcp.2006.06.021.

[68] J. P. Chiles and P. Delfiner. **Geostatistics, Modeling Spatial Uncertainty**. John Wiley & Sons, Inc., 2012. ISBN 978-1-118-13618-8.

[69] T. Jungblut. **Optimisation of a Parallel Multigrid Solver for HPC Fluid Flow Simulations**. Master's thesis, Technische Universität München, November 2016.

[70] K. Bagi. **An algorithm to generate random dense arrangements for discrete element simulations of granular assemblies**. Journal of Applied Physics, Vol. 7(1): 31–43, 2005. doi: https://doi.org/10.1007/s10035-004-0187-5.

[71] K. Han, Y.T. Feng, and D.R.J. Owen. **Sphere packing with a geometric based compression algorithm**. Powder Technology, Vol. 155(1):33–41, 2005. doi: https://doi.org/10.1016/j.powtec.2005.04.055.

[72] K. K. So, X. Y. Hu, and N. A. Adams. **Anti-diffusion method for interface steepening in two-phase incompressible flow**. Journal of Computational Physics, Vol. 230(13):5155–5177, 2011. doi: https://doi.org/10.1016/j.jcp.2011.03.011.

[73] K. M. T. Kleefsman, G. Fekken, A. E. P. Veldman, B. Iwanowski, and B. Buchner. **A Volume-of-Fluid based simulation method for wave impact problems**. Journal of Computational Physics, 206:363–393, 2005. doi: https://doi.org/10.1016/j.jcp.2004.12.007.

[74] K. R. Rijagopal. **On a hierarchy of approximate models for flows of incompressible fluids through porous solids**. Mathematical Models and Methods in Applied Sciences, Vol. 17(2):215–252, 2007. doi: {https://doi.org/10.1142/S0218202507001899}.

[75] K.-S. Choi. **Experimental Research on Turbulent Flow over Compliant Walls**. Fluid Mechanics and Its Applications, Vol. 72:275–292, 2003. doi: {https://doi.org/10.1007/978-94-017-0415-1_12}.

[76] KellyBC. **Open Source STL Geometry File**, 2012. URL https://thingiverse.com/thing:32092.

[77] L. Kudela, N. Zander, S. Kollmannsberger, and E. Rank. **Smart octrees: accurately integrating discontinuous functions in 3D**. Computer Methods in Applied Mechanics and Engineering, 2016. doi: 10.1016/j.cma.2016.04.006. URL http://www.cie.bgu.tum.de/publications/2016_Kudela_SmartOctree.

[78] L. Kudela, S. Kollmannsberger, and E. Rank. **An Immersed Boundary Approach for the Numerical Analysis of Objects Represented by Oriented Point Clouds**. Computational Modeling of Objects Presented in Images. Fundamentals, Methods, and Applications, 10986:33–41, 2019. doi: https://doi.org/10.1007/978-3-030-20805-9_4.

[79] L. Jofre, O. Lehmkuhl, J. Castro, and A. Oliva. **A Method for Capturing Sharp Fluid Interfaces on Arbitrary Meshes**. Computers & Fluids, Vol. 94:14–29, 1999. doi: https://doi.org/10.1016/j.compfluid.2014.02.001.

[80] L. Jofre, O. Lehmkuhl, J. Ventosa, F. Xavier, and A. Oliva. **Conservation Properties of Unstructured Finite-Volume Mesh Schemes for the Navier-Stokes Equations**. Numerical Heat Transfer, Part B: Fundamentals: An International Journal of Computation and Methodology, Vol. 65(1):53–79, 2014. doi: https://doi.org/10.1080/10407790.2013.836335.

[81] Q. Lin. **Integration of Baldwin–Lomax and k-ϵ RANS Numerical Models into the HPC Fluid Flow Simulation Framework**. Master's thesis, Technische Universität München, Jul 2016.

[82] M. A. Oliver and R. Webster. **Kriging: a method of interpolation for geographical information system**. International Jounal of Geographical Information Systems, 4(3): 313–332, 1990. doi: https://doi.org/10.1080/02693799008941549.

[83] M. B. Kocyigit, R. A. Falconer, and B. Lin. **Three-dimensional numerical modelling of free surface flows with non-hydrostatic pressure**. Internationnal Journal for Numerical Methods in Fluids, Vol. 40:1145–1162, 2002. doi: 10.1002/fld.376.

[84] M. Bader, H. J. Bungartz, A. Frank, and R. P. Mundani. **Space Tree Structures for PDE Software**. Lecture Notes in Computer Science, 2331:662–671, 2002. doi: https://doi.org/10.1007/3-540-47789-6_69.

[85] M. E. Hubbard. **Multidimensional Slope Limiters for MUSCL-Type Finite Volume Schemes on Unstructured Grids**. Journal of Computational Physics, Vol. 155:54–74, 1999. doi: {https://doi.org/10.1006/jcph.1999.6329}.

[86] M. Elhaddad, N. Zander, T. Bog, L. Kudela, S. Kollmannsberger, J. Kirschke, T. Baum, M. Ruess, and E.Rank. **Multi-level hp-finite cell method for embedded interface problems with application in biomechanics**. Journal of Computational Physics, 34 (4):1–32, 2018. doi: https://doi.org/10.1002/cnm.2951.

[87] M. Hassanizadeh and W. Gray. **General Conservation Equations for Multiphase Systems: 1. Averaging Procedure**. Advances in Water Resources, 2:131–144, 1979. doi: https://doi.org/10.1016/0309-1708(79)90025-3.

[88] M. Lieb, M. Mehl, T. Neckel, and K. Unterweger. **HPC Fluid Flow Simulations in Porous Media Geometries**. Seventh International Conference on Computational Fluid Dynamics, pages 1–12, 2012. doi: http://iccfd.org/iccfd7/assets/pdf/papers/ICCFD7-2905_paper.pdf.

[89] M. Malik, E. S.-C. Fan, and M. Bussmann. **Adaptive VOF with curvature-based refinement**. Internationnal Journal for Numerical Methods in Fluids, Vol. 55(7):693–712, 2007. doi: https://doi.org/10.1002/fld.1490.

[90] M. Muskat and M. W. Meres. **The Flow of Heterogeneous Fluids Through Porous Media**. Journal of Applied Physics, Vol. 7(9):346–363, 1936. doi: https://doi.org/10.1063/1.1745403.

[91] M. Sussman. **A second order coupled level set and volume-of-fluid method for computing growth and collapse of vapor bubbles**. Journal of Computational Physics, Vol. 187(1):110–136, 2003. doi: https://doi.org/10.1016/S0021-9991(03)00087-1.

[92] M. Sussman, E. Fatemi, P. Smereka, and S. Osher. **An improved level set method for incompressible two-phase flows**. Computers and Fluids, Vol. 27(5–6):663–680, 1998. doi: https://doi.org/10.1016/S0045-7930(97)00053-4.

[93] M. Sussman, K.M. Smith, M. Y. Hussaini, and M. Ohtaand R. Zhi-Wei. **An improved level set method for incompressible two-phase flows**. Journal of Computational Physics, Vol. 221(2):469–505, 2006. doi: https://doi.org/10.1016/j.jcp.2006.06.020.

[94] M. Sussman, K. M. Smith, M. Y. Hussaini, M. Ohta, and R. Zhi-Wei. **A sharp interface method for incompressible two-phase flows**. A sharp interface method for incompressible two-phase flows, Vol. 221(2):469–505, 2007. doi: https://doi.org/10.1016/j.jcp.2006.06.020.

[95] M. van S. Annaland, N. G. Deen, and J. A. M. Kuipers. **Numerical simulation of gas bubbles behaviour using a three-dimensional volume of fluid method**. Chemical Engineering Science, 60(11):2999–3011, 2008. doi: https://doi.org/10.1016/j.ces.2005.01.031.

[96] S. F. McCormick. **Multigrid Methods: Theory, Applications, and Supercomputing**. Series: Lecture Notes in Pure and Applied Mathematics. CRC Press, 1988. ISBN 9780824779795.

[97] Peter Fitzpatrick 'MojoBob'. **Single Span Stone Road Bridge**, 2018. URL https://www.thingiverse.com/thing:3061573.

[98] J. C. Muela. **Modelling and numerical simulation of combustion and multiphase flows using finite volume methods on unstructured meshes**. PhD thesis, Universitat Politecnica de Catalunya, 2018.

[99] B. Müller, S. Krämer-Eis, F. Kummer, and M. Oberlack. **A high-order discontinuous Galerkin method for compressible flows with immersed boundaries**. International Journal for Numerical Methods in Engineering, 110(1):3–30, 2017. doi: https://onlinelibrary.wiley.com/doi/abs/10.1002/nme.5343.

[100] R. P. Mundani. **Hierarchische Geometriemodelle zur Einbettung verteilter Simulationsaufgaben**. PhD thesis, Technische Universität München, 2006.

[101] N. Balcázar, L. Jofre, O. Lehmkuhl, J. Castro, and J. Rigola. **A finite-volume/level-set method for simulating two-phase flows on unstructured grids**. International Journal of Multiphase Flow, Vol. 64:55–72, 2014. doi: https://doi.org/10.1016/j.ijmultiphaseflow.2014.04.008.

[102] N. Balcázar, L. Jofre, O. Lehmkuhl, J. Rigola, and A. Oliva. **A coupled volume-of-fluid/level-set method for simulation of two-phase flows on unstructured meshes**. Computers and Fluids, Vol. 124:12–29, 2015. doi: https://doi.org/10.1016/j.compfluid.2015.10.005.

[103] N. G. Deen, M. van S. Annaland, and J. A. M. Kuipers. **Direct numerical simulation of complex multi-fluid flows using a combined front tracking and immersed boundary method**. International Journal of Multiphase Flow, Vol. 64(9):2186–2201, 2009. doi: https://doi.org/10.1016/j.ces.2009.01.029.

[104] N. Perović, J. Frisch, R. P. Mundani, and E. Rank. **Interactive data exploration for high-performance fluid flow computations through porous media**. 16th Int. Symposium on Symbolic and Numeric Algorithms for Scientific Computing, pages 1–8, 2014. doi: https://doi.org/10.1016/j.advengsoft.2015.02.003.

[105] N. Perović, J. Frisch, A. Salama, S. Sun, E. Rank, and R.P. Mundani. **Multi-scale high-performance fluid flow: Simulations through porous media**. Advances in Engineering Software, Vol. 103:85–98, 2016. doi: https://doi.org/10.1016/j.advengsoft.2016.07.016.

[106] O. Desjardins, V. Moureau, and H. Pitsch. **An accurate conservative level set/ghost fluid method for simulating turbulent atomization**. Journal of Computational Physics, 227(18):8395–8416, 2008. doi: https://doi.org/10.1016/j.jcp.2008.05.027.

[107] O. Ubbink and R. I. Issa. **A Method for Capturing Sharp Fluid Interfaces on Arbitrary Meshes**. Journal of Computational Physics, Vol. 153(1):26–50, 1999. doi: https://doi.org/10.1006/jcph.1999.6276.

[108] German Federal Ministry of Education and Research. **BMBF**, 2015. URL https://www.bmbf.de/.

[109] P. Batten, C. Lambert, and D. M. Causon. **Positively Conservative High-Resolution Convection Schemes for Unstructured Elements**. International Journal for Numerical Methods in Engineering, Vol. 39(11):1821–1838, 1996. doi: https://doi.org/10.1002/(SICI)1097-0207(19960615)39:11<1821::AID-NME929>3.0.CO;2-E.

[110] P. Batten, N. Clarke, C. Lambert, and D. M. Causon. **On the Choice of Wavespeeds for the HLLC Riemann Solver**. SIAM Journal on Scientific Computing, Vol. 18(6):1553–1570, 1997. doi: https://doi.org/10.1137/S1064827593260140.

[111] P. K. Stansby and J. G. Zhou. **Shallow-water flow solver with non-hydrostatic pressure: 2D vertical plane problems**. Internationnal Journal for Numerical Methods in Fluids, Vol. 28(3):541–563, 1998. doi: https://doi.org/10.1002/(SICI)1097-0363(19980915)28:3<541::AID-FLD738>3.0.CO;2-0.

[112] P.-K. Sweby. **High Resolution Schemes Using Flux Limiters for Hyperbolic Conservation Laws**. SIAM Journal on Numerical Analysis, Vol. 21(5):995–1011, 1984. doi: https://www.jstor.org/stable/2156939.

[113] P. L. Roe. **Characteristic-Based Schemes for the Euler Equations**. Annual Review of Fluid Mechanics, Vol. 18:337–365, 1986. doi: https://doi.org/10.1146/annurev.fl.18.010186.002005.

[114] P. Wesseling. **An Introduction to Multigrid Methods**. J.Wiley, 1992. ISBN 978-0-4719-3083-9.

[115] P. Wesseling and C. W. Oosterlee. **Geometric multigrid with applications to computational fluid dynamics**. Journal of Computational and Applied Mathematics, Vol. 128(1–2):311–334, 2001. doi: {https://doi.org/10.1016/S0377-0427(00)00517-3}.

[116] Fabarts Project. **Ponte 25 de Abril**, 2017. URL https://www.thingiverse.com/thing:2350860.

[117] Q. Liang, J. Hou, and X. Xia. **Contradiction between the C-property and mass conservation in adaptive grid based shallow flow models: cause and solution**. Numerical Methods in Fluids, Vol. 78:17–36, 2015. doi: https://doi.org/10.1002/fld.4005.

[118] R. K. Shukla, C. Pantano, and J. B. Freund. **An interface capturing method for the simulation of multi-phase compressible flows**. Journal of Computational Physics, Vol. 229(19):7411–7439, 2010. doi: https://doi.org/10.1016/j.jcp.2010.06.025.

[119] R. P. Mundani, J. Frisch, V. Varduhn, and E. Rank. **A sliding window technique for interactive high-performance computing scenarios**. Advances in Engineering Software, 84(C):1–32, 2015. doi: https://doi.org/10.1016/j.advengsoft.2015.02.003.

[120] Ankit Rohatgi. **WebPlotDigitizer**, 2019. URL https://automeris.io/WebPlotDigitizer.

[121] S. F. Davis. **Simplified Second-Order Godunov-Type Methods**. SIAM Journal on Scientific and Statistical Computing, Vol. 9(3):445–473, 1986. doi: https://doi.org/10.1137/0909030.

[122] S. G. Mikhlin. **On the Schwarz algorithm**. Doklady Akademii Nauk SSSR, Vol. 77: 569–571, 1951. doi: {https://zbmath.org/}.

[123] S. Gottlieb and C.-W. Shu. **Total variation diminishing Runge-Kutta schemes**. Mathematics of Computations, Vol. 152:73–85, 1998. doi: https://doi.org/10.1090/S0025-5718-98-00913-2.

[124] S. K. Godunov. **A finite difference method for the computation of discontinuous solutions of the equations of fluid dynamics**. Matematicheskii Sbornik, Vol. 47: 357–393, 1959.

[125] S. Osher and J. A. Sethian. **Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations**. Journal of Computational Physics, Vol. 79(1):12–49, 1988. doi: https://doi.org/10.1016/0021-9991(88)90002-2.

[126] S. Soares-Franzão and Y. Zech. **Experimental study of dam-break flow against an isolated obstacle**. Journal of Hydraulic Research, Vol. 45:27–36, 2007. doi: https://doi.org/10.1080/00221686.2007.9521830.

[127] S. Soares-Franzão and Y. Zech. **Dam-break flow through an idealised city**. Journal of Hydraulic Research, Vol. 46(5):648–658, 2010. doi: https://doi.org/10.3826/jhr.2008.3164.

[128] S. Takagi, Y. Matsumoto, and H. Huang. **Numerical Analysis of a Single Rising Bubble Using Boundary-Fitted Coordinate System**. JSME International Journal Seriaes B Fluids and Thermal Engineering, 40(1):42–50, 1997. doi: https://doi.org/10.1299/jsmeb.40.42.

[129] S. Whitaker. **Flow in Porous Media I : A Theoretical Derivation of Darcy's Law**. in Transport in Porous Media, Vol. 1(1):3–25, 1986. doi: https://doi.org/10.1007/BF01036523.

[130] S. Whitaker. **The Method of Volume Averaging**. Springer Netherlands, 1999. ISBN 10.1007/978-94-017-3389-2.

[131] H. A. Schwarz. **Über einen Grenzübergang durch alternirendes Verfahren**. Vierteljahrsschrift der Naturforschenden Gesellschaft in Zürich. Zürcher u. Furrer, 1870. URL https://books.google.de/books?id=tMcxYAAACAAJ.

[132] T. Barth and D. Jespersen. **The design and application of upwind schemes on unstructured meshes**. 27th Aerospace Sciences Meeting, pages 1–22, 1989. doi: {https://doi.org/10.2514/6.1989-366}.

[133] T. Fondelli, A. Andreini, and B. Facchini. **Numerical Simulation of Dam-Break Problem Using an Adaptive Meshing Approach**. 70th Conference of the ATI Engineering Association, 82:309–315, 2015. doi: https://doi.org/10.1016/j.egypro.2015.12.038.

[134] T. Wacławczyk. **A consistent solution of the re-initialization equation in the conservative level-set method**. Journal of Computational Physics, Vol. 299:487–525, 2015. doi: 10.1016/j.ijmultiphaseflow.2017.08.003.

[135] T. Wacławczyk. **On a relation between the volume of fluid, level-set and phase field interface models**. Journal of Computational Physics, Vol. 97:60–77, 2017. doi: https://doi.org/10.1016/j.jcp.2015.06.029.

[136] T. Zhu, C. Waluga, B. Wohlmuth, and M. Manhart. **A Study of the Time Constant in Unsteady Porous Media Flow Using Direct Numerical Simulation**. Transp Porous Med, Vol. 104:161–179, 2014. doi: https://doi.org/10.1007/s11242-014-0326-3.

[137] U. Trottenberg, C. W. Oosterlee, and A. Schuller. **Multigrid Methods for Finite Elements**. Academic Press, 2001. ISBN 978-0-1270-1070-0.

[138] Unknown. **High-resolution Dragon Model**, 2017. URL http://3dmag.org/en/market/item/4866/.

[139] V. Casulli. **A semi-implicit Finite Difference Method for Non-Hydrostatic, Free Surface Flows**. Internationnal Journal for Numerical Methods in Fluids, Vol. 30: 425–440, 1999. doi: 0271âĂŞ2091/99/120425âĂŞ16.

[140] V. Coralic and T. Colonius. **Finite-volume WENO scheme for viscous compressible multicomponent flows**. Journal of Computational Physics, Vol. 274:95–121, 2014. doi: https://doi.org/10.1016/j.jcp.2014.06.003.

[141] V.-T. Nguyen and W.-G. Park. **A free surface flow solver for complex three-dimensional water impact problems based on the VOF method**. International Journal for Numerical Methods in Fluids, 81(1):3–34, 2016. doi: https://doi.org/10.1002/fld.4203.

[142] V.-T. Nguyen and W.-G. Park. **A volume-of-fluid (VOF) interface-sharpening method for two-phase incompressible flows**. Computers & Fluids, Vol. 152:104–119, 2017. doi: https://doi.org/10.1016/j.compfluid.2017.04.018.

[143] V. V. Shaidurov. **Multigrid Methods for Finite Elements**. Springer Netherlands, 1995. ISBN 978-94-015-8527-9.

[144] B. van Leer. **B. van Leer**, 2009. URL https://en.wikipedia.org/wiki/Bram_van_Leer.

[145] R. W. C. P. Verstappen, M. T. Dröge, and A. E. P. Veldman. **Symmetry-Preserving Discretization for DNS**. Direct and Large-Eddy Simulation V. ERCOFTAC Series, pages 135–146, 2004. doi: https://doi.org/10.1007/978-1-4020-2313-2_16.

[146] W. Aniszewski, T. Menard, and M. Marek. **Volume of Fluid (VOF) type advection methods in two-phase flow: A comparative study**. Computers and Fluids, Vol. 97:52–73, 2014. doi: https://doi.org/10.1016/j.compfluid.2014.03.027.

[147] W. G. Gray. **A Derivation of the Equations for Multiphase Transport**. Chemical Engineering Science, 30:229–233, 1975. doi: https://doi.org/10.1016/0009-2509(75)80010-8.

[148] W. Hackbusch. **Multi-Grid Methods and Applications**. Springer-Verlag Berlin Heidelberg, 1985. ISBN 978-3-540-12761-1.

[149] Edward W. Washburn. **The Dynamics of Capillary Flow**. Journal of American Physical Society, Vol. 17(3), 1921. doi: https://doi.org/10.1103/PhysRev.17.273.

[150] W.M. Visscher and M. Bolsterli. **Random packing of equal and unequal spheres in two and three dimensions**. Nature, Vol. 239(5374):504–507, 1972. doi: https://doi.org/10.1038/239504a0.

[151] W.S. Jodrey and E.M. Tory. **Simulation of random packing of spheres**. Physical Review A, Vol. 32(1):1–12, 1979. doi: https://doi.org/10.1177/003754977903200102.

[152] W.S. Jodrey and E.M. Tory. **Computer simulation of close random packing of equal spheres**. Physical Review A, Vol. 32(4):2347–2351, 1985. doi: https://doi.org/10.1103/PhysRevA.32.2347.

[153] X. Liu, A. Mohammadian, and J. A. I. Sedano. **Three-dimensional modeling of non-hydrostatic free-surface flows on unstructured grids**. Internationnal Journal for Numerical Methods in Fluids, Vol. 82(3):130–147, 2015. doi: https://doi.org/10.1002/fld.4212.

[154] Y. Morinishi, T. S. Lund, O. V. Vasilyev, and P. Moin. **Fully Conservative Higher Order Finite Difference Schemes for Incompressible Flow**. Journal of Computational Physics, Vol. 143(1):90–124, 1997. doi: {https://doi.org/10.1006/jcph.1998.5962}.

[155] Y. Shi and Y. Zhang. **Simulation of random packing of spherical particles with different size distributions**. Journal of Applied Physics, Vol. 92:621–626, 2008. doi: https://doi.org/10.1007/s00339-008-4547-6.

[156] Z. Wang, J. Yang, B. Koo, and F. Stern. **A coupled level set and volume-of-fluid method for sharp interface simulation of plunging breaking waves**. International Journal of Multiphase Flow, Vol. 35(3):227–246, 2008. doi: https://doi.org/10.1016/j.ijmultiphaseflow.2008.11.004.

[157] Zsolt Horváth, Jürgen Waser, Rui A. P. Perdigão, Artem Konev, and Günter Blöschl. **A two–dimensional numerical scheme of dry/wet fronts for the Saint–Venant system of shallow water equations**. International Journal for Numerical Methods in Fluids, Vol. 77(3):159–186, 2014. doi: https://doi.org/10.1002/fld.3983.