CZECH TECHNICAL UNIVERSITY IN PRAGUE

Faculty Of Mechanical Engeneering

Department of Technical Mathematics

MASTER THESIS

NON-REFLECTIVE BOUNDARY CONDITIONS FOR FREE-SURFACE FLOWS

Author:	Bc. Josef Musil
Supervisor:	Doc. Ing. Jiří Fürst, Ph.D.
Study program:	Mechanical Engineering
Area of study:	Mathematical Modelling in Engineering
Academic year:	2017/2018

ČVUT
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

ZADÁNÍ DIPLOMOVÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení:	**Musil**	Jméno: **Josef**	Osobní číslo: **408595**
Fakulta/ústav:	**Fakulta strojní**		
Zadávající katedra/ústav:	**Ústav technické matematiky**		
Studijní program:	**Strojní inženýrství**		
Studijní obor:	**Matematické modelování v technice**		

II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce:

Bezodrazové okrajové podmínky pro simulace proudění s volnou hladinou

Název diplomové práce anglicky:

Non-reflective boundary conditions for free-surface flows

Pokyny pro vypracování:

Na základě analýzy rovnic popisujících proudění mělké vody odvoďte bezodrazové okrajové podmínky pro případ subkritického proudění. Na zvolené testovací úloze ověřte vlastnosti odvozených podmínek pro případ proudění mělké vody. Popište model dvoufázového proudění vody a vzduchu založený na objemových zlomcích tekutiny (tzv. volume of fluid model, VOF). Popište rozšíření dříve uvedených okrajových podmínek pro VOF model. Tuto podmínku implementujte v programovém prostředí balíku OpenFOAM a její vlastnosti ověřte pro případ simulace 2D a 3D proudění s volnou hladinou.

Seznam doporučené literatury:

Toro, E: Riemann Solvers and Numerical Methods for Fluid Dynamics
Weller, H et al.: A tensorial approach to computational continuum mechanics using object-oriented techniques

Jméno a pracoviště vedoucí(ho) diplomové práce:

doc. Ing. Jiří Fürst, Ph.D., ústav technické matematiky FS

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) diplomové práce:

Datum zadání diplomové práce: **18.04.2018** Termín odevzdání diplomové práce: **19.08.2018**

Platnost zadání diplomové práce: ______

[illegible]	[illegible]	[illegible]
doc. Ing. Jiří Fürst, Ph.D.	prof. Ing. Jaroslav Fořt, CSc.	prof. Ing. Michael Valášek, DrSc.
podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Diplomant bere na vědomí, že je povinen vypracovat diplomovou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v diplomové práci.

3. 5. 2018
Datum převzetí zadání

[illegible]
Podpis studenta

Declaration

Title: Non-reflective boundary conditions for free-surface flows

Abstract: In various fields of applications one is often interested in solving a fluid flow problem computationally in a domain which is much smaller than the actual domain where the governing equations hold. This approach is employed in order to reduce computational time and usually results in possibility to use more complex mathematical model of given physical phenomenon or solve given problem with higher accuracy or on finer grid. On the other hand, one natural problem arising in this context is connected with posing correct boundary conditions on artificial boundaries which are created by reducing the original domain. These boundary conditions have to be non-reflecting, namely they should not give rise to disturbances of the solution that propagates from within computational domain. The topic of non-reflective boundary conditions (also called absorbing or artificial boundary conditions) is subject of wide research activities, see [1], [2], [3], [4].

In this work some description of development of non-reflective boundary conditions for free surface flows described by shallow-water mathematical model is given at first. The main goal thereafter is the implementation of such boundary conditions in the framework of volume of fluid advection method (VOF) for Navier-Stokes equations in the finite-volume CFD toolbox - OpenFOAM. Derivation of such boundary conditions is based upon similarity between shallow water equations and Navier-Stokes equations for two-phase flows in certain flow regimes. Performance of the new boundary conditions is afterwards tested in OpenFOAM on 2D dam-break problem and 3D simulation of vertical water pump, placed in semi-opened basin with one artificial boundary.

Keywords: OpenFOAM, Volume Of Fluid, VOF, Shallow Water, Non-reflective boundary conditions, Finite Volume Method, Navier-Stokes equations.

Název práce: Bezodrazové okrajové podmínky pro proudění s volnou hladinou

Abstrakt: V některých současných inženýrských problémech, spjatých s dynamikou tekutin, narážíme na problém řešitelnosti výchozích rovnic na omezených oblastech. V takovém případě je nutné původní oblast, na které je daný matematický problém formulován, vhodným způsobem zredukovat. To má za následek snížení výpočetní náročnosti konkrétní úlohy a například umožňuje využít sofistikovanější matematické modely, nebo provést výpočet s vyšší přesností na jemnější síti. Spolu s tím ale vyvstává otázka jaké okrajové podmínky předepsat na nově vzniklých hranicích, které jsou nyní pouze artefaktem výpočetní domény a nemají fyzikální význam. Problematika zabývající se těmito, tzv. bez-odrazovými, podmínakmi pro různé typy problémů je v současné době předmětem mnoha výzkůmů, viz [1], [2], [3], [4].

V prezentované práci jsou na základě analýzy rovnic popisujících proudění mělké vody odvozeny bezodrazové okrajové podmínky pro případ subkritického proudění. Dále je popsán model dvoufázového proudění vody a vzduchu založený na objemových zlomcích tekutiny (tzv. volume of fluid model, VOF). Hlavním cílem práce je pak rozšíření bezodrazových okrajových podmínek pro mělkou vodu na VOF model. Tyto podmínky jsou následně implementovány v programovém prostředí softwarového balíku OpenFOAM. Vlastnosti daných okrajových podmínek jsou testovány a porovnávány na případech proudění s volnou hladinou, konkrétně 2D problému protržené hráze a 3D problému vodní pumpy v otevřené nádži.

Klíčová slova: OpenFOAM, objemové zlomky tekutiny, VOF, mělká voda, bez-odrazové okrajové podmínky, metoda konečných objemů, Navierovy-Stokesovy rovnice.

Dedication

I would like to thank my supervisor, Doc. Ing. Jiří Fürst, Ph.D. for patience, guidance, helpful insights during writing of this work and mostly for persistent support throughout my period of study. I also want to thank all my teachers who have enriched me with very valuable knowledge. Last, I would like to dedicate this work to my family, which is endlessly supporting me with my studies.

Contents

List of Figures

List of Tables

1. Introduction

The amount of engineering problems which could be simulated by computers is continually increasing due to the improvements in computational power and precising of description of physical phenomena solved by mathematical tools. In practice, majority of these problems are formulated in terms of partial differential equations (PDE), which are then solved numerically on computers. To solve such equations we need to determine computational domain, which corresponds with area of our interest. Some few typical examples could be: evaluating stress in mechanical part due to the acting forces on its surface, computing heat transfer through a body of given material or determining parameters of fluid flow in some channel. In this work, the further concern is devoted to the category of problems in which the last example belongs - computational fluid dynamics (CFD).

A frequently encountered problem in scientific computing is the design of artificial boundaries. The goal is to limit a computational domain to keep the number of cells within reasonable bounds yet still end up with a solution that approximates the correct result for an unbounded domain. This kind of boundaries, sometimes called absorbing or partially reflecting boundaries, allows disturbances generated within the solution domain to pass through the artificial boundary unhampered, while information from outside the solution domain is simultaneously specified to achieve the desired interior solution,
B. Sanders [2].

In the case of shallow water equations it means that there can be prescribed water height and wave velocity on the artificial boundary with water waves passing out unhampered. In some flow regimes, Navier-Stokes equations describe same physical phenomena as shallow water equations and therefore one can formulate boundary conditions motivated by theory of shallow water equations for corresponding flows solved by Navier-Stokes equations.

An original motivation of this work was simulation of flow in vertical water pump situated in semi-opened basin, practically a box with one missing sidewall partially immersed in water. And exactly this *opened* part of domain boundaries should be properly treated as artificial in the numerical simulation so there is a need to prescribe non-reflective boundary condition here.

At first, simulations of given probelm were performed in OpenFOAM software package [5]. However currently available boundary conditions led to reflecting water waves back to computational domain and due to further wave interactions, depending on geometry of the case, non-physical solution was obtained.

1.1 Structure of the thesis

Section 2 is devoted to description of how Navier-Stokes equations can be simplified into two-dimensional shallow water equations with consideration of some restrictions put on the flow. Further, transformation of one-dimensional version of shallow water equations into invariant form is presented.

In section 3 there are provided some details on prescription of boundary conditions for one-dimensional shallow water equations in general sense followed by derivation of one-dimensional non-reflective boundary conditions, see original article [2]. The section is finished with numerical results of one-dimensional dam-break problem.

Section 4 gives brief explanation of finite volume discretisation of two-dimensional shallow water equations along with implementation of one-dimensional non-reflective boundary conditions into two-dimensional case. In order to present performance of non-reflective boundary conditions in two dimensions results of numerical simulation are included.

Section 5 describes volume of fluid (VOF) method accompanied with two possible treatments of phase interface resolving implemented in OpenFOAM-v5.0 and OpenFOAM-v1712. Presented VOF model is concerned with two phase flow of a liquid-gas mixture, namely water and air.

Section 6 elaborates the transmission of non-reflective boundary conditions from one-dimensional shallow water theory to VOF method. The results of this effort are afterwards tested on two-dimensional dam-break and three-dimensional water pump problems, both solved by Navier-Stokes equations in VOF formulation in OpenFOAM software.

1.2 Main goals

The main goal of this work is to analyze shallow water equations and derive non-reflective boundary conditions for sub-critical regime of flow at first. This is followed by validation of these boundary conditions on 1D dam-break problem and its 2D modification. Subsequently, new non-reflective boundary conditions are developed for the VOF method using the similarity with shallow water theory under certain flow conditions. Performance of newly developed non-reflective boundary conditions is tested numerically on several cases. For the purpose of clarity, main goals can be outlined as follows:

- Analysis of shallow water equations in order to derive non-reflective boundary conditions
- Validation of these boundary conditions numerically
- Extension of the non-reflective boundary conditions from one-dimensioanl shallow water theory to the three-dimensional two-phase VOF method

2. Shallow water equations

2.1 Derivation

Many types of flow, not necessarily involving water as a fluid, can be characterized as shallow water flows. The general characteristic of such flows is that the vertical dimension is much smaller than any typical horizontal scale and this is true in many situations. Shallow water flows are nearly horizontal which allows a considerable simplification in the mathematical formulation and numerical solution by assuming the pressure distribution to be hydrostatic. However, they are not exactly two-dimensional. The flow exhibits a three-dimensional structure due to bottom friction, just as in boundary layers. Moreover, density stratification due to differences in temperature or salinity (in the case of modeling some coastal hydrodynamics) causes variations in the third (vertical) direction. Yet, for the purpose of this work, some simplifications of non-essential effects have been made. Depth varying quantities such as density and gravitational acceleration are considered independent of the z-coordinate. Further, water is considered to be inviscid fluid here, so surface tension or water-air interface friction is not taken into account [6].

By following previous assumptions, desired form of shallow water equations can be derived from Euler equations with appropriate boundary conditions. The following derivation procedure is motivated by [7].

momentum equation:

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} + \frac{1}{\rho} \nabla p = \mathbf{f} \qquad \text{on} \qquad -\eta(x,y) < z < h(x,y,t) \tag{2.1}$$

continutiy equation:

$$\nabla \cdot \mathbf{u} = 0 \qquad \text{on} \qquad -\eta(x,y) < z < h(x,y,t) \tag{2.2}$$

free surface condition:

$$p = 0, \quad \frac{\partial h}{\partial t} + \mathbf{u} \cdot \nabla h = w \qquad \text{on} \qquad z = h(x,y,t) \tag{2.3}$$

bottom condition:

$$\mathbf{u} \cdot \nabla [z + \eta(x,y)] = 0 \qquad \text{on} \qquad z = -\eta(x,y) \tag{2.4}$$

In the equations above, $\rho(x,y,t) = const.$ is density, $\mathbf{u} = (u,v,w)$ is three-dimensional velocity vector, p is pressure, $\mathbf{f} = (0,0,-g)$ body force density, g is gravitational acceleration, h the vertical displacement of free surface and $\eta(x,y)$ is the bottom topography, see Figure 2.1.

For the first step of the derivation of the shallow water equations the global mass conservation is considered. The continuity equation (2.2) is integrated vertically as follows,

$$0 = \int_{-\eta}^{h} [\nabla \cdot \mathbf{u}] \; dz = \int_{-\eta}^{h} \left[\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} \right] \; dz \tag{2.5}$$

$$\begin{aligned} &= \frac{\partial}{\partial x} \int_{-\eta}^{h} u \; dz - u\Big|_{z=h} \cdot \frac{\partial h}{\partial x} + u\Big|_{z=-\eta} \cdot \frac{\partial(-\eta)}{\partial x} \\ &+ \frac{\partial}{\partial y} \int_{-\eta}^{h} v \; dz - v\Big|_{z=h} \cdot \frac{\partial h}{\partial y} + v\Big|_{z=-\eta} \cdot \frac{\partial(-\eta)}{\partial y} \\ &+ w\Big|_{z=h} - w\Big|_{z=-\eta} \end{aligned} \tag{2.6}$$

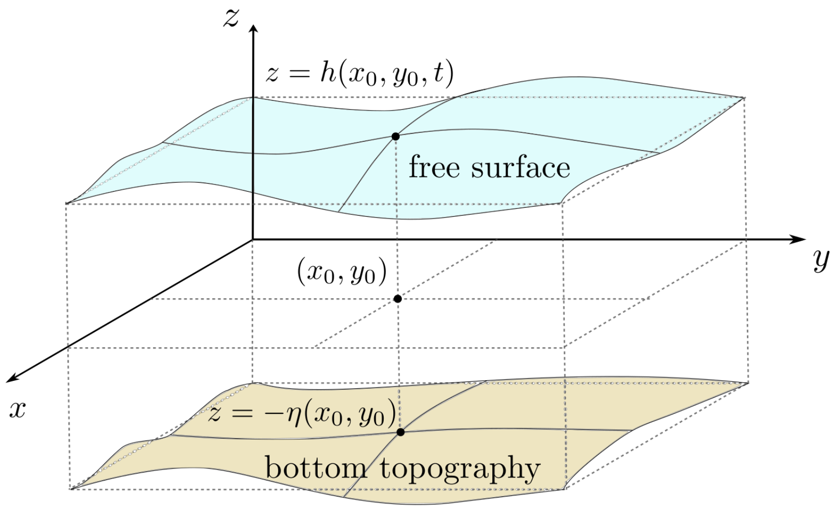


Figure 2.1: Schematic illustration of the Euler's system

where the bottom boundary condition (2.4) and the surface condition (2.3) were used in equation (2.6), so equation (2.5) becomes

$$\frac{\partial h}{\partial t} + \frac{\partial}{\partial x}\int_{-\eta}^{h} u \, dz + \frac{\partial}{\partial y}\int_{-\eta}^{h} v \, dz = 0 \tag{2.7}$$

In the next step long-wave approximation is made, by assuming that the wave length is much longer than the depth of the fluid. However, there is not assumed that perturbations have a small amplitudes, so that nonlinear terms are not neglected. Through the long-wave approximation, one can neglect the vertical acceleration term in (2.1), and deduce the hydrostatic pressure by integrating the vertical component of the momentum equation,

$$\int_{z_0}^{h} \frac{\partial p}{\partial z} \, dz = -\int_{z_0}^{h} \rho g \, dz \tag{2.8}$$

$$p(x, y, h(x, y, t), t) - p(x, y, z_0, t) = -\rho g(h(x, y, t) - z_0) \tag{2.9}$$

$$p(x, y, z, t) = \rho g(h(x, y, t) - z) \tag{2.10}$$

where the surface condition (2.3), $p(x, y, h, t) = 0$ was used and coordinate z_0 was chosen arbitrarily as z. Using this expression for the hydrostatic pressure (2.9) and further assuming that there are no vertical variations in velocities u, v, the horizontal momentum equations of the shallow-water system are obtained as follows,

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} + g\frac{\partial h}{\partial x} = 0 \tag{2.11}$$

$$\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} + g\frac{\partial h}{\partial y} = 0 \tag{2.12}$$

and conservation of mass given by (2.7) becomes

$$\frac{\partial h}{\partial t} + \frac{\partial}{\partial x}[(h + \eta)u] + \frac{\partial}{\partial y}[(h + \eta)v] = 0. \tag{2.13}$$

Then, equations (2.11), (2.12), and (2.13) are called the shallow water equations.

2.2 Mathematical structure

The original shallow water equations (2.11), (2.12) and (2.13), rewritten in more compact matrix form become,

$$\begin{pmatrix} h \\ u \\ v \end{pmatrix}_t + \begin{pmatrix} u & h+\eta & 0 \\ g & u & 0 \\ 0 & 0 & u \end{pmatrix} \begin{pmatrix} h \\ u \\ v \end{pmatrix}_x \begin{pmatrix} v & 0 & h+\eta \\ g & v & 0 \\ 0 & 0 & v \end{pmatrix} \begin{pmatrix} h \\ u \\ v \end{pmatrix}_y = \begin{pmatrix} -u\dfrac{\partial \eta}{\partial x} - v\dfrac{\partial \eta}{\partial y} \\ 0 \\ 0 \end{pmatrix} \tag{2.14}$$

or in symbolic notation

$$\mathbf{U}_t + \mathbb{A}(\mathbf{U})\mathbf{U}_x + \mathbb{B}(\mathbf{U})\mathbf{U}_y = \mathbf{S} \tag{2.15}$$

where lower subscripts (t, x, y) of algebraic vectors of original variables denotes associated partial derivatives. The eigenvalues of the coefficient matrix $\mathbb{A}$ are

$$u,\ u + \sqrt{g(h+\eta)},\ u - \sqrt{g(h+\eta)} \tag{2.16}$$

and those of the coefficient matrix $\mathbb{B}$ are

$$v,\ v + \sqrt{g(h+\eta)},\ v - \sqrt{g(h+\eta)}. \tag{2.17}$$

Since eigenvalues (2.16) and (2.17) are real and distinct, the shallow water equations are hyperbolic partial differential equations. The useful consequence of this is, that one can apply method of characteristics to find analytic form of solution, which will be later used for construction of desired non-reflective boundary conditions.

Note: By substituting term $h + \eta \to \tilde{h}$ in continuity equation (2.13), one gets after some algebraic manipulations following matrix form of shallow water equations:

$$\begin{pmatrix} \tilde{h} \\ u \\ v \end{pmatrix}_t + \begin{pmatrix} u & \tilde{h} & 0 \\ g & u & 0 \\ 0 & 0 & u \end{pmatrix} \begin{pmatrix} \tilde{h} \\ u \\ v \end{pmatrix}_x \begin{pmatrix} v & 0 & \tilde{h} \\ g & v & 0 \\ 0 & 0 & v \end{pmatrix} \begin{pmatrix} \tilde{h} \\ u \\ v \end{pmatrix}_y = \begin{pmatrix} 0 \\ g\eta_x \\ g\eta_y \end{pmatrix} \tag{2.18}$$

where now, $\tilde{h}$ is representing the distance between water surface and bottom and terms $g\eta_x$, $g\eta_y$ are acting in momentum equations as sources, dependent on the slope of bottom surface.

In order to keep further analysis easier, yet valuable, bottom topography will be considered flat and therefore $\eta(x, y)$ will be taken arbitrarily as zero. Moreover, let us for now consider only one-dimensional case. The two-dimensional wave structure and derivation of corresponding boundary conditions will be discussed later, in section 4. The one dimensional form shallow water equations with constant, flat bottom is given by matrix form as

$$\begin{pmatrix} h \\ u \end{pmatrix}_t + \begin{pmatrix} u & h \\ g & u \end{pmatrix} \begin{pmatrix} h \\ u \end{pmatrix}_x = \begin{pmatrix} 0 \\ 0 \end{pmatrix}. \tag{2.19}$$

Now method of characteristics can be used the to find out character of analytic solution. In order to use this method, one has to first transform the initial, coupled equations (2.19) into set of two independent ones. This is done as follows. Let us start with compact matrix form of equations (2.19)

$$\mathbf{U}_t + \mathbb{A}(\mathbf{U})\mathbf{U}_x = \mathbf{0}, \tag{2.20}$$

where now $\mathbf{U} = (h, u)^T$ is algebraic vector of original variables h and u and $\mathbb{A}(\mathbf{U})$ is coefficient matrix. Spectral decomposition of $\mathbb{A}(\mathbf{U})$ is now performed, with resulting terms

$$\mathbb{A} = \begin{pmatrix} u & h \\ g & u \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ \sqrt{\frac{g}{h}} & -\sqrt{\frac{g}{h}} \end{pmatrix} \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix} \frac{-1}{2\sqrt{\frac{g}{h}}} \begin{pmatrix} -\sqrt{\frac{g}{h}} & -1 \\ -\sqrt{\frac{g}{h}} & 1 \end{pmatrix} = \mathbb{R}\mathbb{D}\mathbb{R}^{-1} \tag{2.21}$$

where $\mathbb{D}$ is diagonal matrix of eigenvalues $\lambda_1 = u + \sqrt{gh}$ and $\lambda_2 = u - \sqrt{gh}$ corresponding with eigenvectors $\mathbf{V}_{\lambda_1} = (1,\ \sqrt{\frac{g}{h}})$ and $\mathbf{V}_{\lambda_2} = (1,\ -\sqrt{\frac{g}{h}})$ respectively of which, as columns, is assembled matrix $\mathbb{R} = (\mathbf{V}_{\lambda_1} | \mathbf{V}_{\lambda_2})$ resp. its inverse $\mathbb{R}^{-1}$. With using terms in (2.21), shallow water equations (2.19) might be written in expanded form as

$$\frac{-1}{2\sqrt{\frac{g}{h}}}(-\sqrt{\tfrac{g}{h}}h_t - u_t) + (\lambda_1) \cdot \frac{-1}{2\sqrt{\frac{g}{h}}}(-\sqrt{\tfrac{g}{h}}h_x - u_x) = 0 \tag{2.22}$$

$$\frac{-1}{2\sqrt{\frac{g}{h}}}(-\sqrt{\tfrac{g}{h}}h_t + u_t) + (\lambda_2) \cdot \frac{-1}{2\sqrt{\frac{g}{h}}}(-\sqrt{\tfrac{g}{h}}h_x + u_x) = 0 \tag{2.23}$$

and by some algebraic manipulations, assuming $h \neq 0$, with using following formula

$$\frac{1}{\sqrt{h}} \frac{\partial h}{\partial \zeta} = \frac{\partial (2\sqrt{h})}{\partial \zeta}, \qquad \zeta \in \{t, x\} \tag{2.24}$$

final results of transformation of original equations (2.19) are obtained

$$(u + 2\sqrt{gh})_t + (u + \sqrt{gh}) \cdot (u + 2\sqrt{gh})_x = 0 \tag{2.25}$$

$$(u - 2\sqrt{gh})_t + (u - \sqrt{gh}) \cdot (u - 2\sqrt{gh})_x = 0. \tag{2.26}$$

Or, in more compact form

$$R_1(u, h)_t + \lambda_1 \cdot R_1(u, h)_x = 0 \tag{2.27}$$

$$R_2(u, h)_t + \lambda_2 \cdot R_2(u, h)_x = 0 \tag{2.28}$$

where

$$R_1(u(x, t), h(x, t)) = u + 2\sqrt{gh} \tag{2.29}$$

$$R_2(u(x, t), h(x, t)) = u - 2\sqrt{gh}. \tag{2.30}$$

One can see that original, coupled shallow water equations are now separated into two hyperbolic partial differential equations, which represent transport of two artificial variables R_1, R_2 depending on original ones. These new variables are often called *Riemann variables* or *Riemann invariants*. Eigenvalues λ_1, λ_2 represent characteristic advection speeds. Further, method of characteristics provides following analytic form of solution for each of equations (2.27) and (2.28), given by

$$R_i(x, t) = R_i^0 \left(x - \int_0^t \lambda_i(\xi_i(\tau), \tau)\, d\tau \right), \qquad i = 1, 2 \tag{2.31}$$

where $R_i^0(x) = R_i(x, 0)$ is initial condition. Basically, it is a case of transport of initial conditions along characteristic curves $\xi_1(t)$ and $\xi_2(t)$ in (x, t)-plane, defined by

$$\frac{d\xi_{1,2}}{dt} = \lambda_{1,2}(\xi_{1,2}(t), t) = u \pm \sqrt{gh} \tag{2.32}$$

Solution of the problem in terms of original variables $h(x, t)$ and $u(x, t)$ is given as combination of Riemann invariants, see original article [2],

$$u = \frac{1}{2}(R_1 + R_2) \qquad h = \frac{1}{16g}(R_1 - R_2)^2 \tag{2.33}$$

which will serve as initial point for derivation of non-reflective boundary conditions.

3. Non-reflective boundary conditions for 1-dimensional shallow-water equations

In this part a problematics of well-posedness of boundary conditions will be elaborated, with a focus on shallow water equations. Moreover, an example of boundary condition which is ill-posed, yet still used in numeric computation, will be described. After that derivation of non-reflective boundary conditions will follow, provided with numerical results. Following description and demonstration of non-reflective boundary conditions will be done using one-dimensional Riemann problem which consists of one-dimensional shallow water equations and initial piece-wise constant data with a single discontinuity, see Fig.(3.1). On the right boundary ($x = l$), which is not a point of interest here, there is considered wall. On the left boundary ($x = 0$) the performance of non-reflective boundary condition will be examined. With respect to shallow water problematics carried here, this case may be viewed as a dam-break problem.

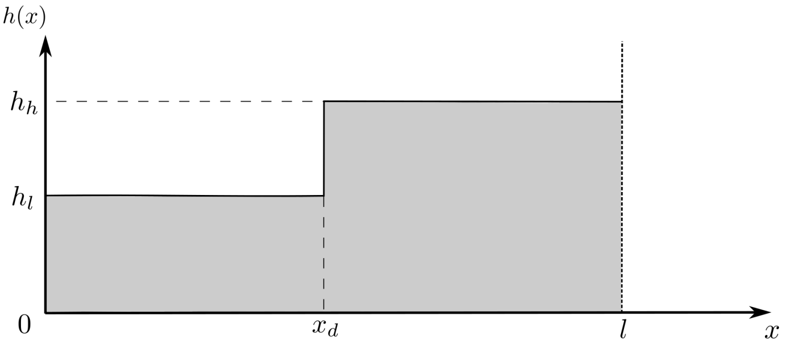


Figure 3.1: Sketch of one-dimensional Riemann problem

3.1 Well-posed boundary conditions

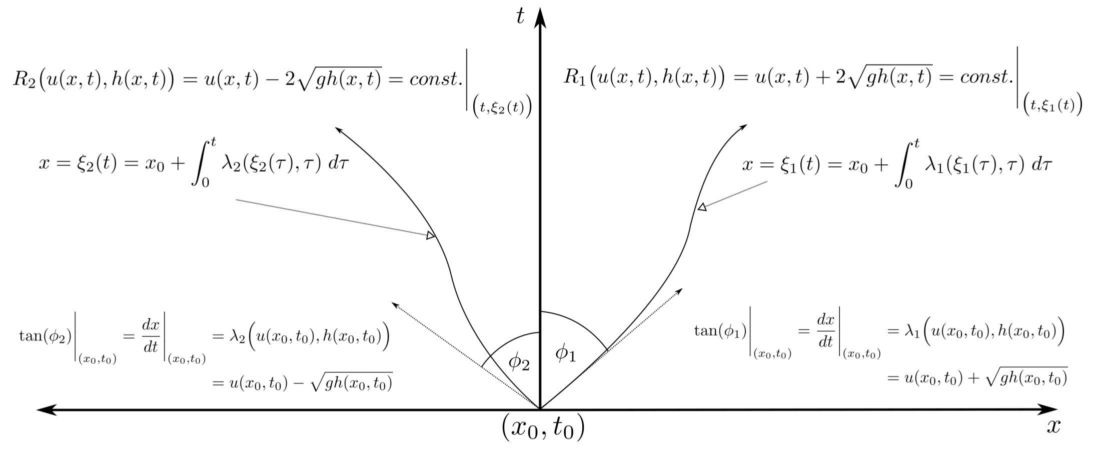
Well-posednes means, by definition, that a problem has a solution, this solution is unique and depends continuously on the initial data and the parameters (including boundary conditions). Therefore, well-posed boundary conditions are the essential ingredient in many areas of computational physics. In order to solve any physical problem described by partial differential equations correctly in a analytic sense, one needs to know: *i*) how many boundary conditions are required, *ii*) where to impose them and *iii*) which form should they have [8]. Generally, this may be very hard task and it is a subject of wide research activities. More on this topic can be found for example in [9].

Therefore one-dimensional shallow water equations (2.19) are 1^{st} order hyperbolic partial differential equations, the well posedness of aforementioned Riemann problem, in the sense of boundary conditions, depends on the flow regime. The regime of flow in a given space and time is dependent on characteristic advection speeds, λ_1 and λ_2 in a such way that if $\lambda_1 \cdot \lambda_2 < 0$ the regime is called sub-critical and otherwise super-critical. The key issue on the boundaries of the one-dimensional computational domain of dam break problem ($x = 0$ and $x = l$) is then whether characteristics are going in or out of the domain. On the Figure 3.2 are shown possible configurations of characteristic curves in (x, t)-plane for an arbitrary point (x_0, t_0). Throughout following text attention will be paid to the left boundary, so let us assume $x_0 = 0$.

The first case where characteristics are going the opposite directions is illustrated on Figure 3.2a. This situation means, that Riemann variable R_2 is transported out of the computational domain, and so there is no need to prescribe a boundary condition for it. The second Riemann variable R_1 is transported in the domain and therefore should be prescribed on the boundary. It means that there must be provided some combination of $u(x_0, t) = u_0(t)$ and $h(x_0, t) = h_0(t)$ to get desired value of R_1.

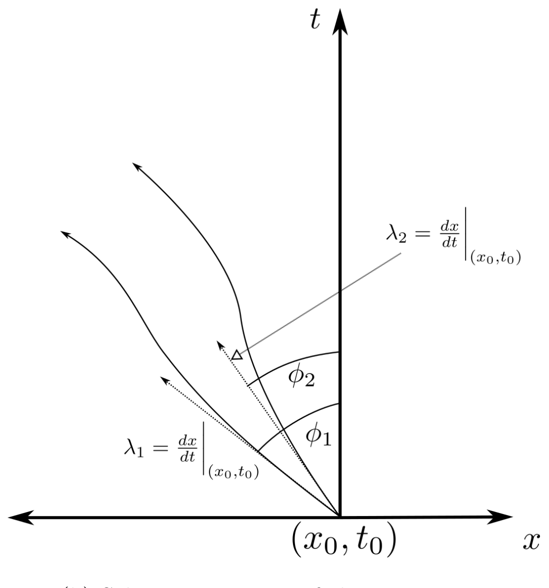
In the second case on Figure 3.2b both characteristics go out of the domain, so no information about the solution is transported inside and therefore no boundary conditions are required in this case.

The last case is shown on Figure 3.2b. This case is the exact opposite to the previous one. Both characteristics go inside and one must specify boundary conditions for both variables R_1 and R_2.



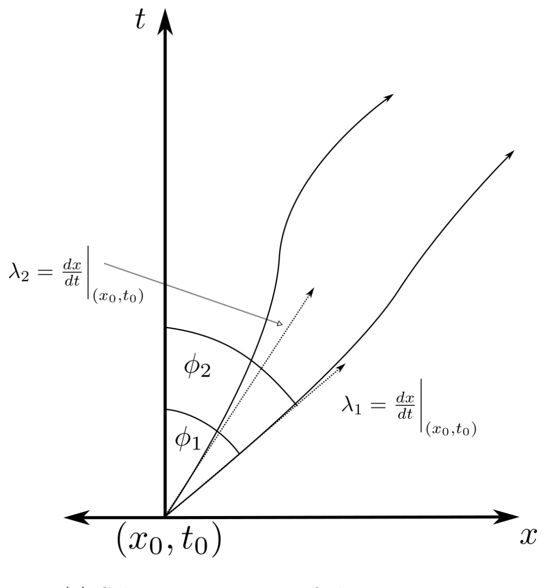
t
$R_2\big(u(x,t),h(x,t)\big) = u(x,t) - 2\sqrt{gh(x,t)} = const.\Big|_{(t,\xi_2(t))}$
$R_1\big(u(x,t),h(x,t)\big) = u(x,t) + 2\sqrt{gh(x,t)} = const.\Big|_{(t,\xi_1(t))}$
$x = \xi_2(t) = x_0 + \int_0^t \lambda_2(\xi_2(\tau),\tau)\,d\tau$
$x = \xi_1(t) = x_0 + \int_0^t \lambda_1(\xi_1(\tau),\tau)\,d\tau$
$\tan(\phi_2)\Big|_{(x_0,t_0)} = \frac{dx}{dt}\Big|_{(x_0,t_0)} = \lambda_2\Big(u(x_0,t_0),h(x_0,t_0)\Big) = u(x_0,t_0) - \sqrt{gh(x_0,t_0)}$
$\tan(\phi_1)\Big|_{(x_0,t_0)} = \frac{dx}{dt}\Big|_{(x_0,t_0)} = \lambda_1\Big(u(x_0,t_0),h(x_0,t_0)\Big) = u(x_0,t_0) + \sqrt{gh(x_0,t_0)}$
ϕ_2
ϕ_1
(x_0,t_0)
x

(a) Schematic picture of characteristics, $\boldsymbol{\lambda_1} \cdot \boldsymbol{\lambda_2} < 0$



t
$\lambda_2 = \frac{dx}{dt}\Big|_{(x_0,t_0)}$
ϕ_2
ϕ_1
$\lambda_1 = \frac{dx}{dt}\Big|_{(x_0,t_0)}$
(x_0,t_0)
x

(b) Schematic picture of characteristics, $\boldsymbol{\lambda_1} \cdot \boldsymbol{\lambda_2} \geq 0$, $\lambda_1 < 0$



t
$\lambda_2 = \frac{dx}{dt}\Big|_{(x_0,t_0)}$
ϕ_2
ϕ_1
$\lambda_1 = \frac{dx}{dt}\Big|_{(x_0,t_0)}$
(x_0,t_0)
x

(c) Schematic picture of characteristics, $\boldsymbol{\lambda_1} \cdot \boldsymbol{\lambda_2} \geq 0$, $\lambda_1 > 0$

Figure 3.2: Characteristic curves on left boundary

3.2 An example of Ill-posed boundary condition

In the ideal case boundary conditions should be derived in correspondence with an analytic form of exact solution of given partial differential equations, specifically shallow water equations in our case. But, as was mentioned in section 1 solving partial differential equations is usually connected with some physical phenomenon and therefore requirements on boundary conditions respecting physical nature of given problem are sometimes difficult to fulfill in strictly analytic sense.

For example let us consider a part of a boundary as a wall, let us say $x = 0$. The prescription for velocity is rather simple $u(0,t) = u_0(t) = 0$, but there is a problem how to determine water height on this boundary. In numerical computing there is a crafty option based on physics. If some contact angle between water and wall is considered, then value on the boundary could be extrapolated from interior of computational domain. This corresponds with prescription of Neumann boundary condition, $\frac{\partial h}{\partial \vec{n}} = \alpha = \tan(90^\circ - \theta)$, where θ is a contact-angle, usually set as 90° (no surface tension), resulting with homogeneous Neumann condition.

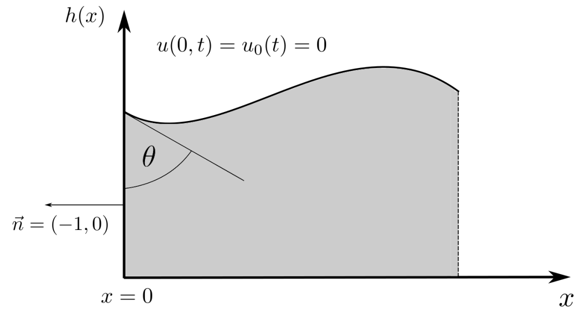


Figure 3.3: Neumann boundary condition for $h(x,t)$

This boundary condition works quite well in numerical simulations, but due to the fact that shallow water equations are 1st order, using any kind of boundary condition which involves first derivative is mathematically incorrect.

3.3 Derivation of non-reflective boundary conditions

In order to describe the derivation process as well as present numerical results, there will be considered 1-dimensional shallow water problem governed by equations (2.19). This problem will be solved on following domain

$$\Omega = \{x \in \mathbb{R},\ 0 < x < 1\} \tag{3.1}$$

$$\partial\Omega = \{0, 1\} \tag{3.2}$$

with the initial conditions

$$u = 0, \qquad x \in \langle 0, 1\rangle \tag{3.3}$$

$$h = \begin{cases} 0.5, & 0 \le x \le 0.4 \\ 0.7, & 0.4 < x \le 1 \end{cases} \tag{3.4}$$

and the above mentioned wall boundary conditions prescribed for $x = 1$ and below derived non-reflective boundary conditions prescribed for $x = 0$. Discretisation of the problem for numerical computing will be done by finite volumes method (FVM) described as follows.

- discretised computational domain:

$$\Omega = \bigcup_{i \in \mathbb{N}} \Omega_i \tag{3.5}$$

$$\Omega_i = \left\{ x \in \left\langle x_i - \frac{\Delta x}{2}, x_i + \frac{\Delta x}{2} \right\rangle,\ x_i = (i - \tfrac{1}{2}) \cdot \Delta x \right\} \tag{3.6}$$

$$x_0 = x_1 - \tfrac{\Delta x}{2} = 0 \tag{3.7}$$

- discrete time:

$$t^n = n \cdot \Delta t, n \in \mathbb{N}_0,\ \Delta t \in \mathbb{R}^+ \tag{3.8}$$

- Notation of any physical quantity $\phi(x,t)$ in discrete space and time is:

$$\phi_i^n = \frac{1}{\|\Omega_i\|} \int_{\Omega_i} \phi(x, t^n) \tag{3.9}$$

3.3.1 Derivation

Implementation of non-reflecting boundary conditions into the framework of FVM for shallow water equations is done by discretisation of terms in equation (2.33). Following Figure (3.4) describes situation on the left boundary for analytic solution of equations (2.19) evolving in time. Regime of the flow is considered subcritical ($\lambda_1 > 0$ and $\lambda_2 < 0$).

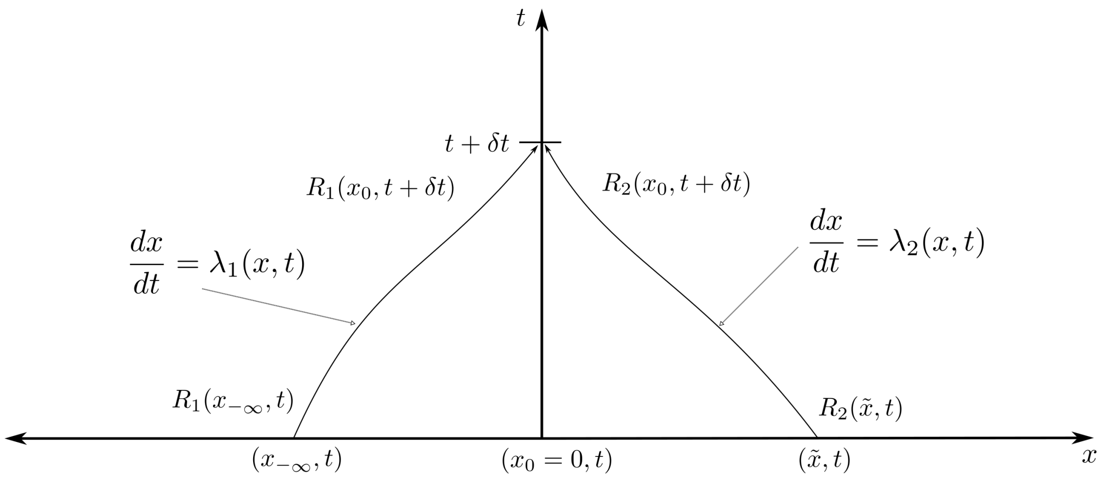


Figure 3.4: Transport of Riemann variables to the left boundary ($x = 0$)

Outside the computational domain ($x < x_0$) is considered auxiliary Riemann invariant (R_1). It depends on water height and wave velocity which are here considered to satisfy calm water surface conditions. So $h(x_{-\infty}, t) = h_{-\infty}$ is constant in time and $u(x_{-\infty}, t) = u_{-\infty} = 0$ for all $x = x_{-\infty} < x_0$. The result of these assumptions is that R_1 is also constant in time and thus during numerical simulation.

$$R_1(x_{-\infty}, t) = R_1(x_0, t + \delta t) = u_{-\infty} + 2\sqrt{gh_{-\infty}} = 2\sqrt{gh_{-\infty}} \tag{3.10}$$

Dealing with the second invariant which is coming to the boundary from inside the computational domain requires some extra treatment. Let us remind the relationship for analytic form of solution of one-dimensional shallow water equations:

$$R_i(x,t) = R_i^0 \left(x - \int_0^t \lambda_i(\xi_i(\tau), \tau)\, d\tau \right), \qquad i = 1, 2 \tag{3.11}$$

$$\frac{d\xi_{1,2}}{dt} = \lambda_{1,2}(\xi_{1,2}(t), t) = u \pm \sqrt{gh} \tag{3.12}$$

Problem is, that equations (3.11) and (3.12) are implicitly coupled. The goal is to determine $R_2(x_0, t + \delta t)$ which is transported along the characteristic curve defined by

$\dfrac{dx}{dt} = \lambda_2(x,t)$. The transport is here considered from initial point $(\tilde{x}, t)$ to $(x_0, t+\delta t)$, so in equation (3.11) is initial condition $R_i^0(x) = R_i(x, t=0)$ replaced by $R_i(x,t)$ and integration limits are changed as follows $(0,t) \to (t, t+\delta t)$ which gives us for R_2 following results

$$R_2(x_0, t+\delta t) = R_2(\tilde{x}, t) = R_2\left(x_0 - \int_t^{t+\delta t} \lambda_2(\xi_2(\tau), \tau)\, d\tau,\ t\right) \tag{3.13}$$

$$\tilde{x} = x_0 - \int_t^{t+\delta t} \lambda_2(\xi_2(\tau), \tau)\, d\tau \tag{3.14}$$

where $\tilde{x}$ is yet unknown point, implicitly determined to satisfy the condition so that characteristic curve is going through this point in time t will reach point x_0 in $t+\delta t$.

There is considered $\lambda_2(x,t)$ (and therefore also λ_1) to be continuous which means that no shock waves are present in the solution. Therefore Lagrange's mean value theorem in equation (3.14) can be used

$$\tilde{x} = x_0 - \int_t^{t+\delta t} \lambda_2(\tilde{x}, \xi)\, d\xi = x_0 - \lambda_2(\tilde{x}^*, t^*) \cdot \delta t \tag{3.15}$$

where $x^* \in (x_0,\ \tilde{x})$ and $t^* \in (t,\ t+\delta t)$.

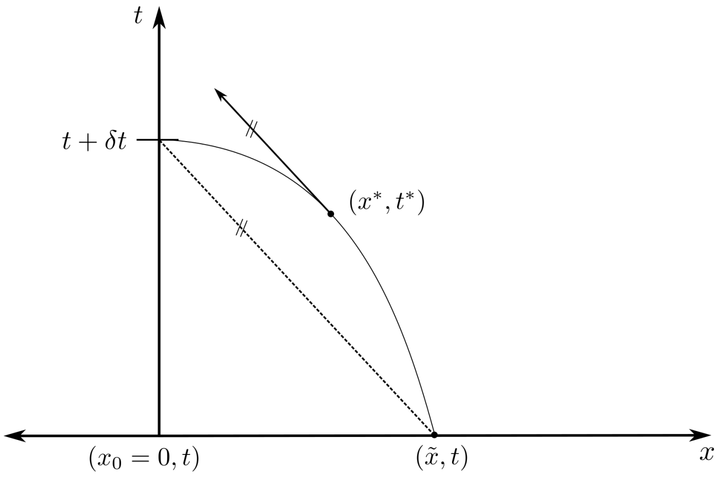


Figure 3.5: Scheme of evaluating $\tilde{x}$ in equation 3.15 using Lagrange's mean value theorem

3.3.2 Final formulas

In numerical simulations of unsteady problems there is a requirement for time step to be small enough that physical quantities don't change rapidly. It means in practice that so called CFL condition have to be satisfied. It implies that for one-dimensional shallow water equations following condition has to hold.

$$C = \max|\lambda_i(x,t)|\frac{\Delta t}{\Delta x} \leq C_{max} \quad \forall x \in \Omega,\ i = 1,2. \tag{3.16}$$

Here $C_{max} < 1$ is used for explicit numerical methods and $C_{max} \geq 1$ for implicit methods could be tolerated.

Since in numerical validation here is supposed that $C_{max} < 1$, it can be shown that $\tilde{x} \in (x_0, \Delta x)$. Finally, time step δt is replaced by numerical time step Δt, resulting in following approximation

$$R_2(x_0, t+\Delta t) = R_2(\tilde{x}, t) = R_2(x_1, t) + \mathcal{O}(\Delta x) \tag{3.17}$$

where $x_1 = \Delta x/2$ (from equation (3.6)). In the end here are included final terms for non-reflective boundary conditions which were implemented in numerical code

$$R_1(x_0, t^{n+1}) = 2\sqrt{gh_{-\infty}} \tag{3.18}$$
$$R_2(x_0, t^{n+1}) = R_2(x_1, t^n) \tag{3.19}$$

$$h_0^{n+1} = \frac{1}{16g}\Big(R_1 - R_2\Big)^2 = \frac{1}{16g}\Big(2\sqrt{gh_{-\infty}} - u_1^n + 2\sqrt{gh_1^n}\Big) \tag{3.20}$$
$$u_0^{n+1} = \frac{R_1 + R_2}{2} = \frac{1}{2}\Big(2\sqrt{gh_{-\infty}} + u_1^n - 2\sqrt{gh_1^n}\Big) \tag{3.21}$$

where only $h_{-\infty}$ will be prescribed as a parameter according to the calm water surface given by initial condition.

3.4 Numerical results

Performance of non-reflective boundary conditions was tested on 1D dam-break simulation together with two other boundary conditions as a reference benchmark. The combination of Dirichlet condition for the velocity, $u = 0$, and Neumann homogeneous condition for the volume fraction, $\frac{\partial h}{\partial \vec{n}} = 0$, was prescribed at the right boundary (wall) whereas on the left boundary the conditions were set according to following table (3.1),

Table 3.1: Setting of boundary conditions in dam-break problem

simulation No.	I.	II.	III.
boundary conditions for $x = 0$	$h(0,t) = h_0$, $\frac{\partial u}{\partial \vec{n}} = 0$	$u(0,t) = u_0$, $\frac{\partial h}{\partial \vec{n}} = 0$	non-reflective

where $h_0 = 0.5$, $u_0 = 0$ and non-reflective conditions according equations (3.20) and (3.21), with $h_{-\infty} = 0.5$. Initial conditions are the same for all cases. Water height, h, is matching with Figure 3.6 and u is taken as zero. In all cases, the numerical solution of equations (2.19) was obtained using the explicit Euler method in time and the first order finite volume scheme with the HLL numerical flux [10]. Computational domain was divided equidistantly with $\Delta x = 0.001$ and CFL condition was chosen as $C = 0.4$.

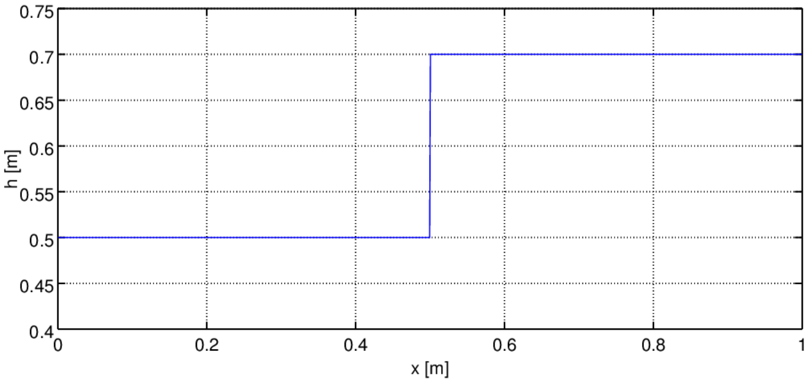
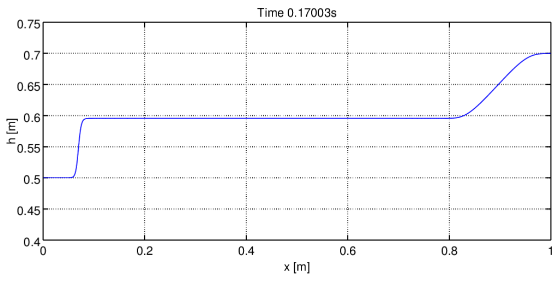
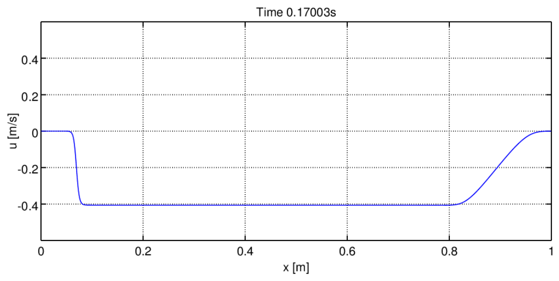


Figure 3.6: Initial condition of 1D dam-break problem, water height

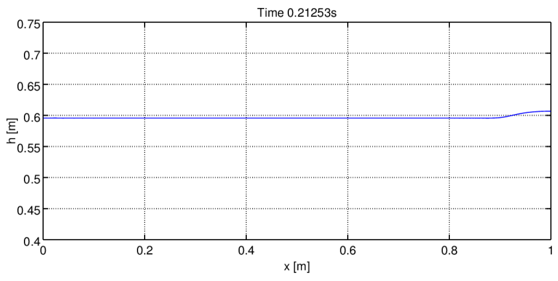


(a) $h(x,t)$

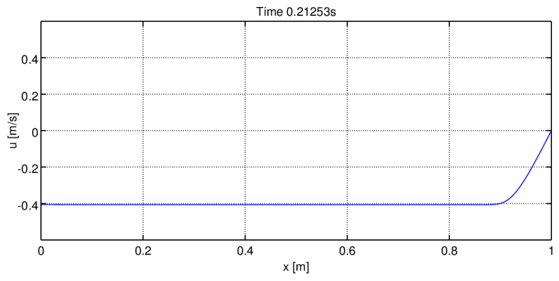


(b) $u(x,t)$

Figure 3.7: Water height and velocity, water wave before approaching left boundary, the same for all boundary conditions

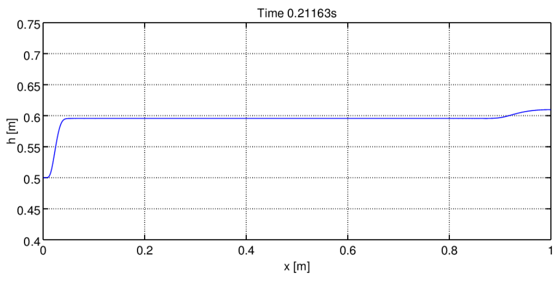


(a) $h(x,t)$

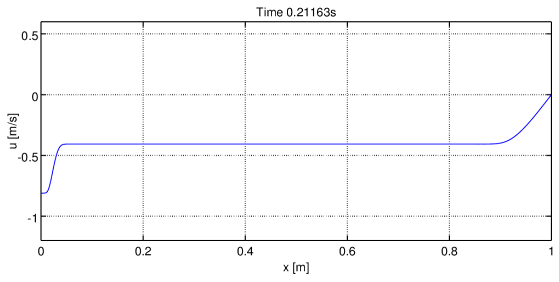


(b) $u(x,t)$

Figure 3.8: Water height and velocity, water wave reaching left boundary, non-reflective boundary conditions

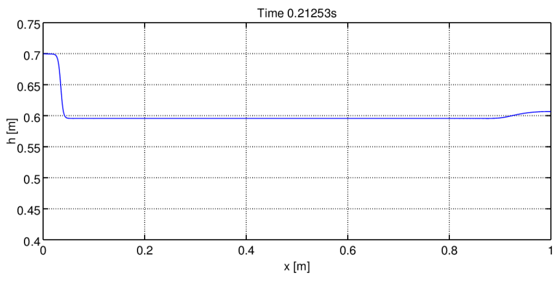


(a) $h(x,t)$

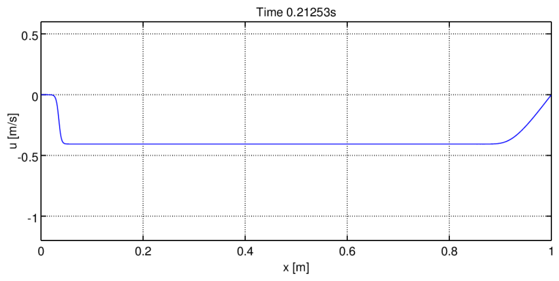


(b) $u(x,t)$

Figure 3.9: Water height and velocity, water wave reaching left boundary, Dirichlet condition for $h(x,t)$ and Neumann homogeneous condition for $u(x,t)$



(a) $h(x,t)$



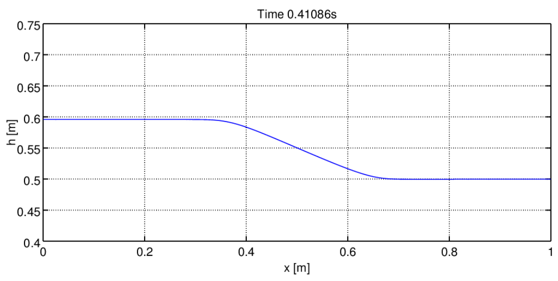
(b) $u(x,t)$

Figure 3.10: Water height and velocity, water wave reaching left boundary, Dirichlet condition for $u(x,t)$ and Neumann homogeneous condition for $h(x,t)$

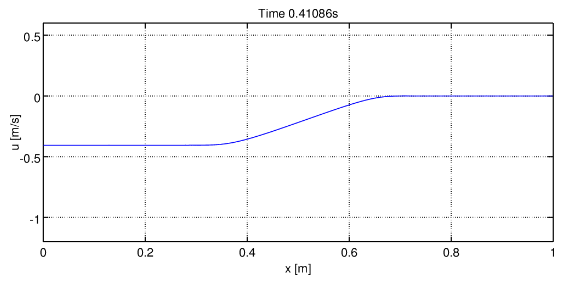
The result of comparison of Figures 3.7 and 3.8 is that both velocity and water height disturbances passed left boundary smoothly, without notable reflections. Although some minor reflections cloud occur due to the boundary condition and numerical flux schemes being first order of spatial accuracy.

On the other hand on Figures 3.9 and 3.10 is shown that simple combinations of Dirichlet and Neumann conditions cause strong reflections. Either velocity or water height is reflected as a result of one variable is always prescribed constant and thus not reply for incoming disturbances from within computational domain.

The boundary condition on the right boundary represents physical wall. Despite the fact it is ill-posed, the performance in numerical simulation is sufficient for this purpose.

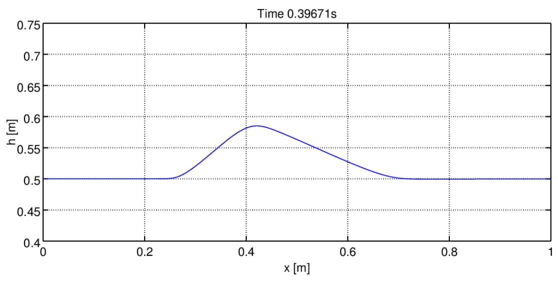


(a) $h(x,t)$

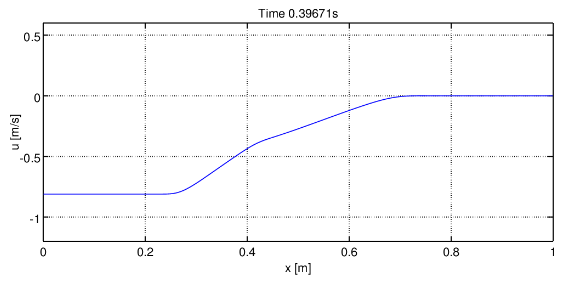


(b) $u(x,t)$

Figure 3.11: Water height and velocity, water wave reflected form right wall now reaching left boundary, non-reflective boundary conditions

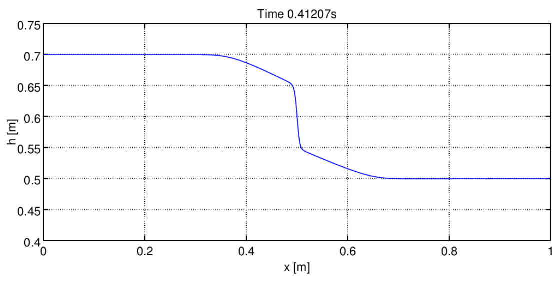


(a) $h(x,t)$

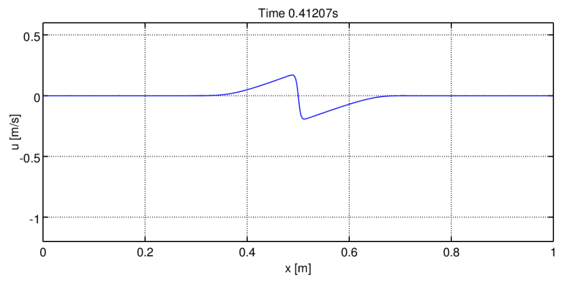


(b) $u(x,t)$

Figure 3.12: Water height and velocity, water height reflected form left boundary, wave interaction in the middle of domain, Dirichlet condition for $h(x,t)$ and Neumann homogeneous condition for $u(x,t)$



(a) $h(x,t)$



(b) $u(x,t)$

Figure 3.13: Water height and velocity, water velocity reflected form left boundary, wave interaction in the middle of domain, Dirichlet condition for $u(x,t)$ and Neumann homogeneous condition for $h(x,t)$

In the series of Figures 3.11, 3.12, 3.12 there is depicted solution of dam-break problem in the moment when reflections from left boundary, if present, reached approximately the middle of domain. The comparison between Figure 3.11 and Figures 3.12 and 3.12 shows that combination of Dirichlet and Neumann condition prescribed on the left boundary not only caused some reflections in the vicinity of the boundary, but spoiled entire solution. In this example, due to the initial condition, reflections are considerably large but even smaller ones could lead to nonphysical solution. Importance of this issue will be discussed in section 6 in connection with 3D volume-of-fluid water pump simulation.

4. Non-reflective boundary conditions for 2-dimensional shallow-water equations

In this part is discussed an extension of previous one-dimensional non-reflective boundary conditions into two-dimensional shallow water mathematical model. The whole procedure is based on work of B.Sanders, described in article [2].

4.1 Finite volume method for 2D shallow-water equations

At the beginning let us perform some formal manipulations with equation (2.14). At first an integration of this equation through computation domain is done

$$\frac{\partial}{\partial t}\int_\Omega \mathbf{U}\, d\Omega + \int_\Omega \nabla\cdot\Big(\mathbf{F},\mathbf{G}\Big)\, d\Omega = \int_\Omega \mathbf{S}\, d\Omega \tag{4.1}$$

where $\Omega \subset \mathbf{E}_2$ is compact and has a piecewise smooth boundary $\partial\Omega$ and

$$\mathbf{U} = \begin{pmatrix} h \\ u \\ v \end{pmatrix}, \quad \mathbf{F} = \begin{pmatrix} uh \\ u^2h + \frac{gh^2}{2} \\ uvh \end{pmatrix}, \quad \mathbf{G} = \begin{pmatrix} vh \\ uvh \\ v^2h + \frac{gh^2}{2} \end{pmatrix}, \quad \mathbf{S} = \begin{pmatrix} -u\dfrac{\partial\eta}{\partial x} - v\dfrac{\partial\eta}{\partial y} \\ 0 \\ 0 \end{pmatrix}. \tag{4.2}$$

In following proceeding there is considered flat bottom topography, so that $\mathbf{S} = \mathbf{0}$. Numerical discretisation of computational domain is done by Cartesian grid

$$\Omega = \bigcup_{i,j\in\mathbb{N}_0} \Omega_{i,j}, \quad \Omega_{i,j}\cap\Omega_{k,l} = \emptyset \quad \forall i \neq k, j\neq l, \quad i,j,k,l \in \mathbb{N}_0, \tag{4.3}$$

$$\Omega_{i,j} = \Big\{[x,y]\in\mathbf{E}_2;\ i\Delta x < x < (i+1)\Delta x;\ j\Delta y < y < (j+1)\Delta y\Big\}. \tag{4.4}$$

where $\Delta x,\ \Delta y > 0$. Discretisation of time is straightforward.

$$t^n := n\Delta t, \quad n\in\mathbb{N}_0, \quad \Delta t > 0 \tag{4.5}$$

Now, several procedures are done with equation (4.1). There is used the Gauss's divergence theorem to rewrite the second term on the left side as a surface integral. Time derivation is substituted by Crank–Nicolson discretisation method. Following notation is employed

$$\mathbf{U}^n_{i,j} = \frac{1}{\|\Omega_{i,j}\|}\int_{\Omega_{i,j}} \mathbf{U}(x,y,t^n)\, d\Omega \tag{4.6}$$

$$\|\Omega_{i,j}\| = \Delta x\Delta y \tag{4.7}$$

The equation (4.1) can be now rewritten as

$$\begin{aligned}\mathbf{U}^{n+1}_{i,j} = \mathbf{U}^n_{i,j} &+ \Delta t\cdot\frac{1}{2}\Big[\Big(\mathbf{F}^{n+1}_{i+1/2,j} + \mathbf{F}^n_{i+1/2,j}\Big) - \Big(\mathbf{F}^{n+1}_{i-1/2,j} + \mathbf{F}^n_{i-1/2,j}\Big)\Big]\Delta y \\ &+ \Delta t\cdot\frac{1}{2}\Big[\Big(\mathbf{G}^{n+1}_{i,j+1/2} + \mathbf{G}^n_{i,j+1/2}\Big) - \Big(\mathbf{G}^{n+1}_{i,j-1/2} + \mathbf{G}^n_{i,j-1/2}\Big)\Big]\Delta x\end{aligned} \tag{4.8}$$

where $\mathbf{F}^{n+1}_{i+1/2,j} = \mathbf{F}(\mathbf{U}^n_{i,j}, \mathbf{U}^n_{i+1,j})$ denotes numerical flux evaluated on boundary between volumes $\Omega_{i,j}$ and $\Omega_{i+1,j}$ in time t^n. The same rule applied to other flux symbols is an analogy.

4.2 Implementation of 2D non-reflective boundary condition

In spite of fact that method of characteristics can not be generalized into 2D shallow water equations [7], there is an alternative of using modified version of 1D boundary conditions. Proposed modification is quite simple. The internal velocity (u_1^n) in equations (3.20) and (3.20) is substituted by normal component of 2D velocity vector. Tangent component of velocity vector is neglected here (taken as $\mathbf{0}$ at boundary), due to the *upwind nature* of scheme used for computing numerical flux. Treatment of water height is the same for both 1D and 2D case. Following terms summarize implementation of 2D non-reflecting boundary condition.

$$\mathbf{u}_b^{n+1} = \frac{1}{2}\Big(2\sqrt{gh_\infty} - (\mathbf{u}_{b+1}^n \cdot \mathbf{n}) - 2\sqrt{gh_{b+1}^n}\Big)(-\mathbf{n}) \tag{4.9}$$

$$h_b^{n+1} = \frac{1}{16g}\Big(2\sqrt{gh_\infty} + (\mathbf{u}_{b+1}^n \cdot \mathbf{n}) + 2\sqrt{gh_{b+1}^n}\Big) \tag{4.10}$$

where subscripts b and $b+1$ denote values at boundary and boundary adjacent cells respectively. Two dimensional velocity is denoted as $\mathbf{u} = (u, v)$, with all belonging indices. Outer-pointing normal at boundary is expressed as $\mathbf{n} = (n_x, n_y)$. Value h_∞ stands for prescribed water height of outer, calm surface.

4.3 Numerical results

Numerical simulations of equation (4.8) have been done using C++ in-house code. The numerical fluxes were calculated by HLL scheme, first order in space. Temporal accuracy is second order due to the Crank–Nicolson scheme and CFL condition is $C = 0.5$. Computational domain was chosen as rectangle: $\Omega = \{[x, y] \in \mathbf{E}_2;\ 0 < x < 2,\ 0 < y < 2\}$, discretised by 200×200 cells. Initial conditions are as follows

$$\mathbf{u}(x, y, 0) = \mathbf{0} \tag{4.11}$$

and

$$h(x, y, 0) = \begin{cases} 1 + 2\sin(\pi\frac{x-0.8}{0.4})\sin(\pi\frac{y-0.8}{0.4}) & x, y \in \langle 0.8, 1.2\rangle \\ 1 \end{cases} \tag{4.12}$$

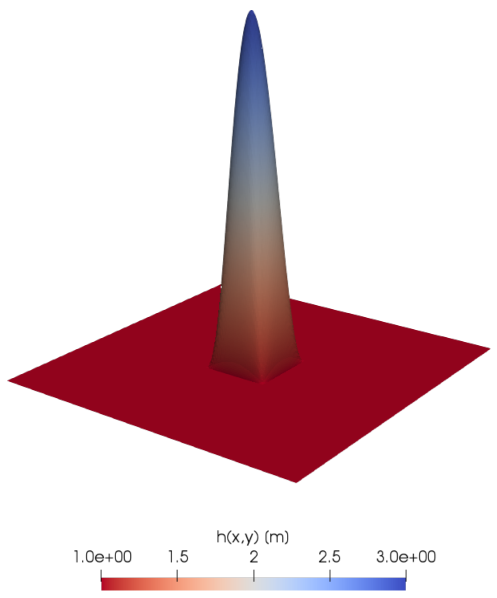


Figure 4.1: Two-dimensional shallow water equations, initial condition for water height

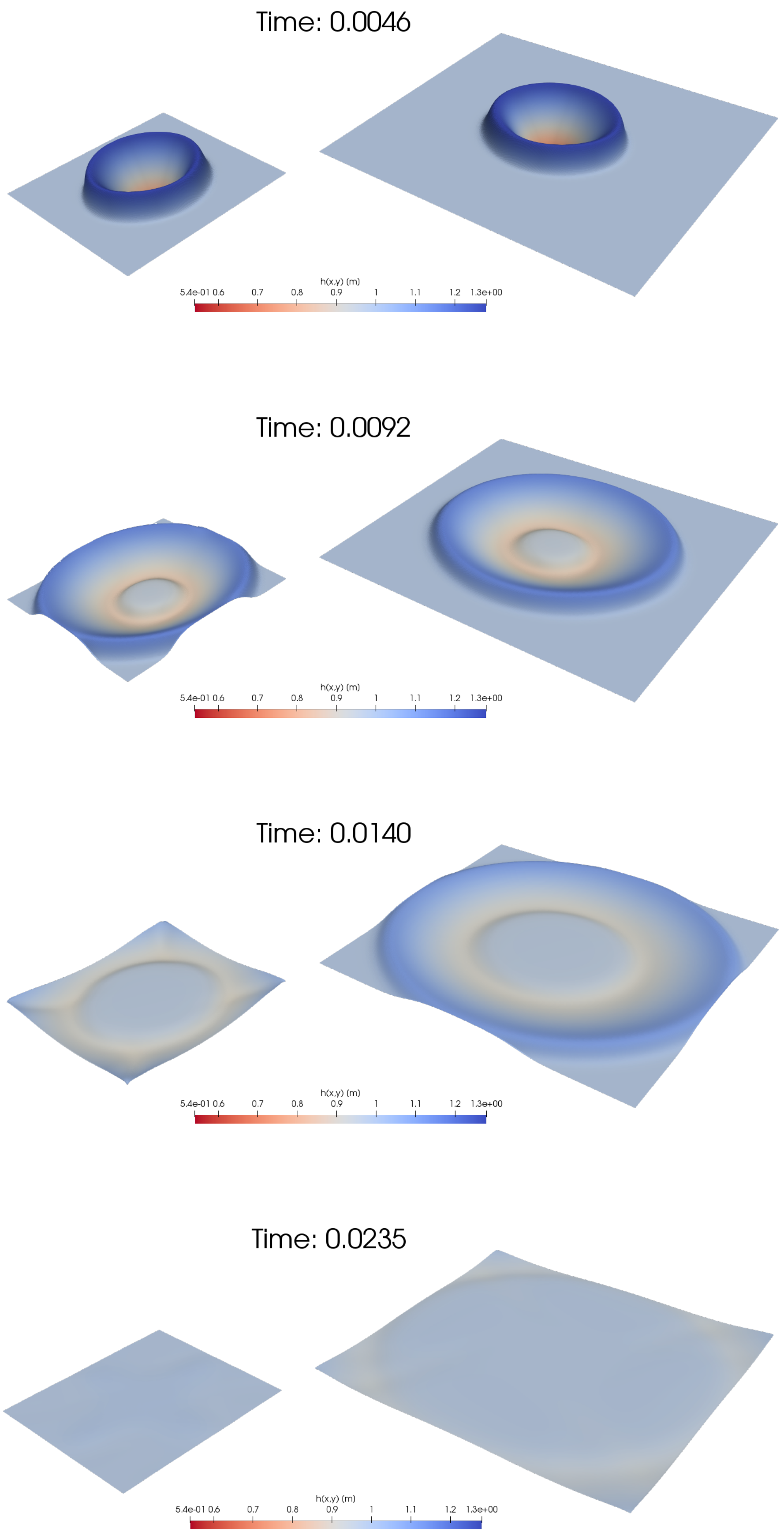

Time: 0.0046
h(x,y) (m)
5.4e-01 0.6 0.7 0.8 0.9 1 1.1 1.2 1.3e+00
Time: 0.0092
h(x,y) (m)
5.4e-01 0.6 0.7 0.8 0.9 1 1.1 1.2 1.3e+00
Time: 0.0140
h(x,y) (m)
5.4e-01 0.6 0.7 0.8 0.9 1 1.1 1.2 1.3e+00
Time: 0.0235
h(x,y) (m)
5.4e-01 0.6 0.7 0.8 0.9 1 1.1 1.2 1.3e+00

Figure 4.2: Two-dimensional shallow water equations, numerical results, water height

On the above Figure 4.2 there is showed comparison between aforementioned simulation (left) and its modificated version (right), equipped with elongated domain $\tilde{\Omega} = \{[x, y] \in \mathbf{E}_2;\ 0 < x < 3,\ 0 < y < 3\}$, divided into 300×300 FVM cells. The initial condition was re-centered in the middle of domain. In both simulations the non-reflective boundary conditions at all boundaries were prescribed.

One can see that initial condition was transported from within computational domain out without any significant reflections. This result supports the idea that there is a possibility to limit an arbitrary computational domain while still getting the same solution on smaller one.

5. Navier-Stokes equations - Volume of Fluid formulation

5.1 Introduction

The subject of multiphase flows encompasses a vast field, a host of different technological contexts, a wide spectrum of different scales, a broad range of engineering disciplines and a multitude of different analytical approaches [11]. Sometimes it is difficult to formulate precise definition of this phenomenon. It is due to the fact that one of the crucial task here - resolving of phase interface - could be dealt with various kinds of approaches. For example, solid-liquid system of some rigid bodies flowing in water can be qualified as a one-phase flow with solid fraction considered as computational boundary. On the other hand, one can simulate this problem - and it has been done - as multiphase one. This and many others examples demonstrate common denominator of multiphase flow - the complexity.

In the previous paragraph one of the key features of multiphase flow was mentioned, namely, the interface of two or more phases. Modeling of this phenomenon is complicated by geometric nature (e.g. curvature of interface, complex and unstructered computational grid) as well as the very physics of the problem (e.g. the advancing of a solid–liquid–gas contact line or the transition between different gas–liquid flow regimes). From topological point of view one can identify multiphase flows as *disperse flows* or *separated flows*. Disperse flows are those consisting of finite particles, drops or bubbles (the disperse phase) distributed in a connected volume of the continuous phase. On the other hand separated flows consist of two or more continuous streams of different fluids separated by interfaces.

In order to solve multiphase problems computationally, several methods were developed since 1960s, [12]. Some of them became obsolete, naturally, but led to many useful and provoking ideas which later turned out to be essential for developing new methods. One such idea proposes that the governing equations are solved on a fixed grid and the different fluids are identified by a marker function that is advected by the flow which allows to use only one set of equations for several fluid phases accompanied by equations for marker functions, describing phase interface. Several methods have been developed for that purpose. The volume-of-fluid (VOF) method is the oldest [12] and after many improvements and innovations continues to be widely used.

5.2 Mathematical model

Volume of fluid method has experienced a few changes since it has been presented in original article by Hirt and Nichols [13]. The original idea was to improve older, so-called SLIC (simple line interface calculation), method by proposing different treatment of advection scheme for the marker function. The new scheme was based on *reconstruction* of the phase interface in each cell using the values of the marker function in the neighboring cells. Since then, main development in domain of VOF method is aimed to propose better reconstruction schemes.

In following text only two phase flow of a mixture of liquid and gas phase will be considered. The liquid phase will be tracked with the step function $\alpha = \alpha(\mathbf{x}, t)$ which represents volumetric fraction occupied by this phase in arbitrary control volume. In the framework of FVM it means that $\alpha = 1$ in computational cells filled entirely with liquid phase and $\alpha = 0$ corresponds to a control volume containing only gaseous phase. As said above, the step function α allows to use only one set of equations in the entire flow domain for describing the local properties instead of one set of equations for each phase. This is done by calculating fluid properties as a linear combination of two phases, using α as a weight factor. For example total density in each computational cell is given by following equation

$$\rho = \alpha\rho_{liquid} + (1-\alpha)\rho_{gas}. \tag{5.1}$$

Because there are many variants, versions and software implementations of VOF method, the following description will be given according to up-to-date implementation in OpenFOAM software package. In subsections 5.3 and 5.4 two solvers treating VOF method of two-phase flows in OpenFOAM software will be discussed:

i) *interFoam* solver [14], equipped by *MULES* numerical scheme

ii) *interIsoFoam* solver [15], equipped by *isoAdvector* method

5.2.1 Governing equations

The fluids studied in this section are considered Newtonian, therefore the governing equations include continuity equation (5.2) and momentum equation (5.3) at first place.

$$\frac{\partial\rho}{\partial t} + \nabla\cdot(\rho\mathbf{u}) = 0 \tag{5.2}$$

$$\frac{\partial\rho\mathbf{u}}{\partial t} + \nabla\cdot(\rho\mathbf{u}\otimes\mathbf{u}) = -\nabla p + \nabla\cdot\mathbf{T} + \rho\mathbf{f} + \mathbf{f}_\sigma \tag{5.3}$$

Here ρ is density, $\mathbf{u}$ is velocity, p is pressure, $\mathbf{f}$ represents body forces and $\mathbf{f}_\sigma$ stands for surface tension. Tensor $\mathbf{T}$ is the deviatoric viscous stress tensor, expressed in full form as

$$\mathbf{T} = 2\mu\mathbf{S} - \frac{2}{3}\mu(\nabla\cdot\mathbf{u})\mathbf{I} \tag{5.4}$$

with the strain rate tensor

$$\mathbf{S} = \frac{1}{2}\Big(\nabla\mathbf{u} + (\nabla\mathbf{u})^T\Big) \tag{5.5}$$

where $\mathbf{I} = (\delta_{ij})$ is Kronecker delta tensor. In order to make previous set of equations complete in the sense of two-phase flows, there has to be provided an equation describing behavior of a phase interface. Supposing that aforementioned marker step function $\alpha(\mathbf{x}, t)$ changes only in region where phase transition is present, it can be regarded as a boundary condition for phase interface. So the following equation describes transport of phase interface in terms of function α.

$$\frac{\partial\alpha}{\partial t} + \nabla\cdot(\alpha\mathbf{u}) = 0, \tag{5.6}$$

This equation can be also viewed as a differential analog of *marker particles* used in Lagranigian oriented approaches [13].

Due to the fact that VOF model in OpenFOAM deals with both phases being incompressible one can drop out the second term from stress tensor, resulting in

$$\mathbf{T} = 2\mu\mathbf{S} \tag{5.7}$$

The continuity equation (5.2) is then reduced into

$$\nabla\cdot\mathbf{u} = 0 \tag{5.8}$$

Note: Transport equation for marker function (5.6) can be also derived strictly from continuity equation (5.2) by substitution of total density term (5.1) as follows

$$\frac{\partial(\alpha\rho_l + (1-\alpha)\rho_g)}{\partial t} + \nabla\cdot\Big((\alpha\rho_l + (1-\alpha)\rho_g)\mathbf{u}\Big) = 0 \tag{5.9}$$

$$\left[\rho_l\frac{\partial\alpha}{\partial t} + \rho_l\nabla\cdot(\alpha\mathbf{u})\right] + \left[\rho_g\frac{\partial(1-\alpha)}{\partial t} + \rho_g\nabla\cdot\Big((1-\alpha)\mathbf{u}\Big)\right] = 0 \tag{5.10}$$

where subscripts l and g stand for liquid and gas. Densities ρ_l and ρ_g are constant and terms in square brackets in equation (5.10) are linearly dependent equations, identical with equation (5.6).

The momentum equation (5.3) has been modified by surface tension $\mathbf{f}_\sigma$, which can be further expressed as

$$\mathbf{f}_\sigma = \sigma\kappa\nabla\alpha \tag{5.11}$$

where

$$\kappa = -\nabla \cdot \frac{\nabla\alpha}{|\nabla\alpha|} \tag{5.12}$$

Regarding the body forces term in momentum equation (5.3), it only act here due to the difference between phase fraction densities which by gravitational field cause buoyancy. Thus this term can be expressed as

$$\rho\mathbf{f} = -\mathbf{x}\cdot\mathbf{g}\nabla\rho \tag{5.13}$$

where $\mathbf{g} = (0, 0, -g)$ is gravitational acceleration vector with $g = 9.81$ and $\mathbf{x}$ is position vector.

Finally, governing equations of complete mathematical model in detailed form are presented hereby.

$$\nabla\cdot\mathbf{u} = 0 \tag{5.14}$$

$$\frac{\partial\alpha}{\partial t} + \nabla\cdot(\alpha\mathbf{u}) = 0 \tag{5.15}$$

$$\frac{\partial\rho\mathbf{u}}{\partial t} + \nabla\cdot(\rho\mathbf{u}\otimes\mathbf{u}) = -\nabla p + \nabla\cdot\left[\mu\Big(\nabla\mathbf{u} + (\nabla\mathbf{u})^T\Big)\right] - \mathbf{x}\cdot\mathbf{g}\nabla\rho - \sigma\kappa\nabla\alpha \tag{5.16}$$

5.3 interFoam

This solver is included in OpenFOAM-v5.0. version and represents the category of *algebraic* methods for phase interface reconstruction. This way of interface reconstruction is generally simple to implement, not restricted to structured grids, but on the other hand considered less accurate than *geometric* methods used *interIsoFoam*, which will be described later in separate section [16].

Above mentioned mathematical model of VOF method for two-phase flow consists of Navier-Stokes equations for incompressible fluid with appropriate modification of momentum equation (5.16) plus one extra equation for phase-interface transport (5.15) to make the model closed. In order to obtain proper coupling of velocity and pressure while solving these equations numerically, there is employed PISO algorithm where equations for $\mathbf{u}$ coupled with appropriately modified Poisson equation for pressure, p, are solved simultaneously with the interface advection problem.

Since the methodology of solving continuity and momentum equations (5.14),(5.16) is the same in both solvers, *interFoam* and *interIsoFoam*, and relies on PISO loop, following description will be focused on different approaches of handling interface transport equation (5.15).

At first there will be provided some description of how transport equation for marker function (5.15) is modified in *interFoam* solver followed by description of MULES (Multi-dimensional Universal Limiter for Explicit Solution) numerical scheme for advection term.

5.3.1 Compression term

Throughout the Section (5) until now there was used concept of flow velocity $\mathbf{u}$ as a by-product of combination of one-phase formulation equipped with marker function, which together allowed to describe two-phase flow. Nevertheless no physical meaning was added to this velocity. So, in following proceeding there is used analogy between VOF approach and two-fluid Eulerian model for two-phase flow, where phase fraction equations are solved separately for each individual phase [17]. This analogy can be expressed by defining the velocity of the effective fluid in a VOF model as linear combination of phase velocities as follows

$$\mathbf{u} = \alpha \mathbf{u}_l + (1-\alpha)\mathbf{u}_g \tag{5.17}$$

where $\mathbf{u}_l$ denotes velocity of liquid phase and $\mathbf{u}_g$ velocity of gas phase. Now, by replacing velocity in equation (5.15) by term (5.17) one gets

$$\frac{\partial \alpha}{\partial t} + \nabla \cdot (\alpha \mathbf{u}_l + (1-\alpha)\mathbf{u}_g) = 0. \tag{5.18}$$

Here, by defining relative velocity of phases

$$\mathbf{u}_r = \mathbf{u}_l - \mathbf{u}_g \tag{5.19}$$

and doing some algebraic manipulations, following form of phase transport equation results

$$\frac{\partial \alpha}{\partial t} + \nabla \cdot (\alpha \mathbf{u}_l) - \nabla \cdot \Big(\alpha(1-\alpha)\mathbf{u}_r\Big) = 0 \tag{5.20}$$

where holds

$$\frac{\partial \alpha}{\partial t} + \nabla \cdot (\alpha \mathbf{u}_l) = 0 \tag{5.21}$$

by definition of liquid phase transport, so finally one gets

$$\nabla \cdot \Big(\alpha(1-\alpha)\mathbf{u}_r\Big) = 0 \tag{5.22}$$

which is usually called *compression term*[1]. As long as the term (5.22) remains valid in continuum formulation, where with assumption of infinitesimal phase interface thickness α becomes step function and therefore $\alpha(1-\alpha)$ is identically zero, it can be added to original transport equation (5.15).

$$\frac{\partial \alpha}{\partial t} + \nabla \cdot (\alpha \mathbf{u}) + \nabla \cdot \Big(\alpha(1-\alpha)\mathbf{u}_r\Big) = 0 \tag{5.23}$$

This reformulation of transport equation for marker function is very useful because of additional *compression term*, which has no meaning in continuum formulation, but in numerical computation plays significant role in protection of phase interface smearing [18].

5.3.2 MULES

For numerical discretisation of convecting terms in equation (5.23) previously mentioned MULES explicit solver was used. Using Gauss's divergence theorem yields the equation (5.23) in semi discrete form follows

$$\frac{\partial \alpha_i}{\partial t} + \sum_f (\alpha_i \mathbf{u}_i)_f \cdot \mathbf{S}_f + \sum_f \Big(\alpha_i(1-\alpha_i)(\mathbf{u}_r)_i\Big)_f \cdot \mathbf{S}_f = 0 \tag{5.24}$$

[1]The term *compression* here refers to 'compress' the free surface towards sharper one, not to compressible flow (two phases of flow, i.e. water and air are here still considered incompressible)

where f denotes summation over all faces of arbitrarily chosen computational cell with index i. All physical quantities with lower index i are then considered as their mean values over computational cell (see Figure 5.1).

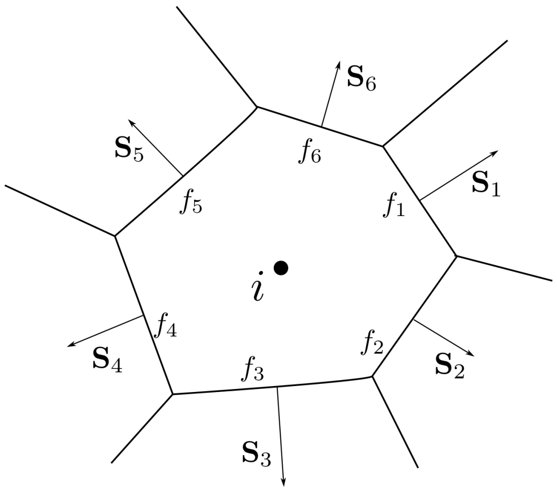


Figure 5.1: Sketch of a 2D unstructured computational cell

By rewriting equation (5.24) in a terms of face fluxes and using forward Euler temporal scheme following form is obtained

$$\frac{\alpha_i^{n+1+\nu} - \alpha_i^n}{\Delta t}\|\Omega_i\| + \sum_f \left[(\alpha_i^{n+\nu})_f \, F^{L,n+\nu} + \left(\alpha_i^{n+\nu}(1-\alpha_i^{n+\nu})\right)_f F^{NL,n+\nu}\right] = 0 \quad (5.25)$$

with $\|\Omega_i\|$ denoting volume of arbitrary computational cell, n and $n+1$ discrete time steps with ν being actual fixed-point inner iteration in order to deal with non-linearity of the fluxes during computation. Flux $F^{L,n}$ is the flux due to the center-of-volume velocity and $F^{NL,n}$ is the flux due to the relative velocity of phases (5.19) at the phase interface, which is calculated indirectly via flux, as in following equation

$$F^{NL,n} = n_f \min\left[C_\alpha \frac{|\Phi^{L,n}|}{|\mathbf{S}_f|}, \max\left(\frac{|\Phi^{L,n}|}{|\mathbf{S}_f|}\right)\right] \quad (5.26)$$

where C_α is adjustment constant, $\Phi^{L,n}$ is the volumetric face flux given by some low order scheme which guarantees to give monotonic results,

$$n_f = \frac{(\nabla\alpha)_f}{|(\nabla\alpha)_f + \delta_n|} \cdot \mathbf{S}_f \quad (5.27)$$

is the face unit normal flux, where $\mathbf{S}_f$ is face normal vector and

$$\delta_n = \frac{\epsilon}{\left(\frac{\sum_i \|\Omega_i\|}{N}\right)^{1/3}} \quad (5.28)$$

being stabilization factor to avoid division by zero [19]. Here N denotes number of all computational cells and $\epsilon = 1\times10^{-8}$. As one can see, direction of relative (or compression) velocity is given by $(\nabla\alpha)_f$ and thus acting perpendicular to phase interface. Computing $F^{L,n}$ face flux is done by high resolution scheme with following form

$$F^{L,n} = \lambda(\Phi_i^n - \Phi_N^n) + \Phi_N^n \quad (5.29)$$

where Φ_i^n is flux in volumetric center of current cell and similarly Φ_N^n the same in corresponding neighbor cell.

Blending factor λ is then given by

$$\lambda = \min\left(\max\left(1 - \max\left((1-4\alpha_i(1-\alpha_i))^2,\ (1-4\alpha_N(1-\alpha_N))^2\right), 0\right), 1\right) \tag{5.30}$$

with α_i and α_N denoted in the same way as fluxes in equation (5.29).

Further treatment of *interFoam* in OpenFOAM involves an adaptive-step control (based on face-computed Courant number) and sub-cycling in the solution of *alpha* transport equation in order to obtain stability of numerical computation.

Brief summation of an approach used in *interFoam* is that *compression term*, equation (5.22), reduces numerical diffusion of α at the phase interface and MULES scheme gives good combination between boundness and convergence [19].

5.4 interIsoFoam

As stated above, in subsection 5.3, *interIsoFoam* belongs in a family *geometric* methods, where treatment of phase interface reconstruction is done from volume fraction data. Original idea, presented in article [16], was to develop an algorithm, which works on unstructured grids is not computationally expensive and yet posses accuracy of the geometric schemes by explicitly approximating the interface. In following proceeding will be provided description of *isoAdvector* algorithm, implemented in OpenFOAM-v1712, which calculates face fluxes for the cells containing the interface. Description is adopted form original article [16] by Roenby, Bredmose and Jasak.

Governing equation for phase interface transport is here derived from mass conservation through arbitrary volume as follows.

$$\frac{d}{dt}\int_{\mathcal{V}} \rho(\mathbf{x},t)\ dV + \int_{\partial\mathcal{V}} \mathbf{u}(\mathbf{x},t)\rho(\mathbf{x},t)\ d\mathbf{S} = 0 \tag{5.31}$$

Here $\mathcal{V} \in \Omega$ is stationary arbitrarily chosen sub-domain of computational domain Ω and position vector $\mathbf{x}$ and velocity vector $\mathbf{u}$ are of corresponding dimensions. Now, considering that both fluids are incompressible with constant densities of liquid and gas phase (ρ_g and ρ_g), following density indicator field could be employed

$$H(\mathbf{x},t) \equiv \frac{\rho(\mathbf{x},t) - \rho_g}{\rho_l - \rho_g} \tag{5.32}$$

such that $H(\mathbf{x},t) = 1$ where liquid phase is present at given position $\mathbf{x}$ and time t. Analogously, for gas phase $H(\mathbf{x},t) = 0$.

Substituting term (5.32) in equation (5.31) and choosing integration domain as an arbitrary computational cell Ω_i one gets

$$\frac{d}{dt}\int_{\Omega_i} H(\mathbf{x},t)\ dV = -\sum_j \int_{f_j} \mathbf{u}(\mathbf{x},t)H(\mathbf{x},t)\ d\mathbf{S} \tag{5.33}$$

where index j denotes neighboring cells and thus f_j is common face between cells i and j. Vector $d\mathbf{S}$ is perpendicular to the boundary and by convention points out from i-th cell, see Figure 5.1.

Volume fraction field in i-th cell is naturally defined as

$$\alpha_i(t) \equiv \frac{1}{\|\Omega_i\|}\int_{\Omega_i} H(\mathbf{x},t)\ dV \tag{5.34}$$

where $\alpha_i(t) \in \langle 0, 1\rangle$ represents volume fraction of given cell filled by liquid phase.
If equation (5.33) is integrated with respect to t on interval $\langle t, \Delta t\rangle$

(formally relabeled with t), following form is obtained after some algebraic manipulations

$$\alpha_i(t+\Delta t) = \alpha_i(t) - \frac{1}{\|\Omega_i\|}\sum_j \int_t^{t+\Delta t}\int_{f_j} \mathbf{u}(\mathbf{x},t)H(\mathbf{x},t)\; d\mathbf{S}\; d\tau. \tag{5.35}$$

For further easier handling is the integral in equation (5.35) denoted as

$$\Delta V_j(t,\Delta t) \equiv \int_t^{t+\Delta t}\int_{f_j} \mathbf{u}(\mathbf{x},t)H(\mathbf{x},t)\; d\mathbf{S}\; d\tau. \tag{5.36}$$

thus $\Delta V_j(t,\Delta t)$ represents total volume of liquid phase fluid, transported during time interval $\langle t, t+\Delta t\rangle$ across the face f_j. In order to present *isoAdvector* algorithm there need to be provided some further terms, used later, such as

$$\mathbf{u}_i(t) \equiv \frac{1}{\|\Omega_i\|}\int_{\Omega_i} \mathbf{u}(\mathbf{x},t)\; dV\ , \qquad \phi_j(t) \equiv \int_{f_j} \mathbf{u}(\mathbf{x},t)\; d\mathbf{S} \tag{5.37}$$

The task for *isoAdvector* algorithm is to solve equation (5.35) numerically, which means that one needs determine its $\Delta V_j(t,\Delta t)$ part in some feasible way. Note, that up to now there hasn't been done any approximations in equation (5.35), so it holds exactly.

First, in the following procedure there is made approximation of velocity field $\mathbf{u}(\mathbf{x},t)$ in equation (5.35) so that velocity is considered constant during time integration, implying

$$\mathbf{u}(\mathbf{x},\tau) \approx \mathbf{u}(\mathbf{x},t) \qquad \forall\tau \in \langle t, t+\Delta t\rangle. \tag{5.38}$$

With this assumption, after some manipulations one can obtain

$$\Delta V_j(t,\Delta t) \approx \frac{\phi_j(t)}{|\mathbf{S}_j|}\int_t^{t+\Delta t} A_j(\tau)\; d\tau. \tag{5.39}$$

where

$$A_j(\tau) \equiv \int_{f_j} H(\mathbf{x},\tau)\; dS \tag{5.40}$$

with $dS \equiv d|\mathbf{S}|$. Hereby, the whole problem of phase interface resolving has shrunk into computing integral

$$\int_t^{t+\Delta t} A_j(\tau)\; d\tau. \tag{5.41}$$

Hence $A_j(\tau)$ in fact represents area of face f_j of computational cell intersected by liquid phase, the integral (5.41) then represents evolution of this intersection in time. In other words one has to determine motion of phase interface in the cell, represented by moving intersections of this interface with cells faces, *within* the numerical time step, see Figure(5.2). The whole procedure can be summarized into three steps.

At first, there is a need to determine initial position and orientation of phase interface in i-th arbitrary cell at time t. In order to do that, one has to interpolate volume fractions (marker function values) from cell centers to the grid points (red dots on Figure 5.2). After this step follows procedure of finding phase interface intersection with cell edges (blue dots on Figure 5.2). This is done by linear interpolation as follows

$$\mathbf{x}_{cut} = \mathbf{X}_k + \frac{f_X - f_{X_k}}{f_{X_l} - f_{X_k}}(\mathbf{X}_l - \mathbf{X}_k) \tag{5.42}$$

where $\mathbf{x}_{cut}$ is the position vector of phase interface intersection with cell edge given by $(\mathbf{X}_k, \mathbf{X}_l)$. Here indexes $k, l \in \{1, ..., N\}$ denote any of the N vertices of cell i connected by an edge. Values $f_{X_k},\ f_{X_l}$ represent previously mentioned volume fractions interpolated to matching cells vertices. The interpolation on given edge is then done only if parameter $f_k < f_X < f_l$ otherwise entire edge lies whole in liquid or gas phase. The parameter f_X here represents unknown position of phase interface in cell i to be computed so that divides cell in two parts in the way that liquid and gas phase are in the same ratio as given by cell centered marker function at initial time $\alpha_i(t)$. Summing up, $\mathbf{x}_{cut}$ and f_X are implicitly connected via phase fraction ratio in given cell. Broadly speaking, it means that in given cell phase fraction position and orientation are solved all together. Details of solution can be found in original article [16].

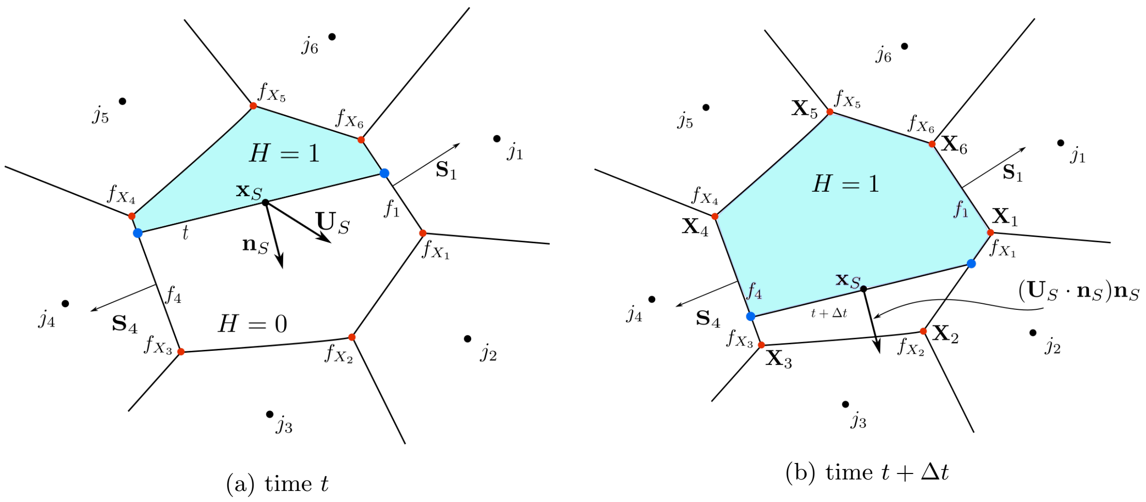


Figure 5.2: Scheme of phase interface surface motion in i-th computational cell during times t and $t + \Delta t$

Second step of the *isoAdvector* algorithm deals with determining of phase fraction velocity in order to estimate its motion within the computational timestep. In the beginning there are used common OpenFOAM tools for computing geometric center, $\mathbf{x}_S$, of moving phase interface and the velocity vector in this center, $\mathbf{U}_S$, as some linear combination of known velocity field at initial time, t. After that is computed normal component of $\mathbf{U}_S$, perpendicular to phase interface, given by $(\mathbf{U}_S \cdot \mathbf{n}_S)\,\mathbf{n}_S$, with $\mathbf{n}_S$ representing unit normal vector pointing out of liquid phase by convention. While phase surface movement is determined, one can calculate its time integral, equation 5.41, by some numerical method. During the time integration on interval $\langle t, \Delta t\rangle$ one has to keep in mind that intersection of phase interface with cells faces (blue dots on Figure 5.2) passes through different faces so the flow has to be always computed on correct face. This is ensured by breaking integration interval into some sub-intervals on which intersection does not change the any face. There is provided closer description of this procedure in original article, including some further implementation details about bounding procedure, employed to provide strict boundedness of solution.

Finally, substituting all adequate terms back to equation 5.35, which governs the transport of phase interface, values of α in next time step can be evaluated.

6. Implementation of non-reflective boundary condition to VOF method

6.1 Motivation

As stated in first section, the original motivation of this work was simulation of water pump placed in a semi-opened basin. In order to make this case easier to handle for developing and implementing non-reflective boundary conditions, original geometry was trimmed of unimportant features. Though using the simplified geometry, shown on Figure (6.1), the original (un)physical phenomena causing problems during the computation of flow are still present.

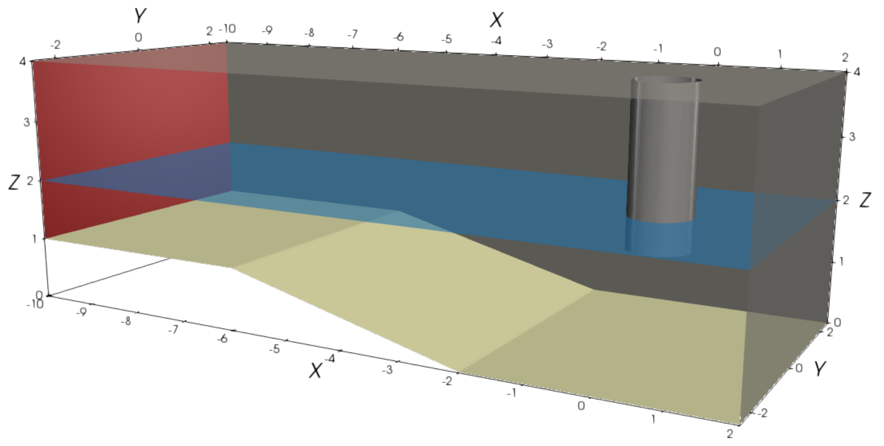


Figure 6.1: Geometry of a 3D semi-opened basin

On the Figure (6.1) is presented 3D geometry of a tank equipped with vertical tube serving as water pump. Solid walls of the tank are composed of bottom (yellow), front and back wall, top, pipe and right wall (all grey). Front wall is transparent here. The red left wall is only artificial computational boundary, physically not present in the basin.

The primary goal of the original engineering task was to determine behavior of water level when its height is 'close' at the inlet of pipe at specified outflow when pumping the water out.

The first numerical simulation was performed with basin placed in the water so that water level set up with flat surface near the pipe. Hence the basin is semi-opened and considered to be placed in sufficiently large water source, the water level is expected to vary around initial state only slightly during the pumping. Nevertheless, after some time from computation start there occurred bigger waves in the tank, presumed to be an artifact of reflecting smaller ones form both sides, right and left wall of the basin. Due to the superposition of waves at the position of pipe inlet (together with their breaking) the pump sometimes sucked mixture of water and air and started working unstationary.

The simulation was performed in the OpenFOAM-v5.0 where reflections from left boundary appeared while using currently available boundary conditions. This initiated the effort to prescribe more suitable boundary conditions and finally led to creating new, non-reflective, boundary conditions inspired by shallow water flows.

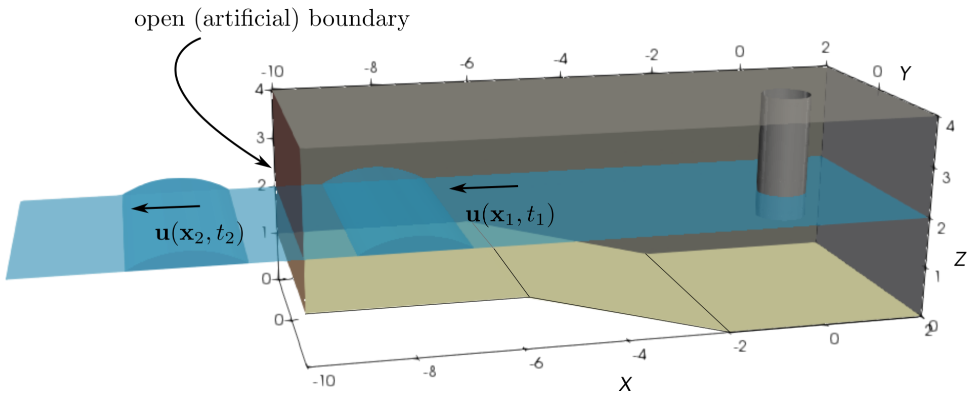

open (artificial) boundary
$\mathbf{u}(\mathbf{x}_2, t_2)$
$\mathbf{u}(\mathbf{x}_1, t_1)$
X
Y
Z

Figure 6.2: Scheme of situation where wave is approaching left boundary in time t_1. After passing the boundary being unhampered, the (imaginary) wave propagates unchanged, time t_2.

6.2 Implementation

The construction of non-reflective boundary conditions will be demonstrated for the case of aforementioned left artificial boundary of the water tank. Considering VOF model governed by equations (5.14), (5.15) and (5.16) with $\rho = \alpha\rho_l + (1 - \alpha)\rho_g$ and $\mu = \alpha\mu_l + (1 - \alpha)\mu_g$ where ρ_l, ρ_g, μ_l and μ_g are constant, one has to prescribe boundary conditions for pressure, p, velocity, $\mathbf{u}$ and marker function, α. So, the task is to prescribe combination of α and $\mathbf{u}$ as an equivalent of h and $\mathbf{u}$ in shallow water terminology. The $\mathbf{u}$ and α in the center of boundary adjacent cells are known at the beginning of each time step t^n, but they are varying along the water depth because of 3D nature of VOF model. To be able to use similarity with shallow water equations one must do some type of vertical averaging of these quantities.

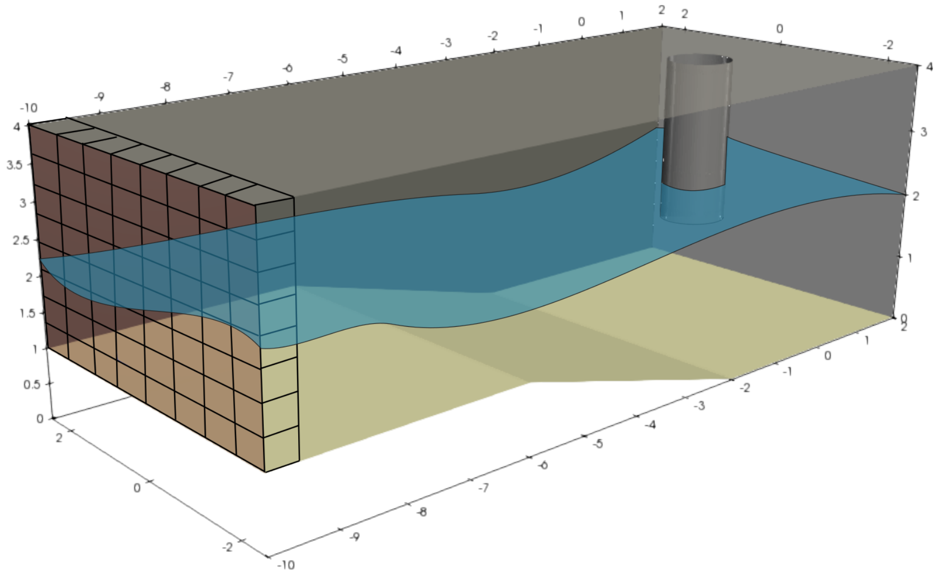


Figure 6.3: Scheme of computational domain with highlighted computational cells at left boundary and with reconstructed water surface.

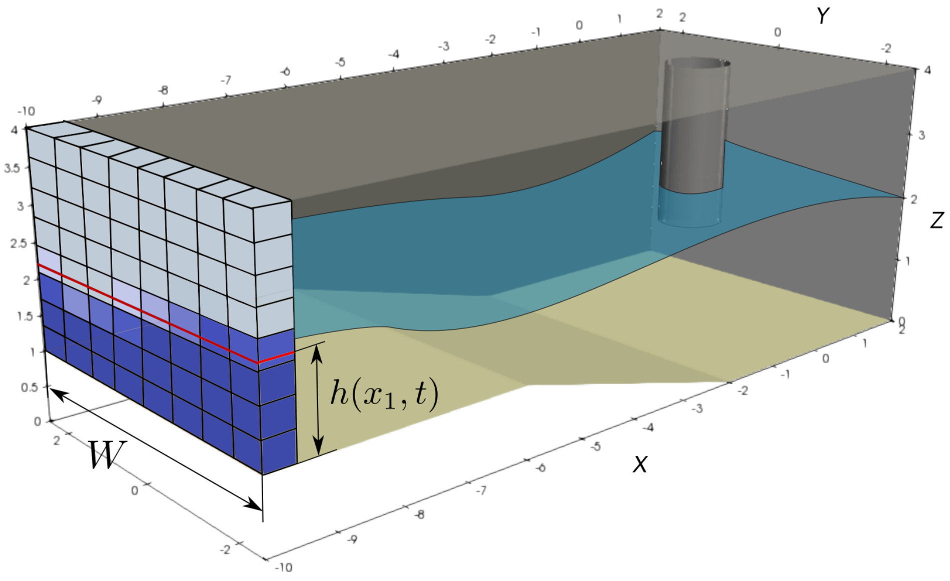


Figure 6.4: Scheme of boundary adjacent computational cells colored according to the value of marker function. Dark blue for $\alpha = 1$ light blue for $\alpha = 0$ and mixed shade for $0 < \alpha < 1$.

In the following approach there is used averaging not only in vertical direction (z), but also in horizontal direction (y) since the implementation in OpenFOAM is simple to proceed in this way. Although that 3D nature of waves incoming to the non-reflective boundary is destroyed by this approximation, the errors of solution are rather small considering that variation of water level in y-direction at the boundary is not large. The averaging of α and $\mathbf{u}$ is expressed by following relations

$$h(x_{b+1}, t^n) = \frac{\displaystyle\int_{\Omega_b} \alpha(x_{b+1}, y, z, t^n)\, dydz}{W} \tag{6.1a}$$

$$u(x_{b+1}, t^n) = \frac{\displaystyle\int_{\Omega_b} \alpha(x_{b+1}, y, z, t^n) \cdot u(x_{b+1}, y, z, t^n)\, dydz}{\displaystyle\int_{\Omega_b} \alpha(x_{b+1}, y, z, t^n)\, dydz} \tag{6.1b}$$

where x_{b+1} denotes x-coordinate of boundary cells centers, Ω_b represents the boundary, W is the width of the tank and u is the velocity in x-direction ($\mathbf{u} = (u, v, w)$).[2] With the $h(x_{b+1}, t^n)$ and $u(x_{b+1}, t^n)$ one can now determine water height and velocity at the boundary in time t^{n+1} in a similar way as it was done in equations (3.20) and (3.20) in the case of one-dimensional shallow water problem. Once $h(x_b, t^{n+1})$ and $u(x_b, t^{n+1})$ are obtained, boundary conditions for $\mathbf{u}$ and α are computed as

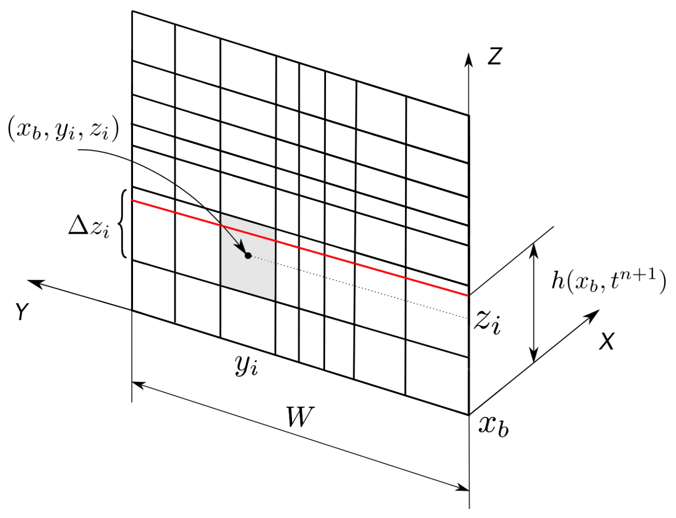
$$\alpha(x_b, y, z, t^{n+1}) = \begin{cases} 0, & z > h(x_b, t^{n+1}) \\ 1, & z < h(x_b, t^{n+1}) \end{cases} \tag{6.2a}$$

$$\mathbf{u}(x_b, y, z, t^{n+1}) = \alpha(x_b, y, z, t^{n+1})\Big(u(x_b, t^{n+1}), 0, 0\Big). \tag{6.2b}$$

[2]Note that integrals over the boundary Ω_b in equations (6.1) turn into sums when implemented to FVM method

For pressure is prescribed homogenous Neumann condition, $\dfrac{\partial p}{\partial \mathbf{n}} = 0$.

The way of spatial discretisation (actually employed in OpenFOAM) of terms in equation (6.2) is described as follows. Since the computational grid could be generally both structured or unstructured the cells faces at boundary are then rectangles or more generally convex polygons. Physical quantities in these faces are then represented by average values in the centers of gravity of each face. Let us denote positions of these centers as (x_b, y_i, z_i) where i represents index of an arbitrary face, see Figure 6.5.



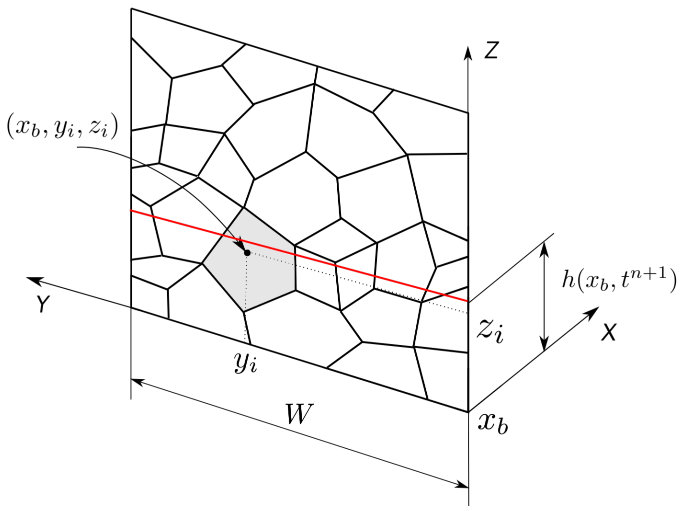


Figure 6.5: Scheme of boundary cells for structured and unstructured computational grid

In the case of structured, rectilinear grid, shown on top picture of Figure 6.5, there is a possibility to make simple linear interpolation of term 6.2a as shown below

$$\alpha(x_b, y_i, z_i, t^{n+1}) = \min\left[1,\ \max\left(0, \left(h(x_b, t^{n+1}) - \frac{z_i - \Delta z_i}{\Delta z_i}\right)\right)\right] \tag{6.3}$$

because Δz_i is easy to determine here. The equation 6.3 expresses that if entire cell face is under computed water height, $h(x_b, t^{n+1})$, its marker function value is set as $\alpha = 1$ and if the face is above, $\alpha = 0$. When face is intersected by $h(x_b, t^{n+1})$, the value of α is set according to face fraction bellow the intersection.

The situation of unstructured grid is more difficult to deal with. The current implementation is done with simple substitution of vertical position, z, by discrete value of arbitrary cell center, z_i

$$\alpha(x_b, y_i, z_i, t^{n+1}) = \begin{cases} 0, & z_i > h(x_b, t^{n+1}) \\ 1, & z_i < h(x_b, t^{n+1}) \end{cases} \quad . \tag{6.4}$$

This simplification is less accurate than linear interpolation used in the case of orthogonal grid. Therefore, the goal of the future work is to use OpenFOAM tools which can do re-mapping of unstructured boundary grid with all physical quantities to cartesian reference grid. The computation on reference grid is then done as in equation 6.3 followed by inverse re-mapping to original grid.

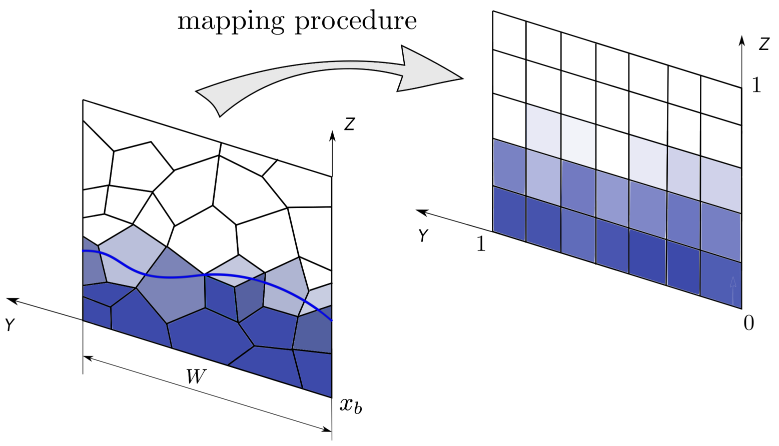


Figure 6.6: Scheme of remapping original boundary grid with values of α onto reference grid

6.3 Numerical results

6.3.1 Two dimensional dam-break problem

All following numerical simulations were performed in OpenFOAM. As a testing case there was naturally chosen dam-break problem, solved by VOF method as two-dimensional flow, which corresponds with one-dimensional shallow water problem. Computational domain is described by $\Omega = \{[x, z] \in \mathbf{E}_2;\ 0 < x < 1,\ 0 < z < 0.5\}$ with finite volume discretisation done by cartesian grid with 200×100 cells in corresponding directions $(x \times z)$. Initial condition for α is set according Figure 6.7, being chosen opposite to the 1D shallow water simulation (Figure 3.6) while artifficial boundary still considered on the left. This is done in order to overcome possible discontinuities in water surface.

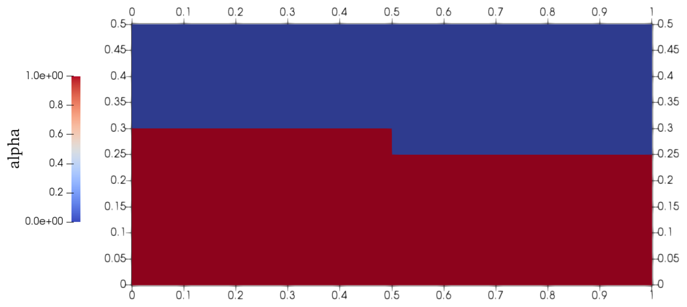


Figure 6.7: VOF, dam-break, initial condition ($t = 0s$)

Initial conditions for velocity and pressure are set as $\mathbf{u} = 0$ and $p = 0$ in entire domain.

At the top, right and bottom boundary the homogeneous Neumann condition is prescribed for α and $p - \mathbf{x} \cdot \mathbf{g}\rho$ and the non-slip boundary condition is used for $\mathbf{u}$. At the left, artificial boundary there are tested two types of boundary conditions. In first case there are prescribed native OpenFOAM boundary conditions for α and $\mathbf{u}$. These conditions are linked to each other allowing water level to vary ($\partial\alpha/\partial\mathbf{n} = 0$) in accordance with prescribed volumetric flow rate. In the second case there are prescribed non-reflective boundary conditions for α and $\mathbf{u}$. For the pressure there is on the left boundary in both cases prescribed boundary condition setting the pressure gradient to the provided value such that the flux on the boundary is that specified by the velocity boundary condition. [20]

Both cases are solved with either by MULES or isoAdvector numerical scheme for advection term in equation (5.23) and with LES model for turbulence equipped by additional turbulence kinetic energy equation.

On the following series of figures are compared water heights (obtained as a isosurface contours for $\alpha = 0.5$) in several times for MULES and isoAdvector schemes with both, standard and non-reflective boundary conditions.

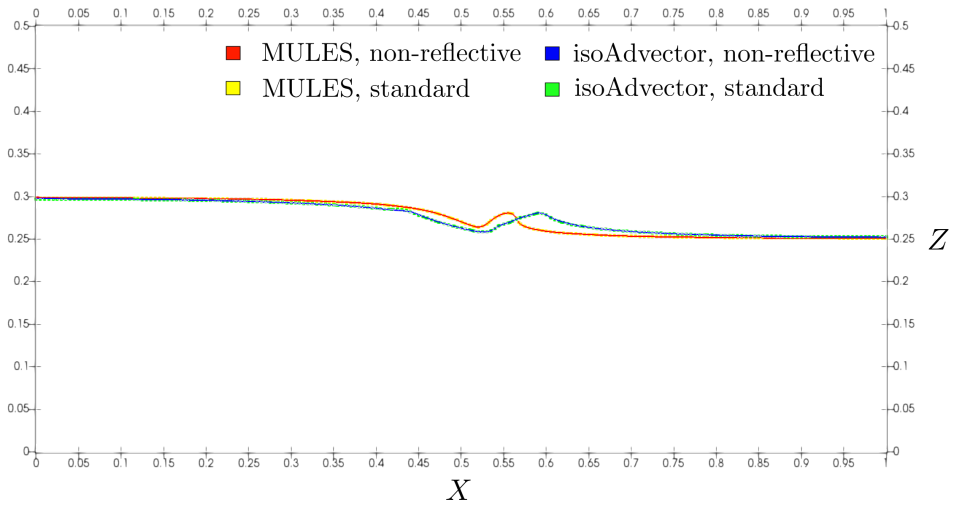


Figure 6.8: VOF, dam-break, water heights, $t = 0.1$ s

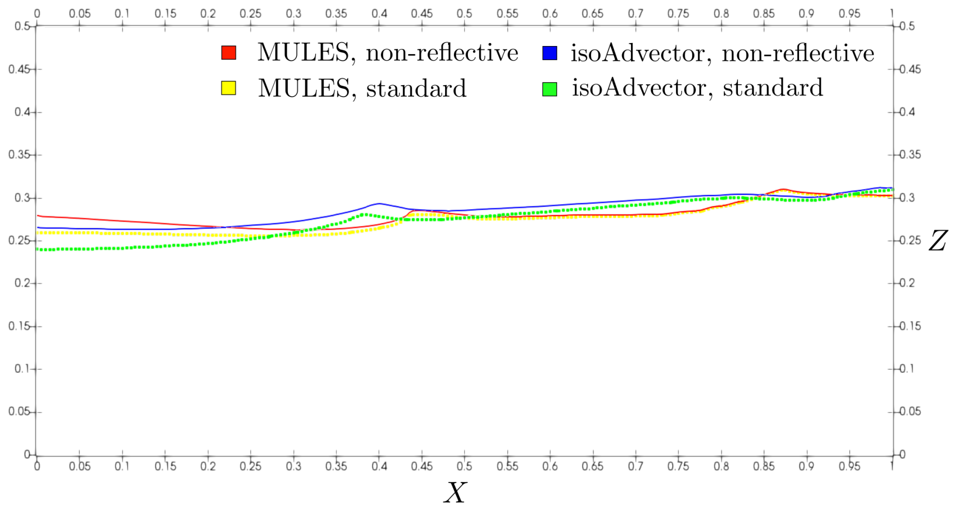


Figure 6.9: VOF, dam-break, water heights, $t = 0.5$ s

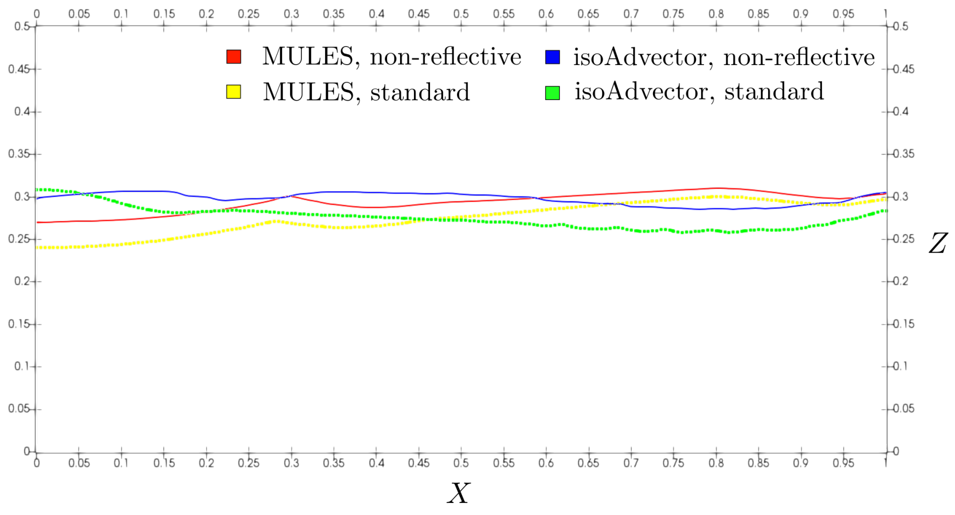


Figure 6.10: VOF, dam-break, water heights, $t = 0.8$ s

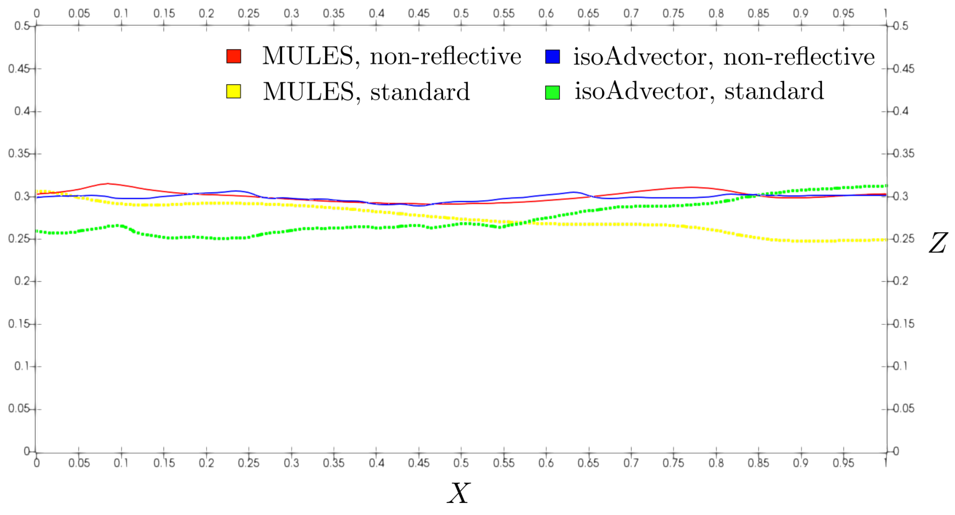


Figure 6.11: VOF, dam-break, water heights, $t = 1.5$ s

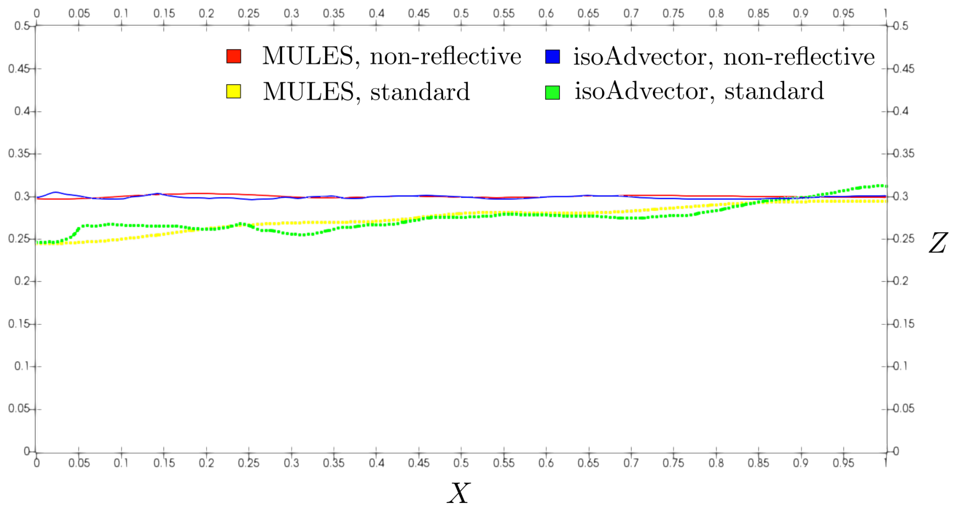


Figure 6.12: VOF, dam-break, water heights, $t = 3.2$ s

On the Figure 6.8 there are depicted water heights slightly after beginning of simulations, therefore no significant differences between used solvers or boundary conditions are present. The Figure 6.9 shows water waves have been reflected from right wall and now approaching to the left artificial boundary. There also start to emerge little variations between MULES and isoAdvector with that isoAdvector tends to resolve water surface slightly sharper, with capturing higher frequency waves. On the other hand MULES seems like to possessing some kind of *numerical* viscosity. Figures 6.10 and 6.11 display significant differences between standard and non-reflective boundary conditions, with waves being reflected from side to side using standard boundary conditions. Moreover, MULES and isoAdvector schemes lead to different water heights here. On the Figure 6.11 are nearly calm water surfaces in the case of non-reflective boundary conditions. This is because majority of disturbances passed out of computational domain and there is constituting calm surface, given by $h_{-\infty} = 0.3$, specified as water level *outside* of computational domain. In the contrast, standard boundary conditions still generate oscillations in solution.

Following figures show comparison between original computational domain and its elongated version, given by $\tilde{\Omega} = \{[x, z] \in \mathbf{E}_2;\ -1 < x < 1,\ 0 < z < 0.5\}$. Water height is obtained by isosurface contours, as in previous cases.

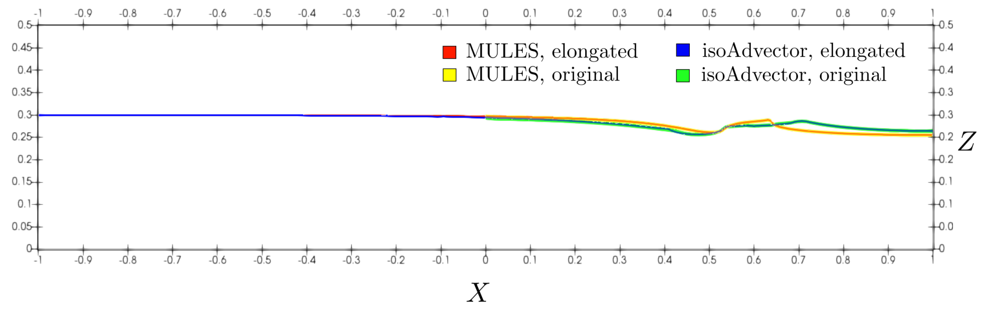


Figure 6.13: VOF, dam-break, water heights, $t = 0.2\ s$

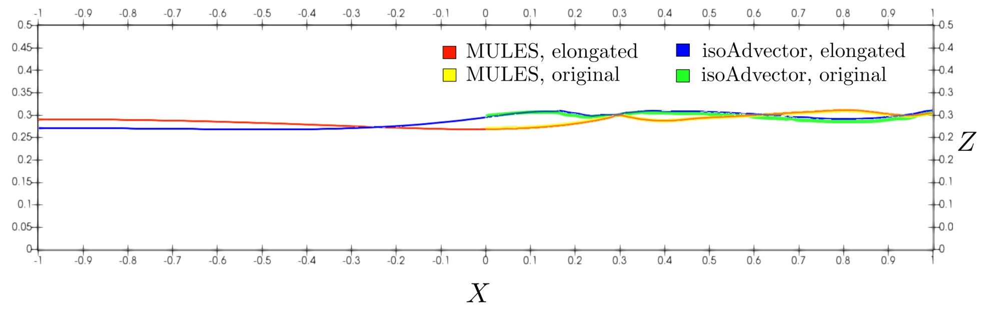


Figure 6.14: VOF, dam-break, water heights, $t = 0.8\ s$

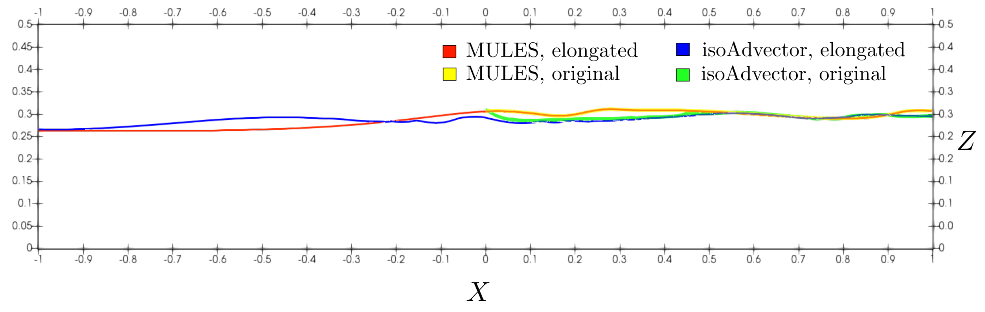


Figure 6.15: VOF, dam-break, water heights, $t = 1.2\ s$

One can see that throughout Figures 6.13-6.15 solutions on the original domain (plotted on $0 < x < 1$) agree with ones on the elongated domain (plots of water heights of related solutions mostly overlap each other). It implies that non-reflective boundary conditions allow to effectively reduce computational domain and therefore computational cost.

6.3.2 Three dimensional water pump

Following case demonstrates the performance of the non-reflective boundary conditions on simulation of vertical water pump draining out water from semi-opened basin, Figure 6.1, as also presented in [20]. This case is computed in three dimensions by VOF method with MULES approach for resolving phase interface. Computational domain is unstructured, mainly composed of hexahedral cells with total number of 380 000 cells. On the left, artificial side ($x = -10$) there is considered either standard, variable water height or non-reflective boundary condition for $\mathbf{u}$ an α. The non-slip boundary conditions for $\mathbf{u}$ are prescribed on front and back wall, bottom and right wall. Boundary conditions for α and $p - \mathbf{x} \cdot \mathbf{g}\rho$ are set the in correspondence with two-dimensional dam-break problem. There is left one extra set of boundary conditions need to be imposed on the water pump outlet, see Figure 6.16. Velocity is here determined by specified volumetric flow rate of $\dot{V}_{out} = 4\ m^3/s$, known as $flowRateOutletVelocity$ in OpenFOAM. For α and $p - \mathbf{x} \cdot \mathbf{g}\rho$ there are prescribed homogeneous Neumann conditions. Initial conditions for $\mathbf{u}$ and p are set as zero in entire domain, α is set in correspondence with Figure 6.16. There is used the same turbulence model (LES with k-Equation) as in the 2D dam-break VOF simulation.

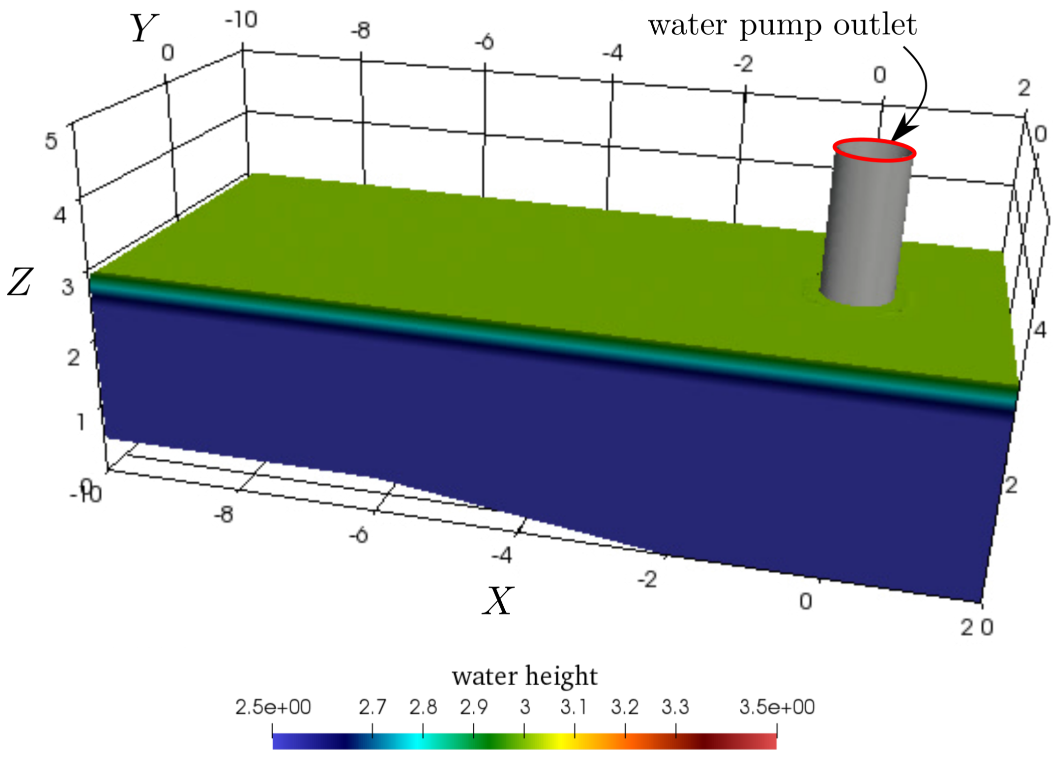


Figure 6.16: VOF, water pump, water heights, $\mathbf{t = 0.0\ s}$

On the Figures 6.17 and 6.18 there can be seen a noticeable difference between non-reflective boundary condition (upper picture) and standard, variable height boundary condition (lower picture). The non-physical behavior is caused by water waves being reflected back into computational domain.

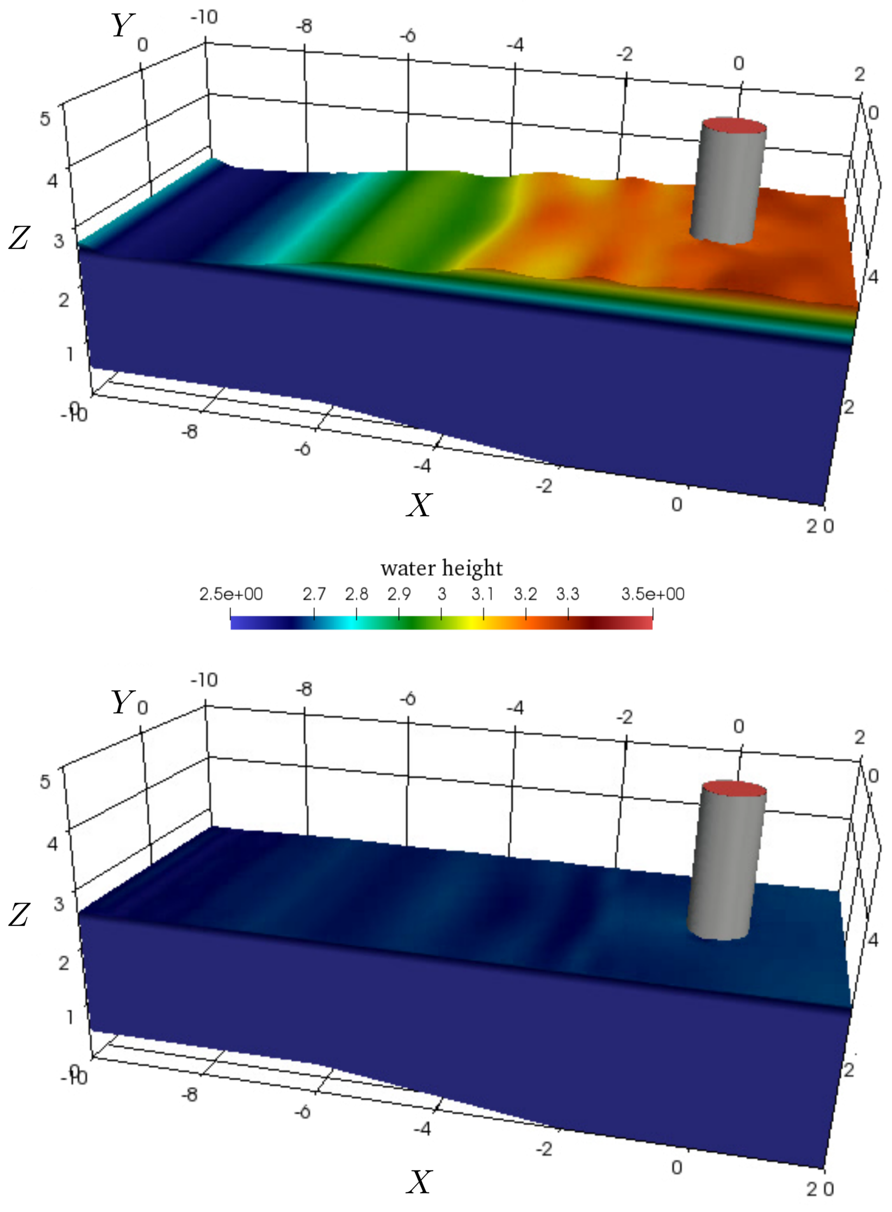


Figure 6.17: VOF, water pump simulation, water heights, variable water height boundary conditions (upper picture), non-reflective boundary conditions (lower picture), $\mathbf{t} = \mathbf{14.0\ s}$

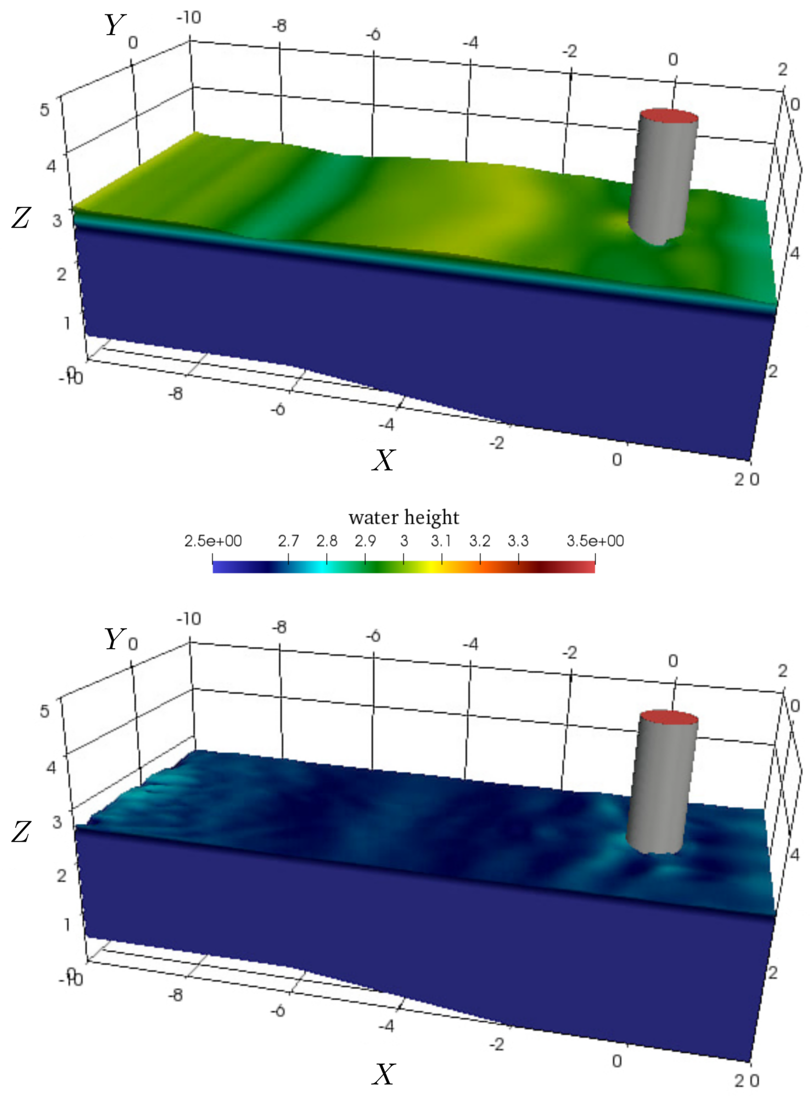


Figure 6.18: VOF, water pump simulation, water heights, variable water height boundary conditions (upper picture), non-reflective boundary conditions (lower picture), **t = 33.0 s**

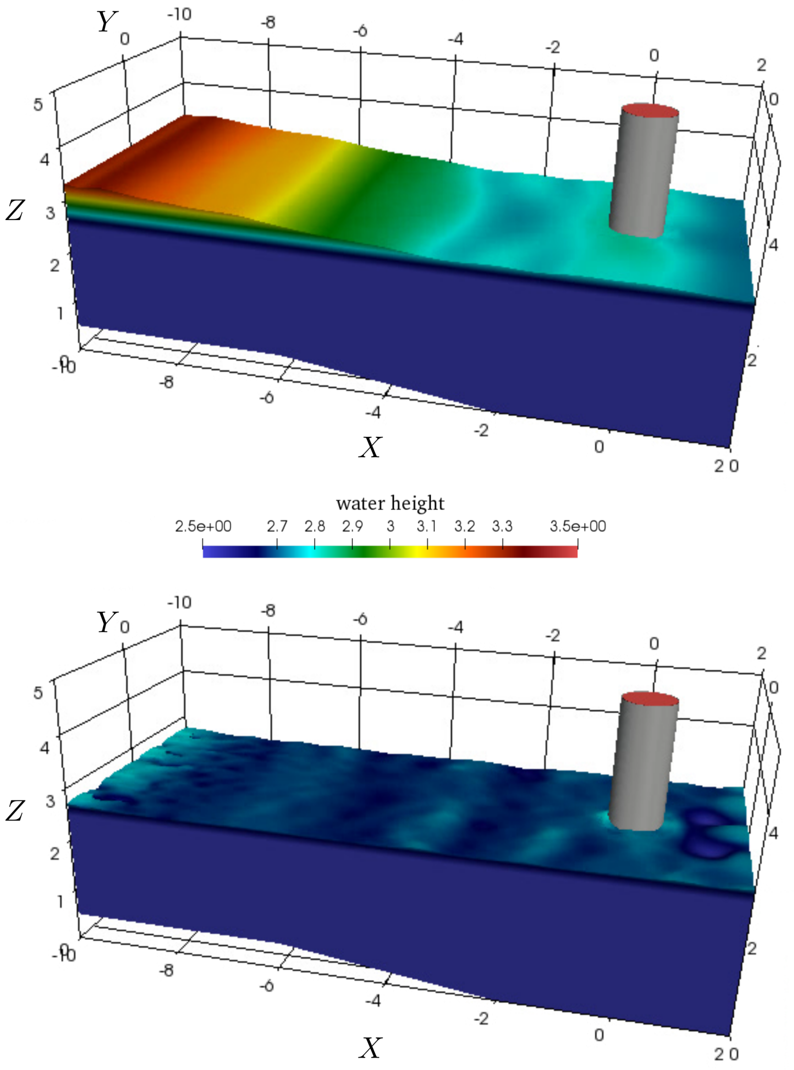


Figure 6.19: VOF, water pump simulation, water heights, variable water height boundary conditions (upper picture), non-reflective boundary conditions (lower picture), $\mathbf{t = 47.0\ s}$

Despite the fact that waves of larger amplitudes are effectively removed by non-reflective boundary condition, there are still present small wavelets on the lower picture of Figure 6.19. This phenomenon might be explained by several facts. First, as was stated in section 6.2, while implementing non-reflective boundary condition there was used averaging of α not only in vertical direction (which would result in $h(x_{b+1}, y, t_n)$, water height still depending on y-coordinate) but also along the width of water tank, compare with equation 6.1a. This kind of treatment is easier to accomplish in OpenFOAM, but makes non-reflective boundary condition for three-dimensional VOF method to behave in correspondence with the one derived for one-dimensional shallow water flow. This implies that spatial character of waves incoming to the artificial boundary is destroyed, causing probably major part of present errors. Second problem may be related with the possibility of violating the flow regime during the VOF simulation. This means that nature of flow computed by VOF method should be as close as possible to the flow described by theory of shallow water from which non-reflective boundary condition is derived (i.e. low vertical velocity of flow at the artificial boundary, continuous water surface, etc.). Another issue could be the first order spatial accuracy of derived boundary condition.

7. Conclusion

Main goals of the thesis presented in the beginning have been fulfilled. At first, analysis of shallow water equations has been carried out in order to derive non-reflective boundary conditions. In following part, performance of the non-reflective boundary conditions has been tested on the one-dimensional dam-break problem and its two-dimensional parallel. Furthermore, description of two-phase VOF mathematical model has been given with the emphasis on two different, widely used methods for capturing phase interface. After that the major contribution of this work, consisting of adoption of shallow water non-reflective boundary conditions into the VOF method has been presented. In the end, performance of non-reflective boundary conditions in the framework of VOF method has been demonstrated on results of numerical simulations of two-dimensional dam-break problem and three-dimensional water pump problem. Some of these results have been also presented in [20] and [21].

Even though non-reflective boundary conditions perform satisfactorily in VOF model, in the end of the section 6 have been mentioned some minor errors related with aspects of derivation of boundary condition. The most significant assumption that has been made during the derivation has been addressed with averaging of water height along the width of water tank at the boundary, hence neglecting its spatial character in that direction.

The future research is therefore aimed to evaluate the water height with respect to the y-coordinate (see i.e. Figure 6.5). Because this could be quite hard task to do on unstructured grids, there is expected a bypass with the mapping procedure being employed, as has been proposed in subsubsection 6.2. This treatment will also allow to compute water height more precisely in vertical direction but moreover it is promising when extending the boundary condition for non-flat bottom topography (especially along y-coordinate of water tank). Last, but not the least perspective for future work is to improve spatial accuracy of the boundary condition to the 2^{nd} order.

The VOF method and shallow water theory are attached to many actual research activities, some of them are either employing or at least proposing the combination of those approaches in order to reduce computational cost (i.e. I.Vandebeek et al. [22], González Rodríguez et al. [23]). This is done with using shallow water equations to describe phenomena of less interest in order to save computational effort (i.e. water waves far form shore), followed by applying VOF method for describing waves breaking on the shore. With awareness of mentioned research subjects there is a hope that the work presented in this thesis might be utilized to a greater extend.

References

[1] Dan Givoli. High-order local non-reflecting boundary conditions: a review. *Wave Motion*, 39(4):319 – 326, 2004. New computational methods for wave propagation. URL: http://www.sciencedirect.com/science/article/pii/S0165212503001203, http://dx.doi.org/https://doi.org/10.1016/j.wavemoti.2003.12.004 doi:https://doi.org/10.1016/j.wavemoti.2003.12.004.

[2] Brett Sanders. Non-reflecting boundary flux function for finite volume shallow-water models. 25:195–202, 02 2002.

[3] John R Dea. *High-Order Non-Reflecting Boundary Conditions for the Linearized Euler Equations*. PhD thesis, Naval Postgraduate School Monterey, California, 2008.

[4] Joseph B. Keller and Dan Givoli. Exact non-reflecting boundary conditions. *Journal of Computational Physics*, 82(1):172 – 192, 1989. URL: http://www.sciencedirect.com/science/article/pii/0021999189900417, http://dx.doi.org/https://doi.org/10.1016/0021-9991(89)90041-7 doi:https://doi.org/10.1016/0021-9991(89)90041-7.

[5] H. G. Weller, G. Tabor, H. Jasak, and C. Fureby. A tensorial approach to computational continuum mechanics using object-oriented techniques. *Comput. Phys.*, 12(6):620–631, November 1998. URL: http://dx.doi.org/10.1063/1.168744, http://dx.doi.org/10.1063/1.168744 doi:10.1063/1.168744.

[6] C.B. Vreugdenhil. *Numerical Methods for Shallow-Water Flow*. Water Science and Technology Library. Springer Netherlands, 1994. URL: https://books.google.cz/books?id=8uohw01mgaEC.

[7] H. Segur and H. Yamamoto. Lecture 8: The shallow-water equations, June 2009. URL: https://www.whoi.edu/fileserver.do?id=136564&pt=10&p=85713.

[8] Jan Nordstrom. Well posed problems and boundary conditions in computational fluid dynamics. In *22nd AIAA Computational Fluid Dynamics Conference*, page 3197, 2015.

[9] Y.P. Petrov and V.S. Sizikov. *Well-posed, Ill-posed, and Intermediate Problems with Applications*. Inverse and ill-posed problems series. VSP, 2005. URL: https://books.google.cz/books?id=J43XGdOcX8sC.

[10] E.F. Toro. *Riemann Solvers and Numerical Methods for Fluid Dynamics: A Practical Introduction*. Springer Berlin Heidelberg, 2013. URL: https://books.google.cz/books?id=zkLtCAAAQBAJ.

[11] Christopher Earls Brennen and Christopher E Brennen. *Fundamentals of multiphase flow*. Cambridge university press, 2005.

[12] A. Prosperetti and G. Tryggvason. *Computational Methods for Multiphase Flow*. Cambridge University Press, 2009. URL: https://books.google.cz/books?id=KBuKZkEUWMIC.

[13] C.W. Hirt and B.D. Nichols. Volume of fluid (vof) method for the dynamics of free boundaries. *J. Comput. Phys.; (United States)*, 39:1, 1 1981. http://dx.doi.org/10.1016/0021-9991(81)90145-5 doi:10.1016/0021-9991(81)90145-5.

[14] The OpenFOAM Foundation Ltd. Openfoam v5.0, August 2018. URL: `https://openfoam.org/version/5-0/`.

[15] ESI Group. Opencfd release openfoam® v1712, August 2018. URL: `https://www.openfoam.com/releases/openfoam-v1712/`.

[16] Johan Roenby, Henrik Bredmose, and Hrvoje Jasak. A computational method for sharp interface advection. *Royal Society open science*, 3(11):160405, 2016.

[17] Edin Berberović, Nils P van Hinsberg, Suad Jakirlić, Ilia V Roisman, and Cameron Tropea. Drop impact onto a liquid layer of finite thickness: Dynamics of the cavity evolution. *Physical Review E*, 79(3):036306, 2009.

[18] S Márquez Damián. Description and utilization of interfoam multiphase solver. URL: `http://infofich.unl.edu.ar/upload/3be0e16065026527477b4b948c4caa7523c8ea52.pdf`.

[19] S Márquez Damián. An extended mixture model for the simultaneous treatment of short and long scale interfaces. *Doktorarbeit. Universidad Nacional Del Litoral. Facultad de Ingenieria y Ciencias Hidricas*, 2013.

[20] J. Fürst and J Musil. Development of non-reflective boundary condition for free-surface flows. *In: ŠIMURDA, D. a BODNÁR, T., eds. Topical Problems of Fluid Mechanics. Topical Problems of Fluid Mechanics 2018. Praha, 21.02.2018 - 23.02.2018. Prague: Institute of Thermomechanics, AS CR, v.v.i.. 2018*, pages 97–104, Mar 2018. URL: `http://www.it.cas.cz/fm/im/im/proceeding/2018/13`.

[21] J. Fürst and J Musil. Development of non-reflective boundary condition for free-surface flows. *In: 33rd conference with international participation Computational Mechanics 2017 - Extended Abstracts*, 33rd CONFERENCE WITH INTERNATIONAL PARTICIPATION Computational Mechanics 2017. Špičák, Železná Ruda, 06.11.2017 - 08.11.2017:29–30, Mar 2017.

[22] Ine Vandebeek, Vincent Gruwez, Corrado Altomare, Tomohiro Suzuki, Dieter Vanneste, Sieglien De Roo, Erik Toorman, and Peter Troch. Towards an efficient and highly accurate coupled numerical modelling approach for wave interactions with a dike on a very shallow foreshore. In *Coastlab 2018*, pages 1–10, 2018.

[23] Ernesto Mauricio González Rodríguez, Iñigo Aniel-Quiroga Zorrilla, Omar Quetzalcoatl Gutiérrez Gutiérrez, et al. Evaluation of tsunami run-up on coastal areas at regional scale. 2017.