der Bundeswehr
Universität München

Fakultät für Luft- und Raumfahrttechnik
Institut für Raumfahrttechnik und Weltraumnutzung

Computational modeling of nonlinear propellant sloshing onboard spacecraft

Manuel Hahn

Vollständiger Abdruck der von der Fakultät für Luft- und Raumfahrttechnik der Universität der Bundeswehr München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs (Dr.-Ing.)

genehmigten Dissertation.

Die Dissertation wurde am 04.12.2020 bei der Universität der Bundeswehr München eingereicht und durch die Fakultät für Luft- und Raumfahrttechnik am 17.07.2021 angenommen. Die mündliche Prüfung fand am 19.07.2021 statt.

Abstract

Propellant sloshing inside a partially filled tank onboard a spacecraft is a major disturbance for attitude and orbit control, causing performance degradations and in severe cases even an unstable spacecraft. Hence the impact of a sloshing propellant on the spacecraft's structure and thus on the Attitude & Orbit Control System needs to be appropriately considered in spacecraft design and verification. Existing propellant slosh modeling approaches comprise mechanical analog models, which mainly consider linear sloshing motion only, and computational models, which lack the accurate and robust description of generic nonlinear sloshing motion in both high-gravity and low-gravity environments as for example high-velocity impacts on the tank wall or generic wetting effects including contact angle hysteresis. Additionally, the high computational cost of the currently existing computational models makes it impossible to use them directly in Attitude & Orbit Control System verification campaigns.

The main objective of this thesis is to develop a computational propellant sloshing model able to reproduce both linear as well as nonlinear sloshing effects typically arising in spacecraft missions. The derived computational model is validated for high-gravity environments through hydrostatic and dynamic sloshing tests, comparing the simulation results to theory, experiments and other numerical models. A validation for low-gravity conditions is done by simulating surface tension as well as wetting effects and comparing the results to theory.

Optimizing the computational propellant sloshing model as well as its software implementation for shorter simulation runtimes and better usability is the second objective of this work. The presented computational propellant sloshing model is capable of simulating problems using a low spatial resolution and thus a low simulation runtime, while still providing global system properties with sufficient accuracy. Additionally, the software implementation of the computational model uses a lightweight parallelization technique that is not adding too much overhead when using low spatial resolutions. The usability of the software is increased by providing a convenient way of pre-processing including automatic geometry creation as well as post-processing comprising data visualization, data analysis and export of simulation results. Also, all parameters used to setup a simulation have a real physical meaning, like e.g. dynamic viscosity or surface tension, hence no parameters specific to the chosen numerical method need to be configured.

Finally, the third objective of the thesis at hand is the validation of the computational propellant sloshing model for Attitude & Orbit Control System verification simulations. This is achieved by developing a generic spacecraft Attitude & Orbit Control System simulator and coupling it to the derived propellant sloshing simulator. Two coupled simulations are performed covering applications of nonlinear propellant sloshing in high-gravity and zero-gravity conditions, showing that the developed computational propellant sloshing model is well suited to be used in Attitude & Orbit Control System verification campaigns.

Kurzfassung

Das Schwappen von Treibstoff in teilweise gefüllten Tanks an Bord von Raumfahrzeugen ist eine bedeutende Störquelle für die Bahn- und Lageregelung und führt zu einer Beeinträchtigung der Leistungsfähigkeit des Reglers sowie in schweren Fällen sogar zu einem instabilen Verhalten. Bei Design und Verifikation des Raumfahrzeugs muss daher der Einfluss des Treibstoffschwappens auf die Raumfahrzeugstruktur und damit auf das Bahn- & Lageregelungssystem angemessen berücksichtigt werden. Bestehende Ansätze zur Modellierung des Treibstoffschwappens umfassen mechanische Analogmodelle, die hauptsächlich nur lineares Schwappverhalten abdecken, sowie rechnergestützte Modelle, die keine genaue und robuste Beschreibung der generischen nichtlinearen Schwappynamik unter Schwerkraft als auch unter Schwerelosigkeit ermöglichen, wie es zum Beispiel bei Hochgeschwindigkeitsaufprallen auf die Tankwand oder bei generischen Benetzungseffekten inklusive der Kontaktwinkelhysterese erforderlich ist. Außerdem macht der hohe Berechnungsaufwand der bestehenden rechnergestützten Modelle es unmöglich, sie direkt gekoppelt in einer Verifikationskampagne des Bahn- und Lageregelungssystems zu verwenden.

Das Hauptziel dieser Arbeit ist die Entwicklung eines rechnergestützten Modells des Treibstoffschwappens, welches in der Lage ist, für Raumfahrtmissionen typisches lineares als auch nichtlineares Schwappverhalten abzubilden. Für Umgebungen mit Schwerkraft erfolgt die Validierung des entwickelten rechnergestützten Modells durch hydrostatische und dynamische Schwapptests, wobei die Simulationsergebnisse mit Theorie, Experimenten und anderen rechnergestützten Modellen verglichen werden. Die Validierung für Umgebungen in Schwerelosigkeit erfolgt durch Simulationen der Oberflächenspannung sowie von Benetzungseffekten, wobei die Simulationsergebnisse mit der Theorie verglichen werden.

Die Optimierung des rechnergestützten Treibstoffschwappmodells sowie dessen Softwareimplementierung hinsichtlich kurzer Simulationslaufzeiten und einer gesteigerten Gebrauchstauglichkeit ist der zweite Hauptbeitrag dieser Arbeit. Das vorliegende rechnergestützte Modell unterstützt die Durchführung von Simulationen mit niedriger räumlicher Auflösung und damit geringer Simulationslaufzeit bei gleichzeitig ausreichender Genauigkeit der Reproduktion globaler Systemeigenschaften. Außerdem wird bei der Softwareimplementierung des rechnergestützten Modells eine leichtgewichtige Parallelisierungstechnik eingesetzt, die besonders bei niedrigen räumlichen Auflösungen keinen großen zusätzlichen Verwaltungsaufwand für die Nebenläufigkeit erfordert. Die Gebrauchstauglichkeit der Software wird gesteigert durch eine komfortable Simulationskonfiguration inklusive automatischer Geometriegenerierung sowie einer praktischen Datennachbereitung, die die Visualisierung, Analyse und den Export der Simulationsergebnisse umfasst. Zusätzlich werden keine künstlichen methodenspezifischen Parameter zur Simulationskonfiguration eingesetzt, sondern lediglich Parameter mit reeller physikalischer Bedeutung, wie zum Beispiel dynamische Viskosität oder Oberflächenspannung.

Der dritte Hauptbeitrag der vorliegenden Arbeit ist die Validierung des rechnergestützten Treibstoffschwappmodells für Verifikationssimulationen des Bahn- & Lageregelungssystems. Dafür wird ein generischer Simulator für das Bahn- & Lageregelungssystem von Raumfahrzeugen aufgebaut und an den zuvor entwickelten Treibstoffschwappsimulator gekoppelt. Zwei gekoppelte Simulationen werden durchgeführt zur Demonstration von nichtlinearem Treibstoffschwappen in Umgebungen mit Schwerkraft sowie unter Schwerelosigkeit. Die Ergebnisse zeigen, dass das entwickelte rechnergestützte Modell des Treibstoffschwappens hervorragend geeignet ist, um in Verifikationskampagnen des Bahn- & Lageregelungssystems verwendet zu werden.

Acknowledgment

The thesis at hand was developed in my spare time, i.e. next to my full-time job, during the last 6 years and many people were supporting me in this period. In the following, I want to express my gratitude to all these people who helped to make this thesis a success.

First I want to thank Prof. Dr. Roger Förstner for giving me the opportunity to do a doctorate at the Institute of Space Technology & Space Applications. Giving me full freedom in my research and at the same time being always available for helping me when I needed it, created a really nice working environment that I cannot imagine being better. Thanks!

I also want to thank my colleagues at the institute. Event though I was an external PhD candidate and additionally working on my research topic only in part-time, I always felt like being part of the team and enjoyed the discussions - both technically and personally - with you a lot. Special thanks goes to Dr. Tom Andert for his help in organizing my PhD defense as well as for the nice time teaching the exercises of Dynamics & Control of Satellites.

A big thank you goes to Dr. Stefan Adami for many fruitful technical discussions about the numerical aspects of my thesis. Even though you never had any obligation to do so cause we were never officially connected via any project etc., you were always available for talking about my ideas, problems and thoughts, what helped me a lot. And all this just out of scientific curiosity and interest in the topic. Thanks so much!

I also want to thank the entire SPH community for their continuous effort in developing the SPH method. When starting my PhD in 2015, I attended the SPHERIC[1] conference in Parma and met many nice people who were all open for discussing my ideas and for answering my questions regarding some aspects of the method. Also later, I profited from their continuous method improvements which are reflected in great scientific papers, as well as from the SPHERIC SPH benchmark test cases that allowed me to validate my code against reproducible experiments.

Finally, special thanks goes to my family. When performing an entire PhD in one's spare time next to a full-time job, it is clear that time for family needs to be shortened. And that I received great support from you, Ania, during all these years and under these circumstances is for sure not a given. I thank you from the bottom of my heart for your infinite patience, support in all kinds of situations, comfort and motivation in tough times as well as fun and great moments in good times. Without you I wouldn't have made it!

Munich, August 2021
Manuel Hahn

[1]SPHERIC is the international organization representing the community of researchers and industrial users of Smoothed Particle Hydrodynamics (SPH).

Contents

Abstract i

Kurzfassung iii

Acknowledgment v

List of figures ix

List of tables xi

Nomenclature xiii

1 Introduction 1
- 1.1 Propellant sloshing as disturbance . . . 2
- 1.2 Existing modeling approaches and shortcomings . . . 4
- 1.3 Research objectives and contributions . . . 6
- 1.4 Outline . . . 8

2 Computational modeling of propellant sloshing using Smoothed Particle Hydrodynamics 11
- 2.1 Physics of propellant sloshing . . . 11
- 2.2 Smoothed Particle Hydrodynamics approximation . . . 14
- 2.3 Transport-velocity formulation for weakly-compressible SPH . . . 18
- 2.4 Discretization of continuity equation . . . 20
- 2.5 Discretization of momentum equation . . . 22
- 2.6 Temporal discretization . . . 25
- 2.7 Discretization of boundary conditions . . . 27

3 Software implementation of the propellant sloshing model 29
- 3.1 Requirements specification . . . 29
- 3.2 Design overview . . . 31

4 Validation of propellant sloshing model in high-gravity conditions 37
- 4.1 Introduction . . . 37
- 4.2 Hydrostatic test . . . 39
- 4.3 Dynamic sloshing test . . . 42
- 4.4 System properties under reduced spatial resolution . . . 45

5 Validation of propellant sloshing model in zero-gravity conditions 49
- 5.1 Introduction . . . 49
- 5.2 Surface tension . . . 51
- 5.3 Wetting effects . . . 55

6 Computational modeling of spacecraft attitude & orbit dynamics and control 59
6.1 Physics of spacecraft maneuvers 59
6.2 Temporal discretization 62
6.3 Spacecraft attitude & orbit control 64

7 Software implementation of the spacecraft attitude & orbit dynamics and control model 67
7.1 Requirements specification 67
7.2 Design overview 69
7.3 Validation 73

8 Coupling propellant sloshing and spacecraft attitude dynamics & control 75
8.1 Coupling strategy 75
8.2 Software implementation of the coupling 77

9 Application to AOCS verification of nonlinear propellant sloshing 79
9.1 Forced roll-motion in high-gravity conditions 79
9.1.1 Experimental & numerical setup 79
9.1.2 Simulation of a forced roll-motion 81
9.2 Earth observation satellite mission scenario in zero-gravity conditions 84
9.2.1 Mission scenario description 84
9.2.2 Numerical setup 86
9.2.3 Artificial sound wave analysis 87
9.2.4 Simulation of an Earth observation satellite mission scenario 91

10 Conclusions 99
10.1 Thesis summary 99
10.2 Future work 101

Bibliography 103

List of Figures

1.1 Introduction: Visualization of the emergence of disturbances due to a sloshing propellant in a partially filled tank onboard a spacecraft and their consequences on the Attitude & Orbit Control System 2
1.2 Introduction: Visualization of the propellant slosh modeling techniques currently used in literature including how they are derived from physics 4

2.1 Computational propellant sloshing modeling: Visualizing the variables involved in the stress balance at the interface of two fluid phases 12
2.2 Computational propellant sloshing modeling: Visualizing a solid-liquid-gas interface 13
2.3 Computational propellant sloshing modeling: Illustration of the SPH approximation for a field quantity 14

3.1 Propellant sloshing simulator: Components of the software implementation including interfaces 31
3.2 Propellant sloshing simulator: Excerpt from a configuration file comprising a subset of parameters needed to setup a simulation 32
3.3 Propellant sloshing simulator: Post-processing tool: Particle states data example with velocity plotted 33
3.4 Propellant sloshing simulator: Post-processing tool: System states data example 34
3.5 Propellant sloshing simulator: Workflow of a simulation using the proposed computational model 35

4.1 High-gravity validation: Experimental test setup 37
4.2 High-gravity validation: Numerical test setup 38
4.3 Hydrostatic test: Pressure distribution at time $t = 1$ s 39
4.4 Hydrostatic test: Pressure and force in z-direction results 40
4.5 Hydrostatic test: Center of mass and force results 42
4.6 Hydrodynamic test: Analysis of forces in x-direction exerted by the fluid on the tank structure 43
4.7 Hydrodynamic test: Power spectral density of the forces in x-direction exerted by the fluid on the tank structure 44
4.8 Resolution study: Center of mass of the fluid for three different resolutions . . . 45
4.9 Resolution study: Forces exerted by the fluid on the tank structure for three different resolutions 46
4.10 Resolution study: Analysis of forces in x-direction exerted by the fluid on the tank structure for three different resolutions 47

5.1 Surface tension: Initial geometrical particle distribution 51
5.2 Surface tension: Converged particle distribution 52
5.3 Wetting effects: Initial geometrical particle distribution 55

5.4 Wetting effects: Resulting droplets on a horizontal solid surface for prescribed static contact angles . . . 56
5.5 Wetting effects: Geometrical determination of simulated static contact angles . . 57

6.1 Computational spacecraft maneuver modeling: Visualization of the body reference frame with respect to an inertial reference frame . . . 60
6.2 Computational spacecraft maneuver modeling: AOCS components overview: Hardware comprising sensors and actuators . . . 64

7.1 Spacecraft AOCS simulator: Components of the software implementation including interfaces . . . 69
7.2 Spacecraft AOCS simulator: Excerpt from a configuration file comprising a subset of parameters needed to setup a simulation . . . 70
7.3 Spacecraft AOCS simulator: Overview of classes including their relationships . . 71
7.4 Spacecraft AOCS simulator: Inertial angular momentum of the rigid-body simulation . . . 73
7.5 Spacecraft AOCS simulator: Maximum relative error of the inertial angular momentum for all involved time integration methods and time step sizes . . . 74

8.1 Coupling: Overview of hardware related classes of the spacecraft AOCS simulation environment including their relationships . . . 77
8.2 Coupling: Components of the software implementation of the coupled propellant sloshing and spacecraft AOCS simulator including interfaces . . . 78

9.1 Forced roll-motion in high-gravity: Experimental setup . . . 80
9.2 Forced roll-motion in high-gravity: Numerical setup . . . 80
9.3 Forced roll-motion in high-gravity: Simulated vs. experimental pressure values . 81
9.4 Forced roll-motion in high-gravity: Time series of screenshots from the overlay of the sloshing motion from experiment and a visualization of the simulation results 83
9.5 Earth observation satellite mission scenario: Angular rate profile of the two successive bang-bang slew maneuvers around the y-axis . . . 84
9.6 Earth observation satellite mission scenario: Experimental setup . . . 85
9.7 Earth observation satellite mission scenario: Setting up an initial particle distribution . . . 86
9.8 Earth observation satellite mission scenario: Sound wave analysis: Accelerations in z-direction . . . 89
9.9 Earth observation satellite mission scenario: Simulation results . . . 91
9.10 Earth observation satellite mission scenario: Image series . . . 92
9.11 Earth observation satellite mission scenario: Total torque as well as control torque in y-direction . . . 93
9.12 Earth observation satellite mission scenario: Simulation without considering the propellant dynamics: Power spectral density of the total torque in y-direction for a time between 40 and 60 s . . . 94
9.13 Earth observation satellite mission scenario: Attitude error Euler vector . . . 95
9.14 Earth observation satellite mission scenario: Attitude error Euler vector comparison between feedback and feedforward simulations . . . 96

List of Tables

4.1 Hydrodynamic test: Comparison of results for the 1st antisymmetric sloshing mode frequency . . . 43

5.1 Surface tension: Comparison of pressure difference across the fluid-fluid interface 53
5.2 Surface tension: Comparison of pressure difference across the fluid-fluid interface for three different spatial resolutions . . . 54
5.3 Wetting effects: Comparison of static contact angles . . . 58
5.4 Wetting effects: Comparison of static contact angles for three different spatial resolutions . . . 58

9.1 Forced roll-motion in high-gravity: Comparison of the experimentally and computationally determined pressure impulses at the sensor location for the four sloshing impacts . . . 82
9.2 Earth observation satellite mission scenario: Sound wave analysis: Estimated fundamental frequencies of sound wave oscillations to be expected in the numerical system due to the weakly compressible numerical approach . . . 88

Nomenclature

Abbreviations

AOCS	Attitude & Orbit Control System
API	Application programming interface
CAD	Computer-aided design
CFL	Courant–Friedrichs–Lewy
CSV	Comma-separated value
HPC	High-performance computing
OBC	On-board computer
OOP	Object-oriented programming
PID	Proportional-integral-derivative
RAM	Random-access-memory
RK4	Fourth-order Runge-Kutta method
SPH	Smoothed Particle Hydrodynamics

Symbols

$\mathbf{A}$	Transport-velocity extra stress tensor
$\vec{\mathbf{a}}$	Acceleration
$\vec{\mathbf{a}}_{s/c}$	Acceleration of spacecraft's center of mass
$_{\alpha}$	Index for particles of fluid phase α
$_{B}$	Index for body reference frame
$_{\beta}$	Index for particles of fluid phase β
c_s	Artificial speed of sound
d	Particle diameter

DCM_{IB}	Direction cosine matrix from body to inertial frame
$\vec{\mathbf{e}}_\omega$	Angular velocity error
$\vec{\mathbf{e}}_q$	Euler vector representation of the attitude error
f_{11}	First antisymmetric sloshing mode frequency
f_N	Nyquist frequency
f_s	Sampling frequency
$\vec{\mathbf{F}}^A$	Force due to transport-velocity extra stress
$\vec{\mathbf{F}}^b$	Body force
$\hat{\mathbf{F}}_b$	Body force per unit volume
$\vec{\mathbf{F}}_{\text{int}}$	Integrated force exerted by propellant on tank wall
$\vec{\mathbf{F}}^m$	Molecular-like pairwise interaction force
$\vec{\mathbf{F}}^p$	Presssure force
$\vec{\mathbf{F}}^v$	Viscous force
γ	Damping ratio
h	Smoothing length
H	Support radius
$\vec{\mathbf{h}}_{\text{int}}$	Internal angular momentum
$\vec{\mathbf{H}}_{\text{total}}$	Total angular momentum
$_I$	Index for inertial reference frame
$\mathbf{I}_{\mathrm{s/c}}$	Spacecraft's moment of inertia
κ	Curvature of the interface
κ_{max}	Prescribed maximum compressibility
m	Mass
μ	Dynamic viscosity
$\vec{\mathbf{n}}$	Normal vector
n	Particle number density
ν	Kinematic viscosity
O	Origin of coordinate system

ω_n	Control bandwidth
$\vec{\boldsymbol{\omega}}_B$	Angular velocity
$\dot{\vec{\boldsymbol{\omega}}}_B$	Angular acceleration
p	Pressure
P	Pressure impulse
ϕ	Field quantity
$\mathbf{q}_{BI}$	Attitude quaternion from inertial to body frame
ρ	Density
$_s$	Index for particles of solid phase s
s	Interaction force strength
σ	Interfacial or surface tension
$\vec{\mathbf{T}}$	External torque
$\vec{\mathbf{T}}_{\text{int}}$	Integrated torque exerted by propellant on tank wall
$\boldsymbol{\tau}$	Deviatoric stress tensor
τ	Specific or interfacial energy
$\vec{\boldsymbol{\tau}}_{\text{c}}$	Total control torque
$\vec{\boldsymbol{\tau}}_{\text{fb}}$	Feedback control torque
$\vec{\boldsymbol{\tau}}_{\text{ff}}$	Feedforward control torque
θ_0	Static contact angle
$\vec{\mathbf{u}}$	Velocity
$\vec{\mathbf{u}}_{s/c}$	Velocity of spacecraft's center of mass
$\tilde{\mathbf{u}}$	Modified transport velocity
V	Volume
W	Interpolation kernel
$\vec{\mathbf{x}}$	Position
$\vec{\mathbf{x}}_{s/c}$	Position of spacecraft's center of mass

Chapter 1

Introduction

A sloshing propellant inside a partially filled tank is a major disturbance for a spacecraft [1, 13, 28, 69, 70]. The propellant induces disturbance forces and torques on the tank and thus on the spacecraft structure, which need to be counteracted by the spacecraft's Attitude & Orbit Control System (AOCS) through its actuators like thrusters or reaction wheels. The impact of propellant sloshing disturbances reaches from performance degradations, which may lead to unfeasible spacecraft mission objectives, up to an unstable spacecraft, eventually even causing a loss of mission (see section 1.1).

Thus it is of utmost importance in spacecraft design to minimize the effect of a sloshing propellant. This is achieved on the one hand passively by using anti-sloshing or propellant management devices like baffles or flexible diaphragms, which are installed inside the tank to restrict the propellant and increase the damping of the sloshing motion. On the other hand, the spacecraft's AOCS is designed in such a way that it can actively counteract the sloshing effects anticipated throughout the entire spacecraft mission duration for a given operational range. In order to consider propellant sloshing in the AOCS design and its verification, a realistic model is required able to describe the propellant's dynamics during all phases of the spacecraft mission. While for linear propellant motion adequate sloshing models are available [13], the models and techniques currently available for describing nonlinear propellant sloshing, like wetting effects or high-velocity impacts, have significant shortcomings (see section 1.2).

In this thesis, a computational propellant sloshing model is presented capable of describing linear and nonlinear effects in both high-gravity and low-gravity environments. The derived computational model as well as its software implementation are specifically tailored to provide reasonable results also with lower spatial resolutions. This yields faster simulation runs, allowing the computational propellant sloshing model to be directly used in AOCS verification campaigns, where a very large number of high-fidelity simulations are run. This approach guarantees that the designed AOCS is able to cope with all kind of linear and nonlinear propellant sloshing effects occurring during the spacecraft mission duration.

1.1 Propellant sloshing as disturbance

Even though there are attempts to utilize a sloshing propellant for a good cause, as for example its potentially positive dissipative effect in flexible wing tanks of aircraft used to alleviate dynamic loads when the sloshing motion is considered in the modeling [19], the propellant sloshing consequences are usually considered as disturbance.

Figure 1.1 visualizes how a sloshing propellant inside a partially filled tank onboard a spacecraft becomes a disturbance. It starts with the Attitude & Orbit Control System (AOCS), which wants to bring the spacecraft into a desired rotational and translational state, e.g. during a rotational slew maneuver in a low Earth orbit or during the spacecraft's end-of-live disposal, and activates some actuators, e.g. a set of reaction wheels or thrusters, to achieve this. The AOCS actuators provide rotational and translational control accelerations acting on the spacecraft structure, thus changing its motional state. Since the partially filled tank is fixedly mounted on the spacecraft, the tank as well as the propellant inside it are also excited by these control accelerations. Depending on the type and magnitude of accelerations the actuators are producing, on the tank geometry and on tank materials used inside the tank as well as on the propellant's properties and its initial state, the propellent starts sloshing in a multitude of ways, showing linear or nonlinear behavior. During its sloshing motion, the propellant interacts with the tank and thus with the spacecraft structure, inducing disturbance forces and torques on the spacecraft. These disturbances then again lead to undesired changes in the spacecraft motional state, which are detected by the AOCS as deviations that need to be counteracted. The AOCS thus activates the actuators again to compensate the motional state errors, which closes the loop in figure 1.1.

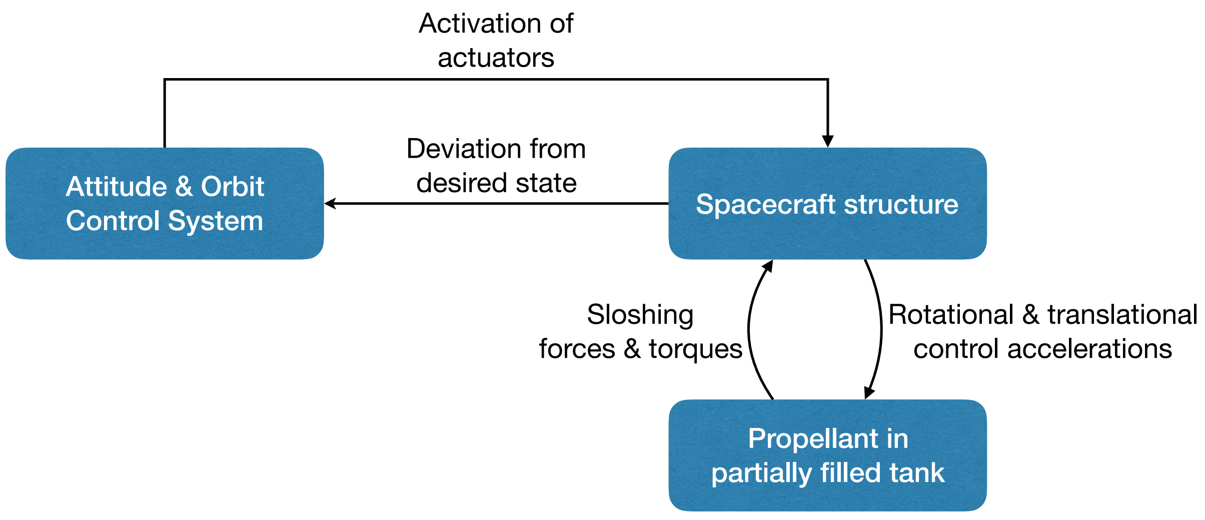

Activation of actuators
Deviation from desired state
Attitude & Orbit Control System
Spacecraft structure
Sloshing forces & torques
Rotational & translational control accelerations
Propellant in partially filled tank

Figure 1.1: Introduction: Visualization of the emergence of disturbances due to a sloshing propellant in a partially filled tank onboard a spacecraft and their consequences on the Attitude & Orbit Control System

One direct consequence of the increased actuator activity due to a sloshing propellant is the increased demand of electrical power in case of reaction wheels or propellant in case of thrusters. Depending on the magnitude and number of expected disturbances to compensate during a mission's lifetime, these additional resource requirements may lead to an unfeasible spacecraft mission design.

Another impact of propellant sloshing is the degraded pointing performance of a spacecraft. During a rotational slew maneuver for example, the additional disturbance due to the sloshing propellant leads to a reduced pointing accuracy. Also after the maneuver is completed, the propellant keeps on sloshing and is constantly being damped until it reaches a stable configuration. During this so-called settling time, the spacecraft is exposed to oscillatory motion and the accuracy of the spacecraft's attitude knowledge is reduced, thus for certain types of actions to be performed, like an image acquisition, the spacecraft mission needs to pause until the propellant is settled sufficiently.

A more dangerous implication a sloshing propellant can cause is an unstable attitude and orbit control. In case the real sloshing forces and torques acting on the spacecraft deviate from the expected disturbances, the AOCS may not be capable of counteracting the sloshing effects in a sufficient manner, yielding an unstable spacecraft. Depending on the extent of the deviation of expected and real disturbance forces and torques, the sloshing propellant may even lead to a loss of mission.

Examples of severe consequences a sloshing propellant may have on a spacecraft reach from over 50 years ago as for example with the ATS-5 spacecraft in 1969, where propellant sloshing caused a loss of control yielding an unplanned flat spin condition and thus a loss of most of the experiments [45, 44] up tp more recent issues as for example with the second demonstration flight of the Falcon 1 in 2007, where an upper stage anomaly due to propellant sloshing caused a loss of control, preventing it to reach its desired orbit [55].

It is thus of utmost importance to consider the disturbances due to the propellant's motion in spacecraft design. The propellant sloshing impact is usually minimized passively by using anti-sloshing or propellant management devices like baffles or flexible diaphragms, which are installed inside the tank to restrict the propellant and increase the damping of the sloshing motion [13]. The remaining disturbance forces and torques caused by the sloshing propellant need to be actively counteracted by the AOCS, making its design and verification two crucial tasks. The next section focusses on the existing models available to incorporate propellant sloshing effects in the design and verification on an AOCS and discusses their advantages and drawbacks.

1.2 Existing modeling approaches and shortcomings

In order to consider propellant sloshing in the AOCS design and its verification, a realistic model is required able to describe the propellant's motion during all phases of the spacecraft mission. Figure 1.2 visualizes the different modeling techniques currently used in literature, which are explained in the following.

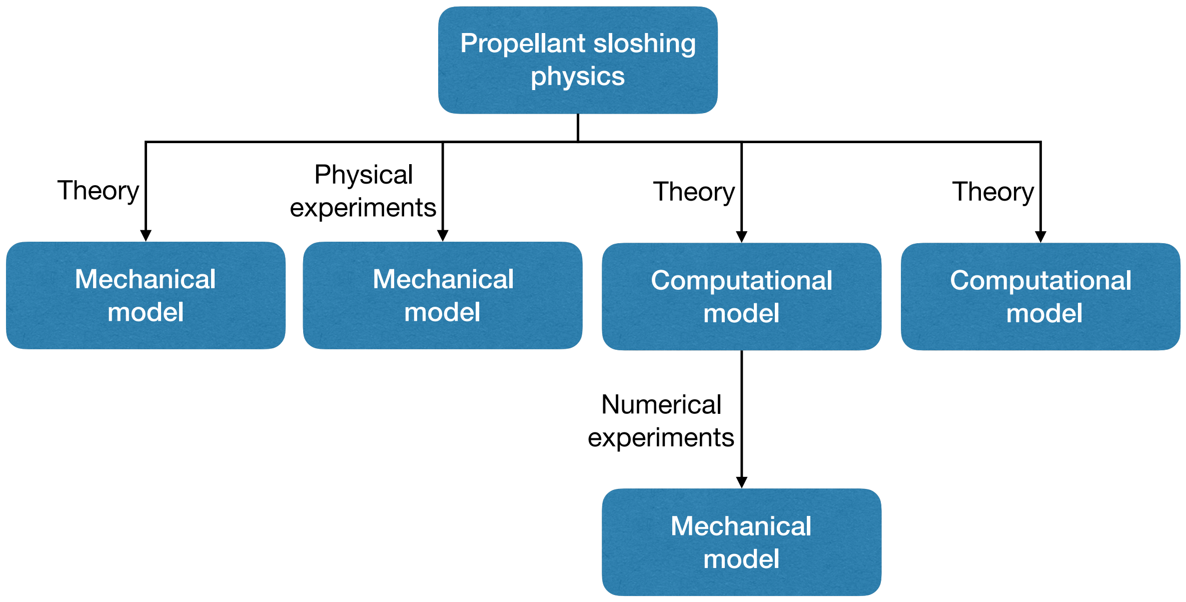


Figure 1.2: Introduction: Visualization of the propellant slosh modeling techniques currently used in literature including how they are derived from physics

For roughly 60 years, mechanical models are used as conceptual equivalent of the propellant for control and stability analyses due to the fact that they can be conveniently included in spacecraft AOCS design and verification due to their simple form [1, 13, 28]. The equivalent mechanical models typically consist of a set of pendulums or spring-mass systems whose equations of motions are used to reproduce the disturbance forces and torques of a sloshing propellant undergoing linear motion in a partially filled tank with rigid walls.

An equivalent mechanical model may be derived from theory (figure 1.2, very left column) by solving special cases of the Navier-Stokes equations for simplifying assumptions like linear sloshing motion in a high-gravity environment, rigid tank walls, a dedicated simple tank geometry and only axis-symmetric accelerations acting on the spacecraft [1, 13, 28]. The advantage of this approach is the very low computational cost due to a simple formula providing sloshing forces and torques. Furthermore, the mechanical model is solely based on theory, thus doesn't require any physical or computational experiments to be performed. A drawback of this approach is that many (simplifying) assumptions are made that may not hold in reality and that may not be applicable to all desired configurations, like for example a dedicated tank geometry for which no special-case solution of the Navier-Stokes equations exists. The main drawback of this approach though is that only linear sloshing motion is reproduced reasonably well, nonlinearities are captured inaccurately or, even worse, completely neglected.

The parameters of an equivalent mechanical model may also be determined entirely or partly by use of experimental data (figure 1.2, second from left column) [1, 13, 28]. One approach is to derive a mechanical model based on theory and use physical experiments of the desired specific tank and propellant to determine parts of the model parameters, which would have been difficultly or impossibly obtained otherwise, as for example the system's damping ratio. Another approach is to perform physical experiments of the desired tank/propellant configuration and fit the entire mechanical model to the experimental results. The latter approach is used especially if a theoretical derivation of the model is infeasible, as for example in case of a flexible membrane installed inside the tank, or impractical as for example in case of a tank subdivided into multiple irregular compartments. This is also the major advantage of this approach. The main drawback is again the fact that only linear sloshing motion is considered with the resulting mechanical model itself.

For many current and future spacecraft missions the requirements on agility, pointing performance and settling time become more demanding [6], mainly yielding nonlinearities which cannot or only hardly be described with equivalent mechanical models [23]. Increased agility typically involves nonlinear sloshing motion including arbitrary propellant impacts on the tank wall. When considering higher pointing performance, low-gravity sloshing disturbances like surface tension or wetting effects may start playing a major role. And a reduced settling time, which can be achieved through a better knowledge of the sloshing motion and the location of the propellant, is important for example for maximizing the number of pictures taken during an orbit of an Earth observation satellite mission. All these points make an equivalent mechanical model unsuitable for the proper description of nonlinear sloshing motion. Already in [1] in 1966 the usage of computational methods for propellant slosh modeling was discussed. These numerical methods have been significantly improved over the decades, so that a multitude of different simulation techniques is available to predict nonlinear sloshing, for example grid-based methods like the finite element method (FEM) [51], the finite volume method (FVM) [66] or the boundary element method (BEM) [9], as well as meshfree methods like the smoothed particle hydrodynamics method (SPH) [23].

Especially for cases where performing physical experiments would be difficult, another approach to derive an equivalent mechanical model is based on a computational sloshing model (figure 1.2, second from right column) [13, 18, 28]. Here a computational model is used to simulate the desired real tank/propellant configuration in combination with the anticipated rotational or translational spacecraft maneuvers that excite the propellant. As described above already for physical experiments, an entire mechanical model is then fitted to these numerical simulation results. The advantage of this technique is the reduced financial cost that a numerical simulation produces as compared to physical experiments. The main disadvantage is again the missing capability of the resulting mechanical model to reproduce nonlinear sloshing motion.

Instead of using a computational model for deriving an equivalent mechanical model, the computational model can also be directly used in AOCS design and verification (figure 1.2, very right column) [37]. Dedicated simulations are performed with this computational model, determining important parameters relevant for the AOCS design like the sloshing frequencies. Additionally, during AOCS verification campaigns, where a very large number of high-fidelity simulations are performed, the computational model may be coupled to the spacecraft's rigid body [20, 66] and AOCS simulator [18, 37] in order to verify that the developed controller is capable of coping with the effects caused by the sloshing propellant. The major advantage of this approach is that the disturbances due to sloshing forces and torques are not restricted anymore to linear motion. Every modeled nonlinearity can be captured well using the computational model directly in the loop with the AOCS. The main drawback of computational models is the large computational cost, making it impossible to perform the required large amount of simulations in a reasonable amount of time. Additionally there are major issues in properly modeling highly nonlinear sloshing motion like high-velocity impacts or low-gravity effects like surface tension or wetting [6, 17].

1.3 Research objectives and contributions

Summarizing the review in the previous section, the propellant sloshing models currently available in literature are mostly mechanical analog models that mainly describe linear sloshing motion and thus lack the possibility to be used for all spacecraft mission phases, especially where nonlinearities in the propellant dynamics occur and sometimes even violent flows prevail. The numerical methods used in currently available computational propellant sloshing models are principally able to reproduce all kind of nonlinearities in the sloshing motion. But the extent to which nonlinearities can then be really simulated depends on the implemented model. Here, currently available computational models lack the possibility to properly describe violent sloshing motion like high-velocity impacts on the tank wall or generic wetting effects, for example also including contact angle hysteresis. Finally, the high computational cost of the currently existing computational models makes it impossible to use these models in AOCS verification campaigns, simply due to the fact that the required very large number of high-fidelity simulations couldn't be performed in a reasonable amount of time.

Contributing to the incorporation of nonlinear slosh modeling in the AOCS design and verification in order to improve spacecraft performance and stability is the motivation of this work. The main objective of this thesis is thus to develop a computational propellant sloshing model able to capture nonlinear sloshing effects typically arising in spacecraft missions. A second objective is to design the computational propellant sloshing model as well as its software implementation in such a way that it provides substantially shorter simulation runtimes than traditionally used computational models, so that it can be directly used in AOCS verification campaigns instead of a mechanical analog model. Additionally, the usability of the software implementation shall be maximized by providing a simulation environment that can be conveniently used without expert knowledge in computational fluid dynamics, especially with respect to simulation setup and post-processing of the simulation results. Finally a third objective is to validate the application of the presented computational propellant sloshing model to AOCS verification simulations regarding nonlinear propellant sloshing in high-gravity and low-gravity environments.

Based on these research objectives, the main contributions of the thesis at hand are outlined in the following and described in detail in the remainder of this work:

1. **Development of a computational propellant sloshing model able to reproduce nonlinear sloshing effects typically arising in spacecraft missions**
   A main contribution of this work is the development of a computational propellant sloshing model that is capable of describing highly nonlinear sloshing motions like high-velocity impacts on tank walls as well as surface tension or wetting effects. The computational model described in this work is able to reproduce linear as well as nonlinear sloshing dynamics and can thus be used for all phases of a spacecraft mission and for both high-gravity and low-gravity environments as well as for transient phases from one environment to the other.

2. **Optimization of the computational propellant sloshing model as well as its software implementation for shorter simulation runtimes and better usability**
   Another major contribution of the thesis at hand is the runtime-optimized design and software implementation of the computational propellant sloshing model. The computational method described in this work is capable of simulating problems using a low spatial resolution while still providing global system properties like the integrated forces and torques exerted by the propellant on the tank structure with a sufficient accuracy. The software implementation of the computational propellant sloshing model uses a lightweight parallelization technique that allows to conveniently parallelize the simulation calculations

without adding too much overhead when using low resolutions. An additional contribution is the increased usability of the software implementation. The software is designed in a way that it provides a convenient way of pre-processing including geometry creation as well as post-processing comprising data visualization, data analysis and export of simulation results. All parameters used to setup a simulation have a real physical meaning, like e.g. dynamic viscosity or surface tension, and no specific computational fluid dynamics expert knowledge is required in order to configure and run a simulation.

3. **Validation of the computational propellant sloshing model for AOCS verification simulations**
   A third main contribution of this thesis is the validation of the applicability of the proposed computational propellant sloshing model to AOCS verification simulations. A generic spacecraft AOCS simulator is developed capable of reproducing the attitude and orbit dynamics and control of a spacecraft and subsequently coupled to the propellant sloshing simulator. Two coupled simulations are performed covering applications of nonlinear propellant sloshing in high-gravity and zero-gravity conditions.

1.4 Outline

The thesis at hand begins in chapter 2 with the description of the computational propellant sloshing model. First, the physics behind propellant sloshing is explained, following an introduction to Smoothed Particle Hydrodynamics (SPH), a computational method used for simulating the dynamics of the propellant. Then the details of the computational model are presented comprising the spatial and temporal discretization of the involved theory.

Chapter 3 discusses the software implementation of the previously described computational propellant sloshing model. Starting with the specification of software requirements, the design of the software is presented including its architecture, data structures, surrounding tools for pre-processing and post-processing as well as the workflow of a propellant sloshing simulation, explaining the implemented algorithm.

In chapter 4, the computational propellant sloshing model is validated for high-gravity environments. Hydrostatic and dynamic sloshing simulations are performed and their results compared to theory, experiments and other numerical simulations. Additionally, an analysis is performed on how the global system properties like integrated forces and torques exerted by the propellant on the tank structure behave under reduced spatial resolution.

The next chapter 5 then validates the computational propellant sloshing model for zero-gravity environments. The validation comprises simulations of surface tension as well as wetting effects and compares the simulation results to theoretically determined values. Finally also the low-gravity environmental effects are assessed with respect to a reduced spatial resolution.

In order to investigate the usage of the derived computational propellant sloshing model for Attitude & Orbit Control System (AOCS) verification, first a spacecraft AOCS simulator needs to be implemented. Thus in chapter 6 the theory behind spacecraft attitude & orbit dynamics and control is presented including multiple temporal discretization techniques of the resulting equations of motion. Then the details of typical AOCS software and hardware components are given and a three-axis Proportional-integral-derivative (PID) attitude controller suitable for agile spacecraft missions is presented.

Chapter 7 contains the software implementation of the computational spacecraft attitude & orbit dynamics and control model. Starting again with the specification of software requirements, the design of the software is presented including its architecture, data structures, pre-processing and post-processing tool as well as the class structure of the resulting spacecraft AOCS simulator. Additionally, a section on the validation of the previously presented computational spacecraft attitude dynamics & control model is included.

In chapter 8 follows the presentation of the coupling of propellant sloshing and spacecraft attitude dynamics & control. First, the coupling strategy is discussed, deriving the equations needed for the coupling. Then, the software implementation of the coupling is shown, detailing how the propellant sloshing simulator and spacecraft AOCS simulator are interconnected.

Chapter 9 finally presents the application of the two previously developed propellant sloshing and spacecraft AOCS simulators to the AOCS verification of nonlinear propellant sloshing. The first case concentrates on a forced roll-motion in a high-gravity environment, comparing the simulation results to experiments. The second case of nonlinear sloshing simulates a typical Earth observation satellite mission scenario in zero-gravity conditions. The implications of the sloshing propellant on the AOCS are discussed including an error analysis. Also included in this chapter is an analysis of artificial sound waves, which are inherent to the used numerical method, how they impact the AOCS and how an influence on the AOCS can be minimized.

Finally this thesis closes with chapter 10, covering a summary of the presented work as well as an outlook on potential future work.

Chapter 2

Computational modeling of propellant sloshing using Smoothed Particle Hydrodynamics

2.1 Physics of propellant sloshing

The physics relevant for propellant sloshing typically involves a tank-propellant-gas system, where the tank is fixedly mounted on the spacecraft and is thus following its translational maneuvers (e.g. during an orbit raising), rotational maneuvers (e.g. during an image take for an Earth observation mission) or combination of both. The two fluids, i.e. propellant and gas, are contained in the tank and thus affected by the tank motion through its outer structure (i.e. the tank outer hull) and inner structure (like e.g. baffles or meshes), but can also freely float in the remaining tank volume. The environment for the dynamics of the tank-propellant-gas system may be characterized by high accelerations, e.g. during an agile rotational maneuver, or almost zero accelerations, e.g. on a stable orbit around Earth. Thus, especially for the fluids in a low-gravity environment this means that interfacial effects need to be taken into account. In the following, the dynamics of isothermal fluids in a tank-propellant-gas system is introduced. The rigid-body dynamics describing the translational and rotational motion of the satellite and hence the tank structure including its coupling with the propellant dynamics is described in chapter 8.

The motion of a fluid phase α is described by the Navier-Stokes equations, which read for an isothermal viscous fluid in a Lagrangian reference frame as [15]

$$\rho_\alpha \frac{d\vec{\mathbf{u}}_\alpha}{dt} = -\nabla p_\alpha + \nabla \cdot \boldsymbol{\tau}_\alpha + \hat{\mathbf{F}}_b \tag{2.1.1}$$

$$\frac{d\rho_\alpha}{dt} = -\rho_\alpha \nabla \cdot \vec{\mathbf{u}}_\alpha \tag{2.1.2}$$

$$\frac{d\vec{\mathbf{x}}_\alpha}{dt} = \vec{\mathbf{u}}_\alpha \tag{2.1.3}$$

with $\vec{\mathbf{u}}_\alpha$ being the velocity, ρ_α the density, p_α the pressure and $\vec{\mathbf{x}}_\alpha$ the position of fluid phase α, t is the time and $\hat{\mathbf{F}}_b$ the body force per unit volume. The deviatoric stress tensor $\boldsymbol{\tau}_\alpha$, i.e. the viscous part of the stress in the fluid, is given by $\boldsymbol{\tau}_\alpha = \left[\mu_\alpha(\nabla\vec{\mathbf{u}}_\alpha + \nabla\vec{\mathbf{u}}_\alpha{}^T)\right]$ with μ being the dynamic viscosity.

Equations 2.1.1 and 2.1.2 are subject to an impermeability boundary condition at the fluid-solid interfaces and initially each fluid is at rest with a prescribed initial density.

Especially in low-gravity conditions, the fluid's motion is driven by interfacial forces, which fundamentally consist of short-range cohesion forces (attraction of like molecules) and adhesion forces (attraction of dissimilar molecules) between the involved gas, propellant and tank molecules. The Navier-Stokes equations describe the dynamics inside a fluid phase, where the interfacial forces vanish. Since the net force of the interfacial effects is only non-zero at interfaces, it typically enters the Navier-Stokes equations through the boundary conditions.

The most important interfacial effects relevant for isothermal propellant sloshing in spacecraft are surface tension at propellant-gas interfaces, where the cohesion forces inside the propellant are stronger than the adhesion forces between propellant and gas, causing the liquid free-surface to tend towards its minimum state, as well as the wetting effects occurring in a tank-propellant-gas system like the contact angle at the triple-points of the solid-liquid-gas interface, which is also driven by the interplay of the involved cohesion and adhesion forces.

Regarding the boundary condition at fluid-fluid interfaces, the stress balance at the interface of two fluid phases α and β reads [8]

$$\vec{\mathbf{n}}(p_\alpha - p_\beta) - \vec{\mathbf{n}}(\boldsymbol{\tau}_\alpha - \boldsymbol{\tau}_\beta) = \vec{\mathbf{n}}\sigma_{\alpha\beta}\kappa - \nabla\sigma_{\alpha\beta} \tag{2.1.4}$$

with $p_{\alpha/\beta}$ and $\boldsymbol{\tau}_{\alpha/\beta}$ being the pressure and deviatoric stress tensor of the fluid phases α and β, respectively, $\vec{\mathbf{n}}$ being the normal vector pointing away from the α phase, $\sigma_{\alpha\beta}$ the interfacial tension (for a liquid-liquid system) or surface tension (for a liquid-gas system) between the two fluid phases α and β, $\kappa = \nabla \cdot \vec{\mathbf{n}}$ the curvature of the interface and $\nabla\sigma_{\alpha\beta}$ the tangential stress due to the gradients in the interfacial or surface tension. A sketch of the fluid-fluid interface visualizing the involved variables can be found in figure 2.1.

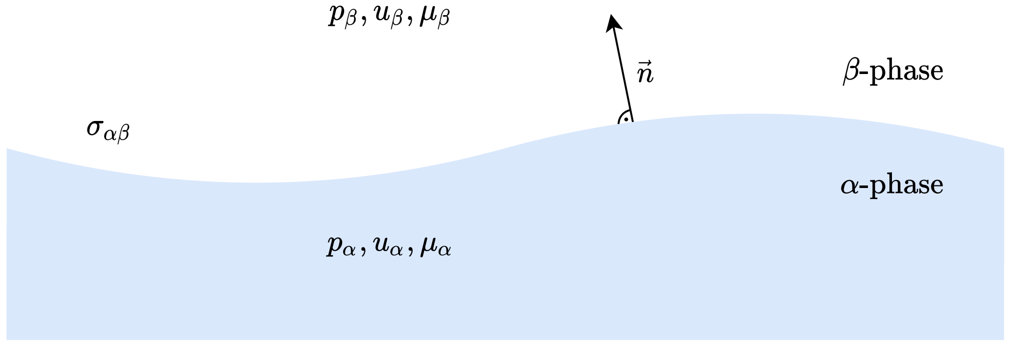

$p_\beta, u_\beta, \mu_\beta$
$\vec{n}$
β-phase
$\sigma_{\alpha\beta}$
α-phase
$p_\alpha, u_\alpha, \mu_\alpha$

Figure 2.1: Computational propellant sloshing modeling: Visualizing the variables involved in the stress balance at the interface of two fluid phases α and β with respective pressures $p_{\alpha/\beta}$, velocities $\vec{\mathbf{u}}_{\alpha/\beta}$ and dynamic viscosities $\mu_{\alpha/\beta}$ as well as the interfacial or surface tension $\sigma_{\alpha\beta}$ between the two fluid phases and the normal vector $\vec{\mathbf{n}}$ pointing away from the α phase

For solid-fluid-fluid interfaces the interplay of cohesion and adhesion forces between the involved phases determines the shape and dynamics of the contact line. Figure 2.2 shows a sketch of a wetting droplet, comprising a solid phase s as well as a liquid phase α and a gas phase β along with their interfacial energies between the fluid phases, i.e. the surface tension $\sigma_{\alpha\beta}$, between solid and liquid phase, $\sigma_{s\alpha}$, and between solid and gas phase, $\sigma_{s\beta}$.

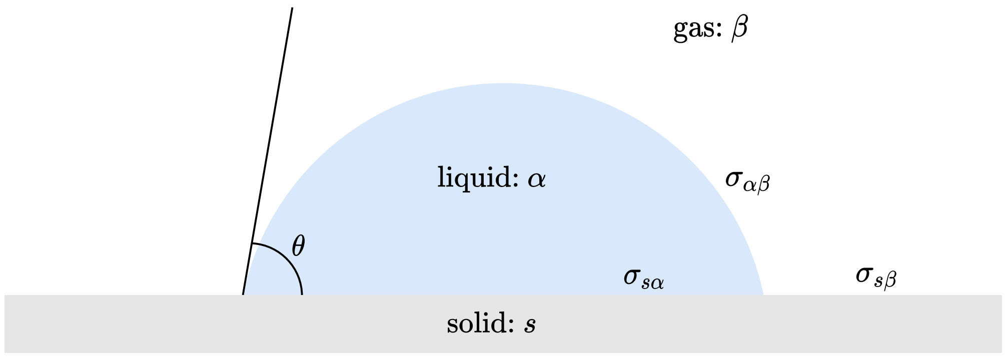


Figure 2.2: Computational propellant sloshing modeling: Visualizing a solid-liquid-gas interface with a solid phase s, a liquid phase α, a gas phase β along with their interfacial energies between the fluid phases, i.e. the surface tension $\sigma_{\alpha\beta}$, between solid and liquid phase, $\sigma_{s\alpha}$, and between solid and gas phase, $\sigma_{s\beta}$. Finally the contact angle θ between the solid-liquid interface and the liquid-gas interface, measured through the liquid, is shown.

A common way of quantifying the wettability of a solid surface by a liquid is the usage of the contact angle θ between the solid-liquid interface and the liquid-gas interface, measured through the liquid, as visualized in figure 2.2. For a system in equilibrium, assuming a flat rigid surface and immiscible fluids, the Young equation then describes the solid-liquid-gas interface at rest by [8]

$$\sigma_{\alpha\beta} \cos \theta_0 + \sigma_{s\alpha} = \sigma_{s\beta} \tag{2.1.5}$$

with θ_0 being the static contact angle. If the contact angle at a solid-liquid-gas interface is $\theta_0 < 90$ deg, the liquid is said to be wetting the solid surface. If the contact angle is $\theta_0 > 90$ deg, the liquid is said to be non-wetting or weakly wetting the solid surface.

Even though Young's equation is commonly used as boundary condition for the solid-liquid-gas interface, it shall be noted that it only applies for static conditions, i.e. fluids as well as solid at rest. In dynamic conditions, the model of a static contact angle needs to be amended since the moving solid-liquid-gas interface makes the contact angle dynamic as well, which is commonly expressed with advancing and receding contact angles and denoted as contact angle hysteresis.

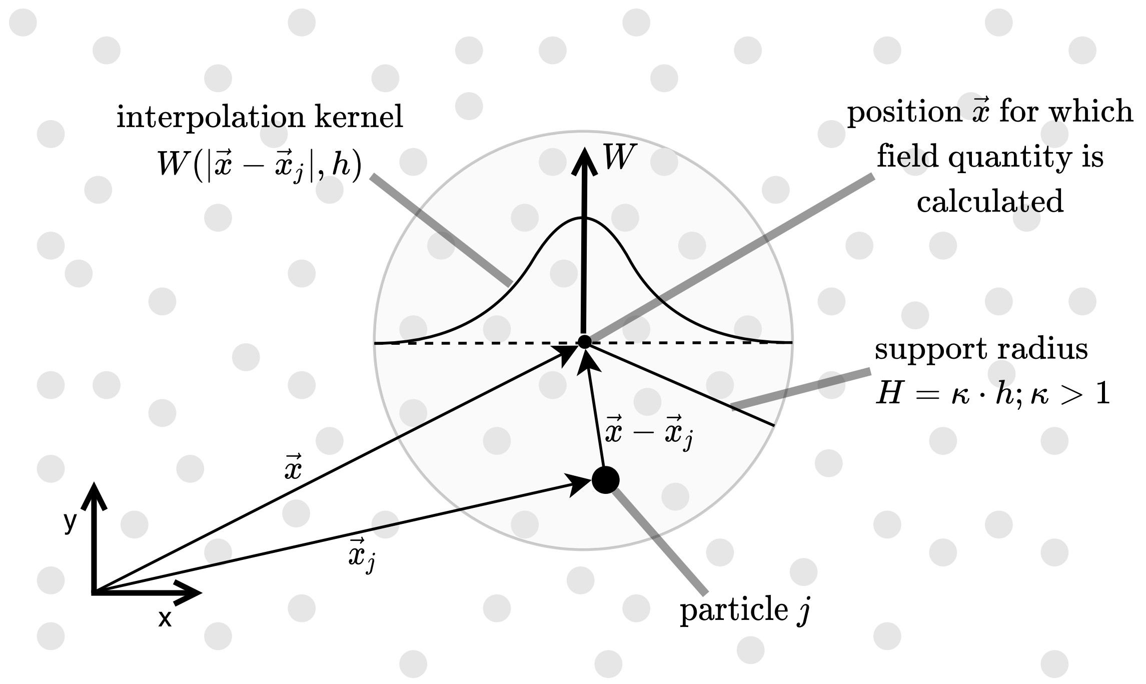
2.2 Smoothed Particle Hydrodynamics approximation

The spatial discretization of the tank-propellant-gas system is done by use of the Smoothed Particle Hydrodynamics (SPH) method. Initially formulated independently by Lucy [36] as well as Gingold and Monaghan [21] for solving astrophysical problems in 1977, the SPH method has been used since then for many applications in various fields [39, 67]. Smoothed Particle Hydrodynamics is a meshfree Lagrangian method that divides the problem domain into a finite set of discretization points, referred to as particles, which carry properties of the respective solid, fluid or gas phase and can freely move with the bulk velocity of the respective phase.

Using the SPH method, a field quantity ϕ at a certain location $\vec{\mathbf{x}}$ in the problem domain is calculated by use of its surrounding particles j at locations $\vec{\mathbf{x}}_j$ as [40]

$$\phi(\vec{\mathbf{x}}) = \sum_j \frac{m_j}{\rho_j} \phi(\vec{\mathbf{x}}_j) W(|\vec{\mathbf{x}} - \vec{\mathbf{x}}_j|, h) \tag{2.2.1}$$

with m_j and ρ_j being the mass and density, respectively, of a neighboring particle j and $W(|\vec{\mathbf{x}} - \vec{\mathbf{x}}_j|, h)$ a weight for the contribution of the neighboring particle's field quantity value $\phi(\vec{\mathbf{x}}_j)$. $W(|\vec{\mathbf{x}} - \vec{\mathbf{x}}_j|, h)$ is referred to as interpolation kernel and h the so-called smoothing length, which is a measure for the width of the kernel. An illustration of the SPH approximation and the parameters involved can be found in figure 2.3.



interpolation kernel
$W(|\vec{x} - \vec{x}_j|, h)$
W
position $\vec{x}$ for which field quantity is calculated
support radius
$H = \kappa \cdot h; \kappa > 1$
$\vec{x} - \vec{x}_j$
$\vec{x}$
$\vec{x}_j$
y
x
particle j

Figure 2.3: Computational propellant sloshing modeling: Illustration of the SPH approximation for a field quantity at position $\vec{\mathbf{x}}$, which is calculated by looping over all neighboring particles j within the support radius $H = \kappa h$, $\kappa > 1$ and summing the field quantities for the respective particle positions $\vec{\mathbf{x}}_j$ weighted with the interpolation kernel function $W(|\vec{\mathbf{x}} - \vec{\mathbf{x}}_j|, h)$ according to equation 2.2.1. Please note that for particles outside the support radius H, the interpolation kernel with compact support returns zero, hence these particles do not contribute to the field quantity at position $\vec{\mathbf{x}}$.

The gradient of a scalar field quantity ϕ and the divergence of a vector field quantity $\vec{\phi}$ can then be calculated using a formulation that is zeroth-order consistent, i.e. that the derivative of a constant is exactly zero, as [10]

$$\nabla\phi(\vec{\mathbf{x}}) = \sum_j \frac{m_j}{\rho_j} \left(\phi(\vec{\mathbf{x}}_j) - \phi(\vec{\mathbf{x}})\right) \nabla W(|\vec{\mathbf{x}} - \vec{\mathbf{x}}_j|, h) \tag{2.2.2}$$

$$\nabla \cdot \vec{\phi}(\vec{\mathbf{x}}) = \sum_j \frac{m_j}{\rho_j} \left(\vec{\phi}(\vec{\mathbf{x}}_j) - \vec{\phi}(\vec{\mathbf{x}})\right) \cdot \nabla W(|\vec{\mathbf{x}} - \vec{\mathbf{x}}_j|, h) \tag{2.2.3}$$

using the derivative of the interpolation kernel $\nabla W(|\vec{\mathbf{x}} - \vec{\mathbf{x}}_j|, h)$.

There exist multiple other gradient and divergence expressions used in the SPH framework, offering higher order consistency as well as conservation of linear and angular momentum, that are applied depending on the desired and needed system properties (see e.g. [4, 10, 67].

The interpolation kernel $W(|\vec{\mathbf{x}} - \vec{\mathbf{x}}_j|, h)$ is basically free to choose, but needs to be continuously differentiable and additionally fulfill two hard requirements. On the one hand, for vanishing smoothing lengths h the kernel needs to reduce to a delta function, thus providing an exact interpolation in the continuous case,

$$\lim_{h\to 0} W(|\vec{\mathbf{x}} - \vec{\mathbf{x}}_j|, h) = \delta(|\vec{\mathbf{x}} - \vec{\mathbf{x}}_j|) \tag{2.2.4}$$

On the other hand, the exact interpolation of constants requires the partition of unity property to be fulfilled,

$$\int W(|\vec{\mathbf{x}} - \vec{\mathbf{x}}_j|, h) d\vec{\mathbf{x}}_j = 1 \tag{2.2.5}$$

Additionally, for computational efficiency it is desirable to choose an interpolation kernel with compact support,

$$W(|\vec{\mathbf{x}} - \vec{\mathbf{x}}_j|, h) = 0 \text{ for } |\vec{\mathbf{x}} - \vec{\mathbf{x}}_j| > H \tag{2.2.6}$$

with $H > h$ being the support radius of the interpolation kernel.

There exist multiple common choices for interpolation kernels [11], of which for the simulations performed in this work two kernels are mainly used. The first one is a cubic spline kernel, which is especially suitable for simulations with a low number of kernel particles, as it is for example done for low-resolution simulations of high-gravity environments (see e.g. section 9.1). The cubic spline kernel has a compact support with support radius set to $H = 1.825742h$ (following [11]) and with $r = \frac{|\vec{\mathbf{x}} - \vec{\mathbf{x}}_j|}{H}$ it is given in 3D space along with its derivative as

$$W(r, H) = \begin{cases} 0 & ; r \geq 1 \\ \frac{16}{\pi} H^{-3} (1-r)^3 & ; 0.5 \leq r < 1 \\ \frac{16}{\pi} H^{-3} \left[(1-r)^3 - 4(0.5-r)^3\right] & ; r < 0.5 \end{cases} \tag{2.2.7}$$

$$\nabla W(\vec{\mathbf{x}} - \vec{\mathbf{x}}_j, H) = \begin{cases} \vec{\mathbf{0}} & ; r \geq 1 \\ -\frac{48}{\pi} H^{-5} \frac{1}{r} (1-r)^2 \cdot (\vec{\mathbf{x}} - \vec{\mathbf{x}}_j) & ; 0.5 \leq r < 1 \\ -\frac{48}{\pi} H^{-5} \frac{1}{r} \left[(1-r)^2 - 4(0.5-r)^2\right] \cdot (\vec{\mathbf{x}} - \vec{\mathbf{x}}_j) & ; r < 0.5 \end{cases} \tag{2.2.8}$$

The second interpolation kernel used in this work is the Wendland C^2 kernel whose resistance against the pairing instability for higher number of kernel particles is its major advantage. It is thus used in simulations involving a higher number of kernel particles, required especially for properly resolving interfacial effects in low-gravity conditions (see chapter 5). The Wendland C^2 kernel has a compact support with support radius set to $H = 1.936492h$ (following [11]) and with $r = \frac{|\vec{\mathbf{x}}-\vec{\mathbf{x}}_j|}{H}$ it is given in 3D space along with its derivative as

$$W(r,H) = \begin{cases} 0 & ; r \geq 1 \\ \frac{21}{2\pi} H^{-3} \left[(1-r)^4(1+4r)\right] & ; 0 \leq r < 1 \end{cases} \tag{2.2.9}$$

$$\nabla W(\vec{\mathbf{x}} - \vec{\mathbf{x}}_j, H) = \begin{cases} \vec{\mathbf{0}} & ; r \geq 1 \\ -\frac{210}{\pi} H^{-5} (1-r)^3 \cdot (\vec{\mathbf{x}} - \vec{\mathbf{x}}_j) & ; 0 \leq r < 1 \end{cases} \tag{2.2.10}$$

In the following, the advantages and drawbacks of the SPH method are briefly discussed.

The main advantage of the SPH method is its meshfree Lagrangian nature. The SPH method is able to easily deal with complex moving and highly deforming geometries as well as multiple phases, providing naturally the respective interfaces without additional cost. Especially for a tank-propellant-gas system this means that no tracking or reconstruction of the solid-liquid, solid-gas and liquid-gas interfaces needs to be done, which is often a big issue for Eulerian methods causing noisy field quantities at the interfaces (see e.g. [6]). Thus the SPH method is specifically useful for simulating highly nonlinear system behaviors like inertia-dominated high-velocity impacts as well as interfacial forces-dominated wetting phenomena.

By discretizing the entire problem domain with particles that are advected with the respective material velocities, different phases can be easily distinguished by assigning a simple integer value. The coupling of solid, liquid and gas particles as well as the coupling of the rigid-body motion of the satellite and the tank-propellant-gas system can then be easily achieved by applying the corresponding physics to the respective particles of the proper integer value type.

Since SPH is a meshfree method, also large areas without any material can be handled efficiently, since they are just not discretized at all. This is useful for example when simulating free-surface propellant-tank systems with a low filling ratio, where most of the tank is devoid of propellant. Here, the propellant only constitutes a small fraction of the tank volume, but due to the manifold attitude and orbit maneuvers a satellite may perform, the propellant could be located anywhere inside the tank.

Another advantage of the SPH method is that it has great conservation features as it naturally conserves mass and linear momentum as well as angular momentum when using appropriate discrete derivatives and interpolation kernels [67].

Regarding the drawbacks of the SPH method, despite the many successful applications of the method to various complex problems (see e.g. [39, 67] for an overview) and its clear superiority simulating systems with multiple phases involving complex moving and highly deforming geometries as compared to conventional, well-established methods like the finite element method or the finite volume method, the SPH method still lacks a consistent mathematical theory regarding convergence, consistency and stability.

In order to address this shortcoming, the SPH community SPHERIC[1] defined a "SPH Grand Challenge" [65] that aims at continuously building up and spreading knowledge regarding SPH convergence, consistency and stability.

Another difficulty with the SPH method is the handling of boundary conditions, especially regarding inflow and outflow boundary conditions [67]. This is mainly due to the fact that for particles close to a boundary their kernel support overlaps the boundary, which needs to be handled in a consistent manner. The SPH community SPHERIC is also working on the subject of boundary conditions in an organized way through a "SPH Grand Challenge" [65]. Please note that the applications considered in this work only require closed rigid tank structures, for which a proper solution regarding the boundary condition exists [2].

In order to solve the equations of motion for a particle in the SPH method, all the neighboring particles inside the kernel support need to be taken into account. Depending on the choice of the kernel as well as the physics involved, the number of neighboring kernel particles amounts to about 30 to 300 in 3D simulations, hence the computational cost is higher than for the calculations needed in the finite element method or finite volume method, for example. The fixed-radius near neighbor search itself as well as the calculations needed for one specific particle make SPH hence a computational more expensive method than other mesh-based formulations, especially for high-resolution simulations. On the other hand, as shown for example in [31] and also analyzed in detail in the validation chapters 4 and 5 of the work at hand, the SPH method allows to use really low resolutions for simulating propellant sloshing cases without affecting much the quality of the results, thus providing a very good overall computational performance.

[1]SPHERIC is the international organization representing the community of researchers and industrial users of Smoothed Particle Hydrodynamics.

2.3 Transport-velocity formulation for weakly-compressible SPH

A classical problem of the Smoothed Particle Hydrodynamics (SPH) method is the so-called tensile instability. This instability causes unphysical effects like particle clumping and void regions in the particle distribution and is mainly due to attractive forces between neighboring particles [42]. In order to account for the tensile instability in the proposed computational propellant sloshing model, the transport-velocity formulation of [3] and [72] is used. The main idea of the transport-velocity formulation is to use an additional correction acceleration for the particle transport that is based on a locally constant background pressure. This correction acceleration is used for calculating a modified particle transport velocity that causes a regularization of the particle distribution and thus suppresses the tensile instability.

The modified transport-velocity formulation is introduced by rewriting equations (2.1.1), (2.1.2) and (2.1.3) as [3, 72]

$$\rho_\alpha \frac{\tilde{d}\vec{\mathbf{u}}_\alpha}{dt} = -\nabla p_\alpha + \nabla \cdot \boldsymbol{\tau}_\alpha + \hat{\mathbf{F}}_b + \nabla \cdot \mathbf{A} \tag{2.3.1}$$

$$\frac{\tilde{d}\rho_\alpha}{dt} = -\rho_\alpha \nabla \cdot \tilde{\mathbf{u}}_\alpha \tag{2.3.2}$$

$$\frac{d\vec{\mathbf{x}}_\alpha}{dt} = \tilde{\mathbf{u}}_\alpha \tag{2.3.3}$$

with $\frac{\tilde{d}(\bullet)}{dt} = \frac{\partial(\bullet)}{\partial t} + \tilde{\mathbf{u}}_\alpha \cdot \nabla(\bullet)$ being defined as the material derivative of a fluid particle moving with modified transport velocity $\tilde{\mathbf{u}}_\alpha$ and an extra stress tensor $\mathbf{A} = \rho\vec{\mathbf{u}}_\alpha \otimes (\tilde{\mathbf{u}}_\alpha - \vec{\mathbf{u}}_\alpha)$ resulting from the equivalent reformulation of the momentum equation 2.1.1 using the modified transport velocity.

The modified transport velocity $\tilde{\mathbf{u}}_\alpha$ used to advect the fluid particles in equation 2.3.3, skipping the fluid phase indicator α from now on and using the index notation for the i-th particle in fluid phase α at time $(t + \Delta t)$ as $\tilde{\mathbf{u}}_i^{t+\Delta t}$, is calculated by

$$\tilde{\mathbf{u}}_i^{t+\Delta t} = \vec{\mathbf{u}}_i^t + \Delta t \left[\left(\frac{\tilde{d}\vec{\mathbf{u}}_i}{dt} \right)^t + \left(\frac{d\vec{\mathbf{u}}_i}{dt} \right)_c^t \right] \tag{2.3.4}$$

with $\left(\frac{d\vec{\mathbf{u}}_i}{dt}\right)_c$ being an additional correction acceleration for the particle transport that is based on a locally constant background pressure p_i^0 and given as

$$\left(\frac{d\vec{\mathbf{u}}_i}{dt} \right)_c = -\frac{1}{\rho_i} \nabla p_i^0 \tag{2.3.5}$$

This equation is discretized with the classical gradient approximation that is not zeroth-order consistent as

$$\left(\frac{d\vec{\mathbf{u}}_i}{dt} \right)_c = -\frac{1}{m_i} p_i^0 \sum_j (V_i^2 + V_j^2) \nabla_i W_{ij} \tag{2.3.6}$$

with V_i and V_j being the particles volumes and $\nabla_i W_{ij} = \frac{\partial}{\partial \vec{\mathbf{x}}_i} W(|\vec{\mathbf{x}}_i - \vec{\mathbf{x}}_j|, h)$ the gradient of the interpolation kernel. The locally constant background pressure p_i^0 depends on the particle pressure p_i and is given by $p_i^0 = \min(10|p_i|, \rho_0 c_s^2)$, where c_s is the artificial speed of sound as described below and ρ_0 the reference density.

Although the gradient of a constant vanishes in the continuous case, since the used gradient approximation given in equation 2.3.6 is not zeroth-order consistent, a residual force remains in the discrete case that regularizes the particle distribution and thus helps preventing the occurrence of the tensile instability.

In case the particles j inside the kernel support of a fluid particle i are uniformly distributed, the correction acceleration in equation 2.3.6 will vanish, $\left(\frac{d\tilde{\mathbf{u}}_i}{dt}\right)_c \approx 0$, and the modified transport velocity will equal the momentum velocity, $\tilde{\mathbf{u}}_i \approx \vec{\mathbf{u}}_i$, thus equations 2.3.1 to 2.3.3 of the modified transport-velocity formulation will translate into the original equations 2.1.1 to 2.1.3. On the other side, if the particles j inside the kernel support of a fluid particle i are inhomogeneously distributed, the correction acceleration will induce a regularization of the particle distribution through the modified transport velocity with a strength proportional to the local particle pressure p_i as described above.

Please also note that for a fluid particle i at a free-surface, where the particles j inside the kernel support of particle i are naturally distributed inhomogeneously, the particle pressure will vanish, $p_i \approx 0$, (see section 2.4), yielding a zero locally constant background pressure $p_i^0 = \min(10|p_i|, \rho_0 c_s^2) \approx 0$ and hence no correction acceleration will be applied, meaning that the free-surface is not disturbed by any undesired regularization.

Lastly, equations (2.3.1) and (2.3.2) are closed following the weakly-compressible approach [38] for simulating incompressible fluids with Smoothed Particle Hydrodynamics by use of a simple equation of state, relating pressure to density as

$$p_\alpha(\rho_\alpha) = c_s^2(\rho_\alpha - \rho_0) \tag{2.3.7}$$

with an artificial speed of sound c_s and the reference density ρ_0. The speed of sound c_s determines the maximum compressibility of the fluid and is chosen such that the density variation of the fluid is less than a prescribed value, which is usually below 1%. The maximum density $\rho_{\max}$ will be reached when the maximum pressure $p_{\max}$ occurs during the simulation. Inserting the desired prescribed maximum compressibility $\kappa_{\max} = \frac{\rho_{\max} - \rho_0}{\rho_0}$ into equation 2.3.7 finally yields a formula allowing to set the speed of sound based on the desired maximum compressibility as

$$c_s = \sqrt{\frac{p_{\max}}{\rho_0 \cdot \kappa_{\max}}} \tag{2.3.8}$$

2.4 Discretization of continuity equation

The density of a fluid particle in a simulation using Smoothed Particle Hydrodynamics (SPH) is traditionally determined by either applying the density summation approach or the discretized form of the continuity equation [40].

Using the density summation approach, the density of a fluid particle i is calculated as [26]

$$\rho_i = m_i \sum_k W_{ik} \tag{2.4.1}$$

with $W_{ik} = W(|\vec{\mathbf{x}}_i - \vec{\mathbf{x}}_k|, h)$. This form conserves mass exactly also in multi-phase simulations, but has the disadvantage that particles close to the free-surface will have an unphysical low density due to their incomplete kernel support.

Using the continuity equation, a spatially discretized form of it can be derived in the framework of the transport-velocity formulation as [72]

$$\frac{\tilde{d}\rho_i}{dt} = \rho_i \sum_j \frac{m_j}{\rho_j} \tilde{\mathbf{u}}_{ij} \cdot \nabla_i W_{ij} \tag{2.4.2}$$

with $\tilde{\mathbf{u}}_{ij} = \tilde{\mathbf{u}}_i - \tilde{\mathbf{u}}_j$ being the particle's relative modified transport velocity and $\nabla_i W_{ij} = \frac{\partial}{\partial \vec{\mathbf{x}}_i} W(|\vec{\mathbf{x}}_i - \vec{\mathbf{x}}_j|, h)$ the gradient of the interpolation kernel. This approach can be directly applied to fluid particles at free-surfaces, but it has the drawback that errors in the density evolution will wind up, leading to instabilities in the simulation.

In order to get the advantages from both methods while simultaneously removing their drawbacks, the density of a fluid particle in this work is determined by combining the two approaches as proposed in [23]. Starting from the spatially discretized form of the continuity equation in transport-velocity formulation, the explicit Forward-Euler method is applied to equation 2.4.2, yielding after rearranging the density of a fluid particle i at time $(t + \Delta t)$ based on the midpoint values at time $(t + \frac{1}{2}\Delta t)$ as

$$\rho_i^{t+\Delta t} = \rho_i^{t+\frac{1}{2}\Delta t} \left(1 + \frac{\Delta t}{2} \left(\sum_j \frac{m_j}{\rho_j} \tilde{\mathbf{u}}_{ij} \cdot \nabla_i W_{ij} \right)^{t+\frac{1}{2}\Delta t} \right) \tag{2.4.3}$$

For preventing errors winding up using equation 2.4.3, the density of fluid particle i at the midpoint time $(t + \frac{1}{2}\Delta t)$ is determined using the density summation formula as shown above,

$$\rho_i^{t+\frac{1}{2}\Delta t} = m_i \sum_k (W_{ik})^{t+\frac{1}{2}\Delta t} \tag{2.4.4}$$

Since fluid particles close to a free-surface suffer from an incomplete kernel support, their density at the midpoint time is set to the reference density ρ_0 in case the density determined by the density summation formula is smaller than the reference density. The fluid particles in the vicinity of a free-surface are simply identified by checking if the number of particles in the kernel support is smaller than 95 % of the initial maximum kernel particles number.

This finally yields the particle densities ρ_i at time $(t + \Delta t)$ as a function of the positions and modified transport velocities at the midpoint time $(t + \frac{1}{2}\Delta t)$ as

$$\rho_i^{t+\Delta t} = \begin{cases} \left(m_i \sum_k W_{ik}\right)^{t+\frac{1}{2}\Delta t} \cdot \left(1 + \frac{\Delta t}{2} \sum_j \frac{\tilde{\mathbf{u}}_{ij} \cdot \nabla_i W_{ij}}{\left(\sum_k W_{jk}\right)}\right)^{t+\frac{1}{2}\Delta t} & ;\text{particle } i \text{ has full kernel support} \\ \rho_0 \left(1 + \frac{\Delta t}{2} \left(\sum_j \frac{m_j}{\rho_0} \tilde{\mathbf{u}}_{ij} \cdot \nabla_i W_{ij}\right)^{t+\frac{1}{2}\Delta t}\right) & ;\text{otherwise} \end{cases} \tag{2.4.5}$$

This combination of the discretized continuity equation with the density summation approach ensures that the density calculation for fluid particles close to a free-surface is done correctly while simultaneously initializing every time integration step the densities at the midpoint using the density summation formula, thus preventing simulation instabilities due to density error accumulation.

2.5 Discretization of momentum equation

The spatial discretization of the momentum equation (2.3.1) for a fluid particle i respecting the commonly used boundary conditions for liquid-gas interfaces as described in equation 2.1.4 and solid-liquid-gas interfaces as described in equation 2.1.5 is given by

$$m_i \frac{\tilde{d}\vec{\mathbf{u}}_i}{dt} = \vec{\mathbf{F}}_i^p + \vec{\mathbf{F}}_i^v + \vec{\mathbf{F}}_i^m + \vec{\mathbf{F}}_i^b + \vec{\mathbf{F}}_i^A \tag{2.5.1}$$

with $\vec{\mathbf{F}}_i^p$ and $\vec{\mathbf{F}}_i^v$ being the discretized versions of the pressure force and viscous force, respectively, which are given by [26] and [27] as

$$\vec{\mathbf{F}}_i^p = \sum_j -(V_i^2 + V_j^2)\tilde{p}_{ij}\nabla_i W_{ij} \tag{2.5.2}$$

$$\vec{\mathbf{F}}_i^v = \sum_j \tilde{\mu}_{ij}(V_i^2 + V_j^2)\frac{\vec{\mathbf{u}}_{ij}}{|\vec{\mathbf{x}}_{ij}|}\left(\nabla_i W_{ij} \cdot \frac{\vec{\mathbf{x}}_{ij}}{|\vec{\mathbf{x}}_{ij}|}\right) \tag{2.5.3}$$

with V_i and V_j being the particles volumes, $\vec{\mathbf{u}}_{ij} = \vec{\mathbf{u}}_i - \vec{\mathbf{u}}_j$ and $\vec{\mathbf{x}}_{ij} = \vec{\mathbf{x}}_i - \vec{\mathbf{x}}_j$ the particles relative velocity and position, respectively, $\tilde{p}_{ij} = \frac{\rho_j p_i + \rho_i p_j}{\rho_i + \rho_j}$ the density-weighted inter-particle averaged pressure and $\tilde{\mu}_{ij} = \frac{2\mu_i \mu_j}{\mu_i + \mu_j}$ a combined dynamic viscosity.

$\vec{\mathbf{F}}_i^b$ is the body force and $\vec{\mathbf{F}}_i^A$ the discretized version of the force resulting from the extra stress tensor $\mathbf{A}$ due to the transport-velocity formulation, and is given by

$$\vec{\mathbf{F}}_i^A = \sum_j (V_i^2 + V_j^2)\tilde{\mathbf{A}}_{ij}\nabla_i W_{ij} \tag{2.5.4}$$

with $\tilde{\mathbf{A}}_{ij} = \frac{\rho_j \mathbf{A}_i + \rho_i \mathbf{A}_j}{\rho_i + \rho_j}$ being the density-weighted inter-particle averaged extra stress tensor $\mathbf{A}$.

Pressure jumps across liquid-gas interfaces (eq. (2.1.4)) as well as wetting phenomena at solid-liquid-gas interfaces (eq. (2.1.5)) are effects caused by interactions of dissimilar (adhesion) and like (cohesion) molecules. As such, these physical phenomena are modeled on a macroscopic level by use of generic molecular-like pairwise interaction forces according to [58] as

$$\vec{\mathbf{F}}_i^m = \sum_j -F_{\alpha\beta}^m(|\vec{\mathbf{x}}_{ij}|)\frac{\vec{\mathbf{x}}_{ij}}{|\vec{\mathbf{x}}_{ij}|} \tag{2.5.5}$$

with $F_{\alpha\beta}^m(|\vec{\mathbf{x}}_{ij}|)$ being the magnitude of the interaction force acting between particle i of the α phase und particle j of the β phase. The major requirement on this interaction force is that it is long-range attractive and short-range repulsive, so it is chosen to be [58]

$$F_{\alpha\beta}^m(|\vec{\mathbf{x}}_{ij}|) = s_{\alpha\beta}|\vec{\mathbf{x}}_{ij}|\left[-A \cdot e^{-\frac{|\vec{\mathbf{x}}_{ij}|^2}{2\epsilon_0^2}} + e^{-\frac{|\vec{\mathbf{x}}_{ij}|^2}{2\epsilon^2}}\right] \tag{2.5.6}$$

with $\epsilon = H/3.5$, $\epsilon_0 = \epsilon/2$, $A = (\epsilon/\epsilon_0)^4$ and $s_{\alpha\beta}$ being the strength of the interaction force between particles of phase α and β.

It is important to note that this approach based on generic molecular-like cohesion and adhesion forces between like and dissimilar particles is able to describe effects like surface tension, static and dynamic contact angles, contact angle hysteresis etc. without any specific configuration, adjustments or model changes.

An open question remains on how to determine the interaction force strengths between particles of the respective phases for specific real-world applications. Originally these interaction force strengths needed to be calibrated manually for every single use case before any simulation could be performed [57, 31], what already led to good results on the one hand, but also made it too complicated for direct industrial applications due to the calibration overhead on the other hand. Finally, in [58] a mathematical derivation based on first principles was presented that explicitly maps the interaction force strengths of the model to the static contact angle and the dynamic surface tension.

This means that the user of the computational propellant sloshing model described in this work is able to easily setup the generic molecular-like pairwise interaction force model by just specifying the static contact angle and dynamic surface tension, which are commonly used to characterize a tank-propellant-gas system regarding cohesion and adhesion effects. These parameters, for which a large library of experimentally determined values for many fluids and solids already exists, is then used to automatically determine the interaction force strengths of the model. Then during the simulation the interplay of the involved cohesion and adhesion forces of the respective phases will produce the desired surface tension and wetting effects.

When modeling a propellant and gas inside a tank onboard a spacecraft, it can be usually assumed that the gas has no significant influence on the propellant's dynamics. Hence for the simulations in this work, without loss of generality, the gas-phase is not considered and only the propellant and tank wall are discretized.[2] In [58], the relationship between interaction force strengths and static contact angle and dynamic surface tension has been derived for a solid-liquid-gas system. Omitting the gas phase, this relationship has been established for a solid-liquid system in [43], whose approach is followed in this work.

The surface tension $\sigma_{\alpha\beta}$ at the interface of two phases α and β shall be produced by the molecular-like pairwise interaction forces as described in equation 2.5.5, what can be related to each other by [58]

$$\sigma_{\alpha\beta} = \tau_{\alpha\alpha} + \tau_{\beta\beta} - 2\tau_{\alpha\beta} \tag{2.5.7}$$

with $\tau_{\alpha\alpha}$ and $\tau_{\beta\beta}$ being the specific energies in phases α and β, respectively, as well as $\tau_{\alpha\beta}$ the interfacial energy between phases α and β given by

$$\tau_{\alpha\beta} = \frac{1}{8}\pi n_\alpha n_\beta \int_0^\infty z^4 F^m_{\alpha\beta}(z)dz = \lambda n_\alpha n_\beta s_{\alpha\beta} \tag{2.5.8}$$

with n_α and n_β being the particle number densities for the phases α and β, respectively, $F^m_{\alpha\beta}$ the interaction force from equation 2.5.6, z the direction perpendicular to an assumed flat interface, $\lambda = \pi(-A \cdot \epsilon_0^6 + \epsilon^6)$ a constant coming from the integration and $s_{\alpha\beta}$ the strength of the interaction force between particles of phase α and β.

Inserting equation 2.5.8 for energies $\tau_{\alpha\alpha}$, $\tau_{\beta\beta}$ and $\tau_{\alpha\beta}$ into equation 2.5.7 yields

$$\sigma_{\alpha\beta} = \lambda \left(n_\alpha^2 s_{\alpha\alpha} + n_\beta^2 s_{\beta\beta} - 2n_\alpha n_\beta s_{\alpha\beta}\right) \tag{2.5.9}$$

[2]Please note that the only difference between simulating a tank-propellant system and a tank-propellant-gas system using the computational propellant sloshing model at hand is the mapping of the interaction force strengths with static contact angle and dynamic surface tension. Hence if a simulation of all three phases is desired, one simply needs to exchange the interaction force strengths $s_{\alpha\alpha}$ and $s_{s\alpha}$ of this work with the interaction force strengths of the involved phases as given in [58].

For the free-surface simulations performed in this work, assuming that the surface tension $\sigma_{\alpha\beta}$ at the interface of the two phases α and β shall be produced solely by the liquid phase α, i.e. omitting the gas phase β by setting $s_{\beta\beta} = s_{\alpha\beta} = 0$, an expression for the interaction strength between liquid particles $s_{\alpha\alpha}$ in dependance of the surface tension $\sigma_{\alpha\beta}$ is obtained as

$$s_{\alpha\alpha} = n_\alpha^{-2} \frac{\sigma_{\alpha\beta}}{\lambda} \tag{2.5.10}$$

A formula for relating the static contact angle θ_0 that is formed at the triple-point of the tank-propellant-gas interface to the interaction strengths of the respective phases can be obtained by inserting equation 2.5.9 for the surface tension $\sigma_{\alpha\beta}$ between the liquid and gas phase, $\sigma_{s\alpha}$ between solid and liquid phase and $\sigma_{s\beta}$ between solid and gas phase into Young's equation 2.1.5, yielding

$$\cos\theta_0 = \frac{\sigma_{s\beta} - \sigma_{s\alpha}}{\sigma_{\alpha\beta}} = \frac{s_{\beta\beta} - 2s_{s\beta} - s_{\alpha\alpha} + 2s_{s\alpha}}{s_{\alpha\alpha} + s_{\beta\beta} - 2s_{\alpha\beta}} \tag{2.5.11}$$

Here it was assumed that the initial particle number density is the same for all phases, $n_\alpha = n_\beta = n_s$. Assuming further for the free-surface simulations at hand that the gas phase is omitted, $s_{\beta\beta} = s_{\alpha\beta} = s_{s\beta} = 0$, an expression for the interaction strength between solid and liquid particles $s_{s\alpha}$ in dependance of the static contact angle θ_0 and the previously derived interaction strength between liquid particles $s_{\alpha\alpha}$ can be found as

$$s_{s\alpha} = \frac{1}{2} s_{\alpha\alpha}(1 + \cos\theta_0) \tag{2.5.12}$$

Please note that although the interaction strength between solid and liquid $s_{s\alpha}$ is derived using the static contact angle in an equilibrium state, this interaction strength is also valid for dynamic conditions. The idea is to use a commonly available (measurable) parameter as the static contact angle as one solution to obtain the appropriate interaction strength. When the interaction strengths of all involved phases have been found, these can then be used also in dynamic conditions and naturally provide the observed dynamic effects like e.g. contact angle hysteresis due to the molecular-like pairwise interaction forces between particles of like and dissimilar phases.

2.6 Temporal discretization

The spatially discretized versions of the equations of motion as described in the previous sections are integrated in time using a kick-drift-kick form of the Leapfrog integration scheme based on [72], [2] and [23].

The time integration scheme for a fluid particle i starts by using the acceleration coming from the momentum equation 2.5.1 as well as the correction acceleration due to the transport-velocity formulation coming from equation 2.3.6 of the previous time step in order to calculate the modified transport velocity at the midpoint time as

$$\vec{\mathbf{u}}_i^{t+\frac{1}{2}\Delta t} = \vec{\mathbf{u}}_i^t + \frac{\Delta t}{2}\left(\frac{\tilde{d}\vec{\mathbf{u}}_i}{dt}\right)^t \tag{2.6.1}$$

$$\tilde{\vec{\mathbf{u}}}_i^{t+\frac{1}{2}\Delta t} = \vec{\mathbf{u}}_i^{t+\frac{1}{2}\Delta t} + \frac{\Delta t}{2}\left(\frac{d\vec{\mathbf{u}}_i}{dt}\right)_c^t \tag{2.6.2}$$

The positions of the fluid particles are integrated to the midpoint time using the previously calculated modified transport velocities and both midpoint time positions and modified transport velocities are then used to calculate the fluid particle densities at the new time step using equation 2.4.5,

$$\vec{\mathbf{x}}_i^{t+\frac{1}{2}\Delta t} = \vec{\mathbf{x}}_i^t + \frac{\Delta t}{2}\,\tilde{\vec{\mathbf{u}}}_i^{t+\frac{1}{2}\Delta t} \tag{2.6.3}$$

$$\rho_i^{t+\Delta t} = \rho_i\left(\vec{\mathbf{x}}_i^{t+\frac{1}{2}\Delta t}, \tilde{\vec{\mathbf{u}}}_i^{t+\frac{1}{2}\Delta t}\right) \tag{2.6.4}$$

Finally the positions of the fluid particles are integrated to the new time step by use of the modified transport velocities and the momentum equation 2.5.1 is used to calculate the momentum velocities at the new time step, thereby also completing the time integration cycle.

$$\vec{\mathbf{x}}_i^{t+\Delta t} = \vec{\mathbf{x}}_i^{t+\frac{1}{2}\Delta t} + \frac{\Delta t}{2}\,\tilde{\vec{\mathbf{u}}}_i^{t+\frac{1}{2}\Delta t} \tag{2.6.5}$$

$$\vec{\mathbf{u}}_i^{t+\Delta t} = \vec{\mathbf{u}}_i^{t+\frac{1}{2}\Delta t} + \frac{\Delta t}{2}\left(\frac{\tilde{d}\vec{\mathbf{u}}_i}{dt}\right)^{t+\Delta t} \tag{2.6.6}$$

Please note that although positions and velocities of the fluid particles are used at a midpoint time step, the computationally costly calculation of the fluid particles accelerations due to the momentum equation 2.5.1 is done only once per time step.

In a numerical simulation, the time step size Δt of the time integration needs to be limited due to stability reasons. An overview of commonly used time-stepping criteria for the Smoothed Particle Hydrodynamics (SPH) method can be found in [67].

In this work, the time step size Δt is first limited by the Courant–Friedrichs–Lewy (CFL) condition as

$$\Delta t \leq 0.25\,\frac{h}{c_s + |\vec{\mathbf{u}}_{\max}|} \tag{2.6.7}$$

with h being the smoothing length, c_s the artificial speed of sound and $\vec{\mathbf{u}}_{\max}$ the maximum velocity of the fluid particles.

Secondly, the time step size is limited due to the viscous condition as

$$\Delta t \quad \leq \quad 0.25 \, \frac{h^2}{\nu} \tag{2.6.8}$$

with $\nu = \mu/\rho_0$ being the kinematic viscosity.

Thirdly, a time step size restriction based on the body force is implemented as

$$\Delta t \quad \leq \quad 0.25 \, \left(\frac{h}{|\vec{\mathbf{a}}_{\max}|} \right)^{1/2} \tag{2.6.9}$$

with $\vec{\mathbf{a}}_{\max}$ being the maximum acceleration of the fluid particles due to body forces.

For the computational propellant sloshing model proposed in this work, a constant time step size is used throughout the entire simulation and the variables in equations 2.6.7 to 2.6.9 are estimated based on a priori knowledge. During the initialization of the simulation, the maximum fluid particles acceleration $\vec{\mathbf{a}}_{\max}$ is set by determining the maximum body acceleration for the entire simulation time, which are prescribed in the configuration of the simulation. The maximum velocity of the fluid particles $\vec{\mathbf{u}}_{\max}$ is then estimated by a free-fall through the longest dimension of the tank due to this maximum body acceleration. Since usually for the applications targeted with the proposed computational propellant sloshing model, i.e. Attitude and Orbit Control System (AOCS) verification campaigns, spacecraft orbit or attitude maneuvers are performed in a low-gravity environment that lead to an accelerated tank structure, which then induces fluid particle movement through pressure gradients, the maximum velocity of the fluid particles $\vec{\mathbf{u}}_{\max}$ also considers the acceleration of the spacecraft maneuvers performed. During the simulation run it is further more ensured that the fluid particles velocities and accelerations don't exceed the initially determined maximum values, otherwise an error is thrown and the user is informed about it.

2.7 Discretization of boundary conditions

For the propellant sloshing simulations performed in this work, as described in section 2.5 the gas phase has no significant influence on the propellant's dynamics and is thus simply not discretized at all, yielding void regions in the simulation domain. Hence two boundaries remain that need to be taken care of in the computations, i.e. the liquid-gas interface (or rather the liquid-void region interface) also denoted as free-surface as well as the solid-liquid interface.

Due to the Lagrangian nature of the used Smoothed Particle Hydrodynamics (SPH) method, the particles at the free-surface move with their momentum velocity, hence the kinematic boundary condition at the free-surface is implicitly fulfilled. Regarding the dynamic boundary condition presented in equation 2.1.4, which requires a continuous stress across the free-surface, in section 2.3 and 2.4 it is described that the reference density ρ_0 is prescribed for a fluid particle at a free-surface, yielding zero pressure from the equation of state 2.3.7. If surface tension shall be simulated, it is produced by the molecular-like pairwise interaction forces defined in section 2.5, so that the pressure jump across the free-surface due to surface tension becomes apparent in the total pressure calculated using the virial theorem as defined in equation 5.2.3.

Regarding the solid-liquid interface, the wall boundary condition of [2] is used. Every interface in SPH has the problem of incomplete kernel support for particles close to the boundary. For a solid-liquid interface this is solved in [2] through discretizing the solid wall by use of multiple layers of so-called dummy particles, which complete for a liquid particle close to a boundary the missing support of the kernel and mimic a continuous liquid phase. The wall particles thus contribute to the liquid particle's equation of motion and are initialized with the reference density $\rho_s = \rho_0$ and zero pressure $p_s = 0$, at the desired position given by the wall geometry $\vec{\mathbf{x}}_s$ and with the prescribed velocities $\vec{\mathbf{u}}_s$ and accelerations $\vec{\mathbf{a}}_s$. If now a liquid particle comes closer to a wall particle, its density increases through the continuity equation 2.4.5, leading to an increased pressure through the equation of state 2.3.7 and thus to a pressure force through the momentum equation 2.5.1, having as result that the liquid particle is being repelled from the solid wall represented by its constituting wall particles. Thus the impermeability boundary condition at the solid-liquid interface is implicitly enforced. Additionally a free-slip boundary condition is applied by simply not including the wall particles in the viscous force calculation in the momentum equation 2.5.1.

Since the wall particles are dummy liquid particles that contribute to the liquid particle's equation of motion, the pressure p_s and thus density ρ_s of the wall particles need to be extracted from the liquid under consideration of body forces as well as the motion of the solid wall. This is done by describing a force balance at the solid-liquid interface as [2]

$$\frac{d\vec{\mathbf{u}}_\alpha}{dt} = -\frac{\nabla p_\alpha}{\rho_\alpha} + \vec{\mathbf{a}}_b = \vec{\mathbf{a}}_s \tag{2.7.1}$$

with indices α and s referring to liquid and wall particles, respectively, $\vec{\mathbf{a}}_b$ being the acceleration due to body forces and $\vec{\mathbf{a}}_s$ the acceleration of the solid wall.

Rearranging equation 2.7.1 and integrating along the centerline of a single wall-liquid particle pair using the fundamental theorem for line integrals then yields

$$p_s = p_\alpha + \rho_\alpha \left(\vec{\mathbf{a}}_b - \vec{\mathbf{a}}_s\right) \cdot \vec{\mathbf{x}}_{s\alpha} \tag{2.7.2}$$

with $\vec{\mathbf{x}}_{s\alpha} = \vec{\mathbf{x}}_s - \vec{\mathbf{x}}_\alpha$ being the wall and liquid particle's relative position.

The effective pressure of a wall particle is calculated by taking into account all liquid particles in its kernel support weighted by the kernel, yielding

$$p_s = \frac{1}{\sum_\alpha W_{s\alpha}} \cdot \left[\sum_\alpha p_\alpha W_{s\alpha} + (\vec{\mathbf{a}}_b - \vec{\mathbf{a}}_s) \cdot \sum_\alpha \rho_\alpha \vec{\mathbf{x}}_{s\alpha} W_{s\alpha} \right] \tag{2.7.3}$$

Finally the density of the wall particle ρ_s is determined by use of the previously calculated pressure p_s and the equation of state 2.3.7 as

$$\rho_s = \frac{p_s}{c_s^2} + \rho_0 \tag{2.7.4}$$

Summarizing it can be said that through ρ_s and thus p_s the wall particles will contribute to the liquid's pressure gradient at the solid-liquid interface such that the force balance 2.7.1 is valid. Compared to other wall boundary conditions (an overview can be found in [67]), the solid wall boundary condition from [2] using dummy particles has the advantage to be easily applicable to complex geometries and that it can be conveniently used for coupling the solid-liquid system to a rigid-body model through the state variables $\vec{\mathbf{x}}_s$, $\vec{\mathbf{u}}_s$ and $\vec{\mathbf{a}}_s$ as is further described in chapter 8.

Please note that no dedicated boundary conditions regarding surface tension at the liquid-gas interface or wetting effects at the solid-liquid-gas interface need to be implemented, since those effects are implicitly handled through the injection of the molecular-like pairwise interaction forces into the momentum equation as described in section 2.5.

Chapter 3

Software implementation of the propellant sloshing model

3.1 Requirements specification

The computational propellant sloshing model described in chapter 2 has been developed to be used for an industrial application, i.e. the verification of a spacecraft's Attitude and Orbit Control System (AOCS). As such, the following requirements are derived for the software implementation of the computational propellant sloshing model:

R1 The software shall provide a convenient way of pre-processing, i.e. configuring the simulation including geometry creation.
Rationale: To be used for an industrial application, it needs to be assured that an engineer of the respective field, in the current case an AOCS engineer, is able to setup a propellant sloshing simulation without deeper knowledge of Computational Fluid Dynamics (CFD) tools, especially regarding geometry creation.

R2 All parameters needed to setup a simulation using the software shall have a real physical meaning, like e.g. dynamic viscosity or surface tension.
Rationale: The software shall not contain any "magical parameters", i.e. unphysical parameters like the artificial viscosity, which is been added to a simulation in order to account for numerical stability and whose value needs to be set before each simulation based on the specific application. The parameters used for the simulation setup shall be based on measurable variables that are available to the AOCS engineer.

R3 It shall be possible to couple the software comprising the computational propellant sloshing model to a rigid-body simulator.
Rationale: An AOCS engineer typically has access to an already existing spacecraft simulator including a rigid-body model. During an AOCS verification campaign it is desired to couple the spacecraft through its rigid-body formulation with the solid tank structure of the propellant sloshing model described in this work.

R4 The software shall run on a Linux operating system.
Rationale: AOCS verification campaigns, which comprise a large amount of simulations that can be performed mostly in parallel, are the main target of the software to be developed. Hence it is straightforward to run the software in a cloud infrastructure making use of its high scalability and on-demand availability. Multiple cloud providers and cloud computing service models exist, which all support the Linux operating system.

R5 When running the software, it shall be possible to perform the calculations in parallel.
Rationale: In order to reduce the real time needed to perform a simulation, the calculations being done by the software shall be executable in parallel as far as possible.

R6 The software shall run well with low spatial resolutions.
Rationale: As has been shown e.g. in [31] for the simulation of static contact angles, the main characteristics of a SPH simulation should not change too much by reducing the spatial resolution. Since the main target of the software to be developed is the verification of a spacecraft's AOCS, where a large number of simulations are performed, it would be beneficial to have a software optimized for low resolutions, which can then be performed in an even shorter amount of time.

R7 There shall be an easy to use post-processing tool for data visualization.
Rationale: Since the target users for the software to be developed are AOCS engineers, it shall be possible to conveniently visualize, analyze and export the simulation results without deep knowledge of CFD tools.

3.2 Design overview

Following the requirements in section 3.1, a software is written that implements the computational propellant sloshing model presented in chapter 2. The software implementation basically consists of a propellant sloshing simulator and a post-processing tool, which can be seen in figure 3.1 along with their interfaces.

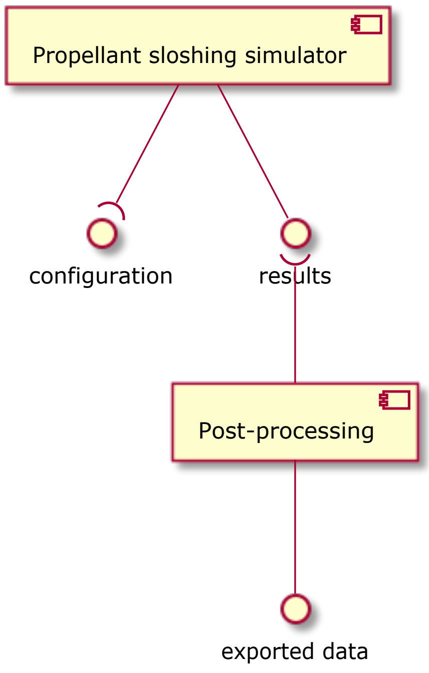


Figure 3.1: Propellant sloshing simulator: Components of the software implementation including interfaces.

The propellant sloshing simulator is a newly developed software written in C++ [56] and takes as input a configuration file. The configuration file comprises a class named *SPH::PARAMS*, which contains all needed simulation parameters. An excerpt of such a configuration file can be found in figure 3.2. In order to satisfy requirement [R1], no specific pre-processing steps are needed, which is especially relevant for the geometry creation. The geometry of the problem domain doesn't need to be created beforehand using any Computer-aided design (CAD) tool, creating models and discretizing them etc., but the software at hand offers predefined geometries offering the most relevant forms of spacecraft tanks including different ways of initial propellant distribution inside the tank. On the one hand this approach has the advantage of reducing the complexity of preparing a simulation significantly and also most of the users will suffice the tank forms and propellant distributions available. On the other hand, if yet another geometry is required, it can be easily added by amending the respective geometry creation class.

Further on, the configuration file only comprises parameters that have a real physical meaning, satisfying requirement [R2]. Despite the choice of an interpolation kernel to be used as well as the desired compressibility due to the weakly-compressible approach described in section 2.3, a user has no simulation-specific parameters to set but only needs to provide the variables relevant for describing the problem domain, e.g. tank size and fluid physical characteristics like the fluid density.

```
// Create new SPH parameter structure
SPH::PARAMS params;

// Name of the test case
params.testName = "phd_spheric_test10";

// Type of the tank:
//      1: Rectangular
//      2: Cylindrical
//      3: Spherical
//      4: Cylindrical with spherical domes
params.tankType = 1;

// Width/Diameter, depth and height of the tank [m]
params.tankWidth = 0.900;
params.tankDepth = 0.062;
params.tankHeight = 0.508;

// Fluid height at rest [m]
params.fluidHeight = 0.093;

// Fluid density [kg/m^3] and fluid dynamic viscosity [Pa*s]
params.fluidDensity = 998.0;
params.fluidDynamicViscosity = 8.94e-4;

// Definition of body accelerations like gravity
// (time [s], acc_b_x [m/s^2], acc_b_y [m/s^2], acc_b_z [m/s^2])
params.acc_b.push_back(std::array<double, 4> { {0.0, 0.0, 0.0, -9.81} });

// Position of origin of body frame in tank frame
params.pos_ref[0] = 0.0;
params.pos_ref[1] = 0.0;
params.pos_ref[2] = -0.508/2.0; // i.e. at tank bottom

// Add probes for analysis purposes
params.probes.push_back(SPH::PROBE(std::array<double, 3> { {-0.450, 0.0, 0.093 - 0.508/2.0} },
                                   "pressureSmoothed", "press_smoothed__Pa")); // Probes "array"
// Note that the location of the pressure sensor is in a height of 93mm measured from the
// tank bottom. The probe position needs to be given in the tank frame!
```

Figure 3.2: Propellant sloshing simulator: Excerpt from a configuration file comprising a subset of parameters needed to setup a propellant sloshing simulation.

The propellant sloshing simulator is written in C++, since it allows to use convenient high-level abstractions through its Object-oriented programming (OOP) approach, while at the same time offering machine-oriented features like low-level memory manipulation important for efficient High-performance computing (HPC) tasks.

Regarding the data structure, in the namespace *SPH* there are four classes defined containing variables or arrays of primitive data types, namely for particle states used for storing information like the particle's density or position, *SPH::PARTICLE*, system states used for storing information like the center of mass of the propellant, *SPH::SYSTEM*, probes used for measuring a desired information like pressure at arbitrarily defined locations in the problem domain, *SPH::PROBE*, and simulation parameters comprising all the configuration variables needed to define a simulation, *SPH::PARAMS*. Arrays of custom-build classes like *SPH::PARTICLE* as well as multidimensional arrays of primitive data types are managed via dynamic memory allocation using the new & delete operators for performance reasons. Additionally, since C++ uses row-major order, multi-dimensional arrays in the software are designed in a way that the first index changes slowest and the last index fastest, thus ensuring the most efficient data access.

Closing requirement [R5], a parallelization of the computations needed for approximating the propellant sloshing dynamics inside a tank is achieved through a shared-memory multiprocessing approach by using the OpenMP Application programming interface (API) [46]. Here, the run-time of a simulation is reduced by using multiple concurrently-running threads for calculating independent tasks.

The previously described setup allows to perform simulations using higher resolutions, but is especially well suited for simulations using lower resolutions due to the reduced parallelization overhead through usage of a shared-memory approach as well as a simple and efficient software implementation, thus satisfying requirement [R6]. Additionally, requirement [R4] is fulfilled trivially since C++ runs on the Linux operating system.

The output of the propellant sloshing simulator consists of the simulation results, i.e. a binary file containing the particle data as well as a Comma-separated value (CSV) text file containing the system state variables. The binary file contains the state of all simulation particles like e.g. their position, velocity, acceleration or density. The CSV file contains system state variables like the center of mass of the propellant or the integrated forces and torques exerted by the propellant on the tank wall.

The simulation results coming from the propellant sloshing simulator are fed into a newly developed post-processing tool written in Matlab [61]. The post-processing tool is able to display particle states data as well as system states data. The particle states are displayed in 3D as spheres with different colors indicating the value of the selected variable, as can be seen in figure 3.3 showing an example. The tank-propellant-gas system can be rotated in 3D as well as zoomed into. It is possible to hide dedicated types of particles (e.g. wall particles) and certain cuts and slices through a desired material phase can be shown as well. Convenient exporting functionality is available including the automatic creation of a video from the selected particle states view.

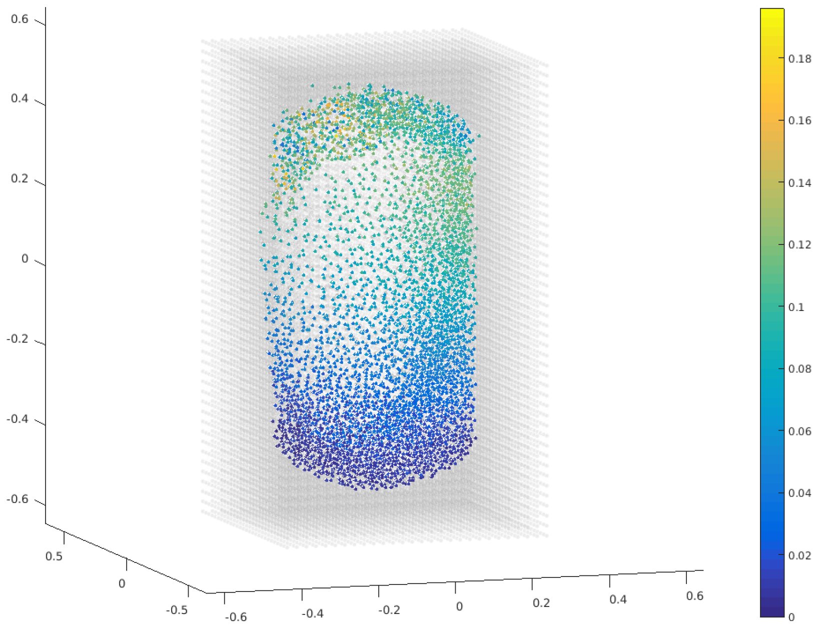


Figure 3.3: Propellant sloshing simulator: Post-processing tool: Particle states data example with velocity plotted

The post-processing tool is able to display one or multiple system states like the forces exerted by the propellant on the tank wall in one view, either one figure for each variable or all variables in one figure as desired. Please find in figure 3.4 an example of such a view displaying multiple variables at once. Limits for the x-/y-axis can be set for all figures in the view simultaneously, which is a convenient way of analyzing data sets with multiple variables involved. Further more, a suite of analysis options exist that can be applied to variables across multiple data sets at once, comprising a simple moving average filter, a Butterworth low-pass filter of desired degree, a peak analysis used for determining mean frequency, its standard deviation and the damping ratio of sloshing states as well as a power spectral density plot. Finally, an easy-to-use export functionality is available that allows to create figures of the desired data and also to export a data set into a Matlab file, which can be later then used for further analysis using Matlab directly.

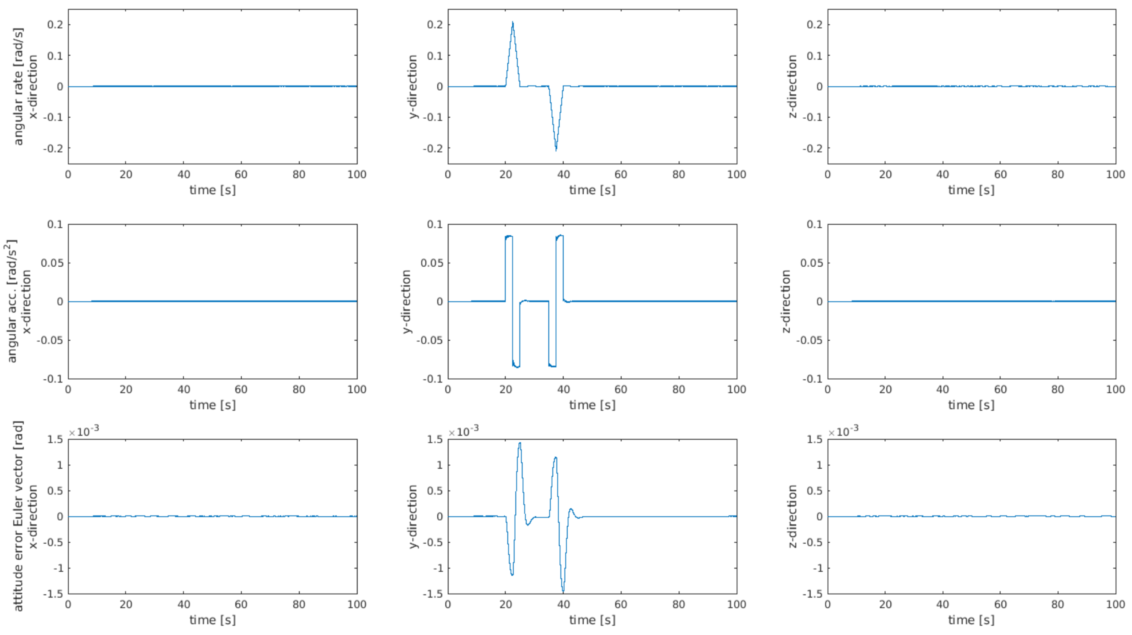


Figure 3.4: Propellant sloshing simulator: Post-processing tool: System states data example

In order to satisfy requirement [R7], Matlab has been chosen as environment to perform the post-processing inside, since it is an already known tool for most AOCS engineers and additionally provides convenient ways of built-in data visualization techniques. Also the post-processing tool fulfills requirement [R4] trivially since Matlab runs on the Linux operating system.

In the following, the workflow of a simulation using the proposed computational propellant sloshing model and its software implementation as described above is explained. As can be seen in figure 3.5, the end-to-end process of a sloshing simulation starts with the pre-processing, which in the implementation at hand only consists of the creation or modification of a configuration file as drafted in figure 3.2. As stated above, the geometry creation for the material phases involved is done automatically based on a configuration parameter.

The computer simulation starts and the previously defined configuration file is loaded into memory. Then the simulation variables are initialized and computed, as e.g. the speed of sound c_s or the minimum required time step size Δt, which are automatically calculated based on given configuration parameters. Also the geometry creation is performed in this step.

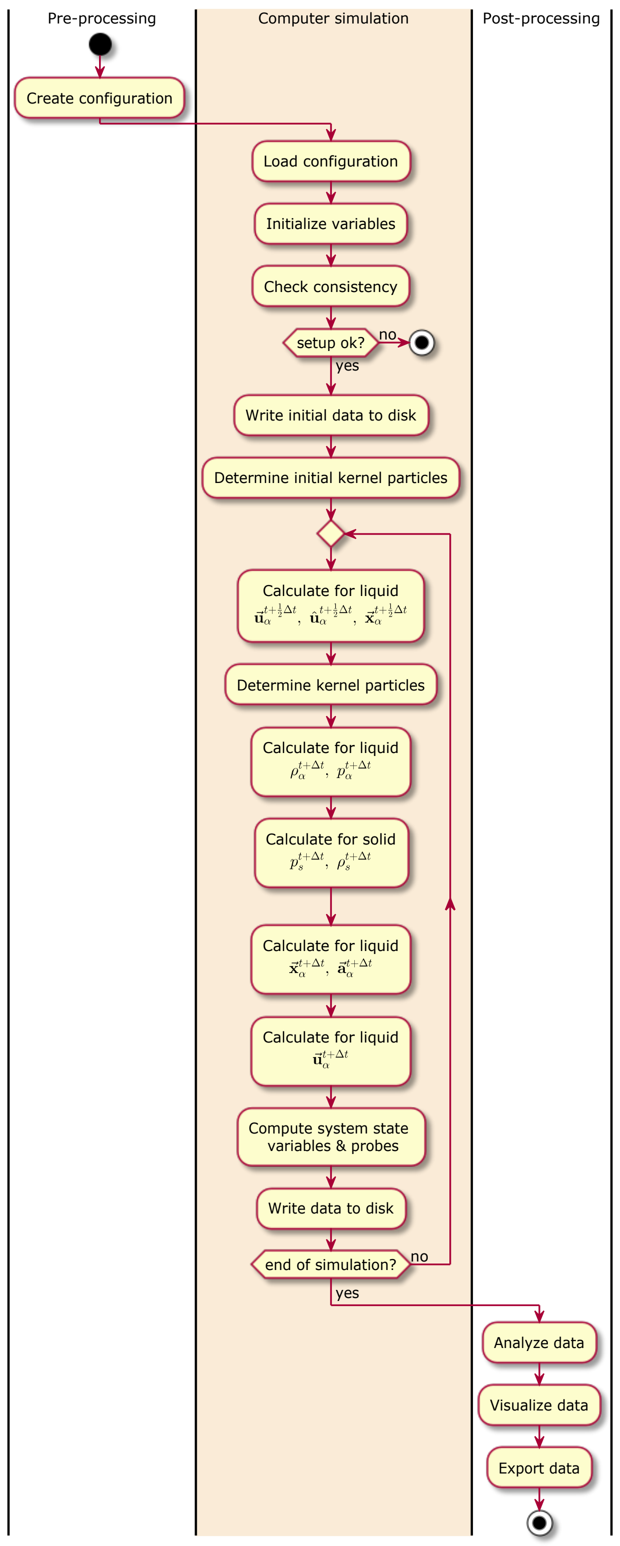

Pre-processing
Computer simulation
Post-processing
Create configuration
Load configuration
Initialize variables
Check consistency
setup ok?
no
yes
Write initial data to disk
Determine initial kernel particles
Calculate for liquid $\vec{\mathbf{u}}_\alpha^{t+\frac{1}{2}\Delta t}$, $\hat{\mathbf{u}}_\alpha^{t+\frac{1}{2}\Delta t}$, $\vec{\mathbf{x}}_\alpha^{t+\frac{1}{2}\Delta t}$
Determine kernel particles
Calculate for liquid $\rho_\alpha^{t+\Delta t}$, $p_\alpha^{t+\Delta t}$
Calculate for solid $p_s^{t+\Delta t}$, $\rho_s^{t+\Delta t}$
Calculate for liquid $\vec{\mathbf{x}}_\alpha^{t+\Delta t}$, $\vec{\mathbf{a}}_\alpha^{t+\Delta t}$
Calculate for liquid $\vec{\mathbf{u}}_\alpha^{t+\Delta t}$
Compute system state variables & probes
Write data to disk
end of simulation?
no
yes
Analyze data
Visualize data
Export data

Figure 3.5: Propellant sloshing simulator: Workflow of a simulation using the proposed computational propellant sloshing model.

After that, a consistency check is done, if the provided input parameters and the derived simulation parameters are meaningful values, as e.g. a check if the provided fluid height inside the tank is a multiple of the provided particle diameter or if the discretized volume equals the geometrical volume of the fluid. If any of the check fails, an error message is displayed and the simulation is stopped. If the consistency checks are successful, the initial data set comprising the initial particle states and system states is written to disk.

Then the initial kernel particles are determined, i.e. a mapping of each particle i to his neighbor particles j within the kernel support. This is done by using a cell index approach as described in [5], where first the problem domain is divided into a regular 3D grid of cells with a cell width equal to the kernel support radius. Then each fluid particle is assigned a cell and for finding the neighbor particles j of a fluid particle i, the particles inside the 27 cells surrounding the fluid particle i are tested for being within the kernel support.

Following this simulation setup, subsequently the time loop starts with calculating for all liquid particles the velocities, transport velocities as well as positions at the midpoint time step $t + \frac{1}{2}\Delta t$ according to equations 2.6.1 to 2.6.3. Due to the new positions of the liquid particles its kernel particles are determined again, in the same way than done before. Then the density as well as pressure of the liquid particles are calculated at time $t + \Delta t$ by use of equations 2.6.4 and 2.3.7. Then for the solid wall particles the pressure is calculated according to equation 2.7.3, yielding the solid wall particles density by use of equation 2.7.4. The positions at time step $t + \Delta t$ are now calculated using equation 2.6.5 and the new accelerations at time step $t + \Delta t$ according to equation 2.5.1. Finally the liquid velocities at time $t + \Delta t$ are computed using equation 2.6.6.

Afterwards, the system state variables, e.g. the forces exerted by the propellant on the tank wall, as well as the probe values, e.g. a pressure sensor as defined in the configuration file shown in figure 3.2 and used in section 9.1, are calculated and subsequently written to disk. Then it is checked if the end of the simulation is reached. If not, a new time loop starts by calculating the liquid's midpoint velocities, transport velocities and positions as explained above. If the end of the simulation has come, the post-processing starts and the resulting data is analyzed, visualized and exported by use of the post-processing tool as described above.

This completes the workflow of a simulation using the proposed computational propellant sloshing model and its software implementation. Since the tasks mentioned above are reoccurring, it is important to have the propellant sloshing simulation end-to-end process fully automated so that simulations including the analysis and export of the resulting data can be performed easily in the frame of AOCS verification campaigns, where a large number of simulations is performed.

Please note that the remaining open requirement [R3] about the coupling of the propellant sloshing simulator with a rigid-body simulator is discussed and closed in chapter 8.

Chapter 4

Validation of propellant sloshing model in high-gravity conditions

4.1 Introduction

In the following, the computational propellant sloshing model presented in chapter 2 and its software implementation shown in chapter 3 are validated for high-gravity environments. Following [23, 50], simulations of a hydrostatic test and a dynamic sloshing test of a partially filled 3D rectangular tank under Earth-gravity are performed and the results are compared to analytical, other numerical as well as experimental results coming from literature. Then the same test setup is used to analyze the impact of different spatial resolutions on global system properties like the integrated forces acting on the tank structure. Please note that in [23], simulations of the same test setup have been performed using a different computational propellant sloshing model, i.e. a less advanced predecessor of the computational model presented in chapter 2.

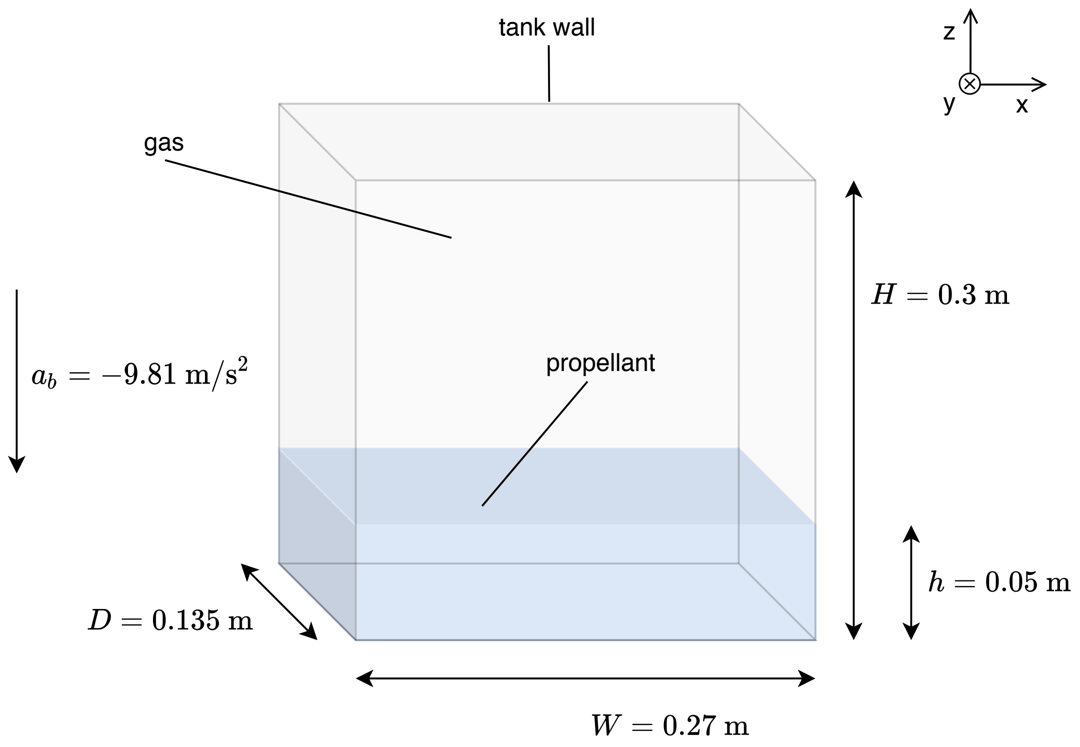


Figure 4.1: High-gravity validation: Experimental test setup

The test setup for the simulations presented in this chapter consists of a partially filled 3D rectangular tank of width $W = 0.27$ m, depth $D = 0.135$ m and height $H = 0.3$ m. The tank is filled with water of density $\rho = 1000$ kg/m^3 up to a height of $h = 0.05$ m and Earth-gravity, i.e. a vertical acceleration of $a_b = -9.81$ m/s^2, is acting on the system. A sketch of the test setup can be found in figure 4.1.

The simulations are performed using the computational propellant sloshing model presented in chapter 2 and its software implementation presented in chapter 3. The tank structure including fluid is discretized with particles of diameter 0.005 m, yielding 14,580 fluid particles. As can be seen in figure 4.2, the simulation particles are geometrically distributed using a Cartesian grid around the center of the tank with dimensions of the tank plus 3 layers of wall particles around in order to provide full kernel support for fluid particles close to the tank wall. The time step size is set to 0.13 ms and the maximum density variation of the fluid to 1 %. A Wendland C^2 kernel as described in section 2.2 is used with a compact support chosen such that the kernel comprises approximately 60 particles.

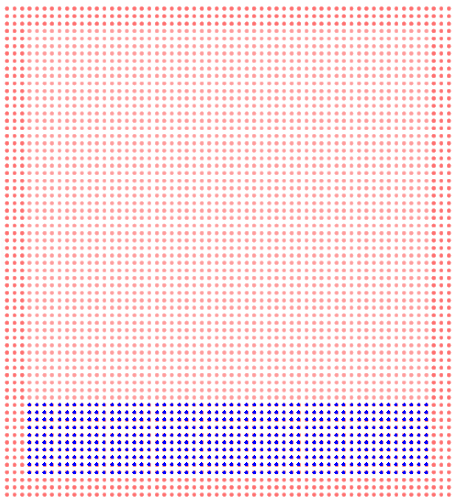


Figure 4.2: High-gravity validation: Numerical setup: z/x axis view of a cut through tank and fluid at y = 0 m

4.2 Hydrostatic test

To validate the hydrostatic behavior of the proposed computational propellant sloshing model, the pressure field as well as the forces acting on the tank structure of the test setup as described before under Earth-gravity shall be simulated and the results compared to analytical results.

A simulation is performed with the gravitational acceleration initially set to 0 $\mathrm{m/s^2}$ and then gradually ramped up to a value of -9.81 $\mathrm{m/s^2}$ within 0.5 s. Please note that no additional damping of the system in the form of e.g. artificial viscosity or increased dynamic viscosity (cf. [41]) is used in this simulation for the initialization process, demonstrating the numerical stability of the computational approach presented in this work. Figure 4.3 shows the resulting pressure field at time $t = 1$ s, where the expected hydrostatic pressure distribution can be clearly seen qualitatively.

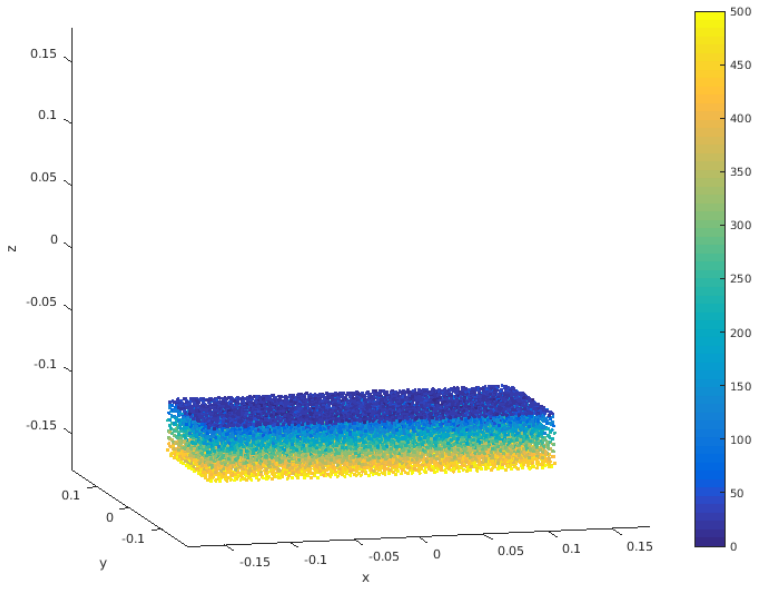


Figure 4.3: High-gravity validation: Hydrostatic test: Pressure distribution [Pa] at time $t = 1$ s

It can be observed that the fluid particles are not anymore arranged according to their initial geometric distribution using a Cartesian grid as shown in figure 4.2. This is due to the fact that the fluid density in the proposed computational propellant sloshing model is determined by use of a combination of the discretized continuity equation and the commonly used density summation approach as described in chapter 2. This density calculation, already including the treatment of the fluid free surface, together with the chosen kernel as well as the number of particles inside the kernel support radius determine the final converged particle distribution.

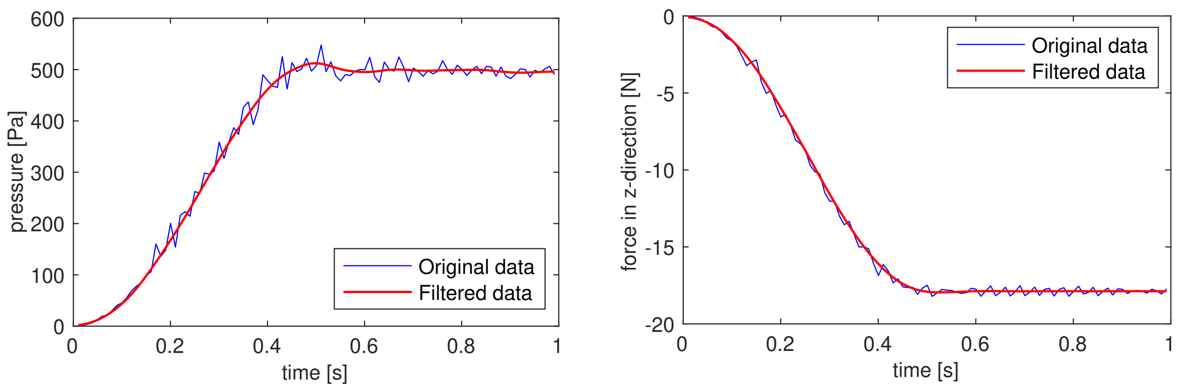
Assuming the fluid free surface as reference with zero pressure, the hydrostatic pressure of the fluid at the bottom of the tank can be calculated by

$$p_{\text{ana}} = \rho \cdot a_b \cdot h = 490.5 \text{ Pa} \tag{4.2.1}$$

with a_b being the vertical acceleration due to Earth-gravity and h the height of the fluid inside the tank.

Figure 4.4(a) shows the simulated pressure at the center of the bottom of the tank as well as a low-pass filtered version of it using a 3rd order Butterworth filter with a cutoff frequency of 5 Hz.

The ramp-up of pressure due to the increasing vertical acceleration can be observed very well, having a converged solution at time $t = 1$ s of $p_{\text{sim}} = 496.0$ Pa. This means that the simulated result is slightly over-estimating the theoretical pressure value by 1.1 %. Additionally it can be seen that there are high-frequency oscillations present in the pressure signal. This is due to the chosen weakly-compressible numerical approach and the resulting sound wave oscillations, what is explained in detail in section 9.2.3. For now it can be concluded that a low-pass filter as described above removes the noise effectively and provides the expected pressure signal.



(a) Pressure (b) Force in z-direction

Figure 4.4: High-gravity validation: Hydrostatic test: (a) pressure at the center of the bottom of the tank, (b) forces in z-direction exerted by the fluid on the tank wall. Blue lines represent the original data, red lines the low-pass filtered version of this data using a 3rd order Butterworth filter with a cutoff frequency of 5 Hz.

The forces exerted by the fluid on the tank structure at the bottom of the tank can be theoretically calculated as

$$F = p_{\text{ana}} \cdot (W \cdot D) = 17.88 \text{ N} \tag{4.2.2}$$

with W being the width and D the depth of the tank.

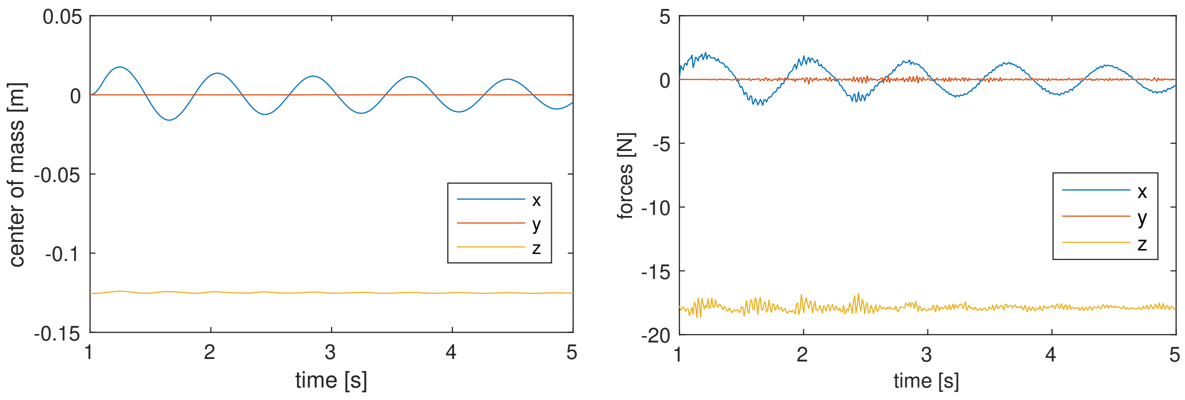
The simulation results for these forces acting on the bottom of the tank, i.e. the forces in z-direction, are shown in figure 4.4(b). It is evident that the increasing simulated forces in z-direction qualitatively reproduce the increasing vertical acceleration. At $t = 0.5$ s the maximum vertical acceleration, i.e. Earth-gravity, is reached and the simulated force value converges exactly to the theoretically determined value of 17.88 N. Also in this force signal the numerical noise due to the weakly-compressible numerical approach can be identified. In the same way as for the pressure, a 3rd order Butterworth low-pass filter with a cutoff frequency of 5 Hz eliminates the noise entirely and provides the expected signal.

The previously mentioned numerical noise is always present in the simulation results, it is basically an inherent part of the numerical method itself. Thus also variables like the forces exerted by the fluid on the tank structure are affected by this noise, which needs to be taken care of if a propellant sloshing model is interfaced to an external AOCS simulator for including propellant sloshing effects in the spacecraft dynamics and thus in the attitude & orbit controller (as done later in chapter 9). An effective method of eliminating the numerical noise is to apply a low-pass filter as for example done before in figure 4.4 using a 3rd order Butterworth filter. But usually a dedicated filtering is not needed, since the frequency of the artificial oscillations is way higher than the controller bandwidth of the AOCS, hence the AOCS is not affected by the numerical noise. This topic will be discussed in detail in section 9.2.3, but it can already be mentioned that the noise frequency can be controlled by the numerical compressibility of the fluid and is usually set an order of magnitude higher than the controller bandwidth.

4.3 Dynamic sloshing test

The proposed computational propellant sloshing model is now applied to a dynamic sloshing example based on the test and numerical setup as described in section 4.1. Starting from the converged solution of the hydrostatic test from the previous section 4.2 at time $t = 1$ s, an additional body acceleration of 2 m/s^2 is applied for 0.1 s in x-direction, which makes the fluid start sloshing laterally.

Figure 4.5(a) shows the resulting movement of the fluid's center of mass with the characteristic oscillation in x-direction due to the lateral excitation. Additionally, a small center of mass shift in z-direction can be observed. This vertical sloshing motion is an order of magnitude smaller than the x-direction lateral sloshing and thus not of relevance for the current analysis.



(a) Center of mass (b) Forces on the tank structure

Figure 4.5: High-gravity validation: Hydrodynamic test: (a) Center of mass of the fluid, (b) Forces exerted by the fluid on the tank structure.

For classical mechanical analog propellant sloshing models (like pendulums), the center of mass shift is an essential parameter needed for the model setup. The important physical parameter needed for describing the rigid-body / fluid dynamics interaction properly however is the force exerted by propellant on the tank structure. Hence the following sloshing analysis, i.e. the determination of sloshing frequency and damping, will be done based on this force data, which is plotted in figure 4.5(b). Here, the previously described oscillations in x- and z-direction are also visible, but additionally some high-frequency parts in the force can be observed that stem from the weakly-compressible numerical approach as mentioned already in section 4.2 and as will be discussed in detail in section 9.2.3.

An analysis of the force in x-direction exerted by the fluid on the tank structure can be seen in figure 4.6. Here the original data is plotted along with a filtered version of it using a 5th order Butterworth filter with a cutoff frequency of 2 Hz. Using the local maxima of the filtered data, a linear damped oscillator model is fitted, which yields the first antisymmetric sloshing mode frequency as $f_{11} = 1.24 \pm 0.02$ Hz, a damping of $\gamma = 2.1$ % and a maximum force exerted by the fluid on the tank structure of $F_0 = 1.80$ N. The resulting fitted curve using the linear damped oscillator model can be found in figure 4.6.

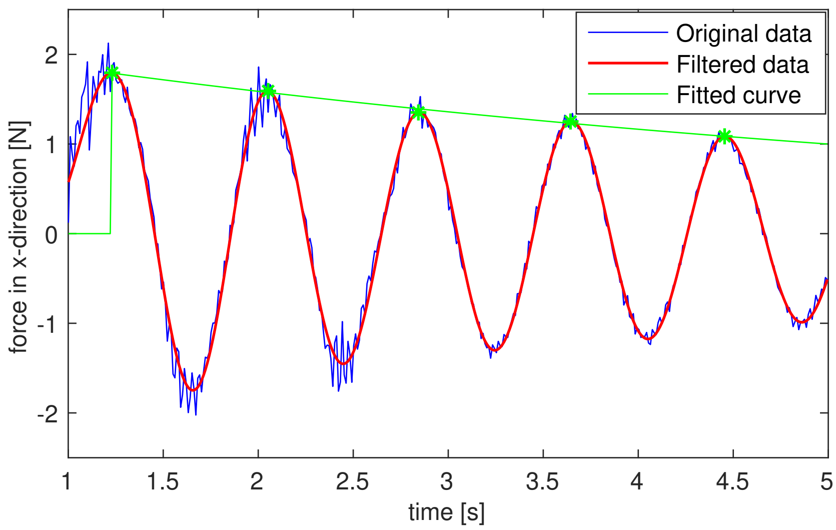


Figure 4.6: High-gravity validation: Hydrodynamic test: Analysis of forces in x-direction exerted by the fluid on the tank structure. Blue line represents the original data, red line the low-pass filtered version of this data using a 5th order Butterworth filter with a cutoff frequency of 2 Hz, green line the fitted curve as output from the estimation of sloshing frequency and damping using a linear damped oscillator.

The first antisymmetric sloshing mode frequency obtained from the fitted linear damped oscillator model, $f_{11} = 1.24 \pm 0.02$ Hz, is compared to the values obtained from theory [25, 13], experiments [30], simulations with other numerical codes [30, 50] and also to the values obtained from simulations using a less advanced predecessor [23] of the computational model presented in this work. The results can be seen in table 4.1, where it becomes obvious that the sloshing frequency determined by the proposed computational sloshing model fits nicely to the other values. The values obtained by analytical formulas and other numerical codes fit within 0.8 % very well to the result determined by the new sloshing model. The sloshing frequency obtained from experiment [30] is slightly higher than all other analytical and numerical values and deviates from the frequency of the new sloshing model by 1.6 %, but still lies within the error bounds.

Table 4.1: High-gravity validation: Hydrodynamic test: Comparison of results for the 1st antisymmetric sloshing mode frequency, f_{11}

	f_{11} [Hz]
Analytical: Housner [25]	1.24
Analytical: Dodge/Abramsson [13]	1.23
Experimental: Jaiswal [30]	1.26
Numerical: Ansys [30]	1.23
Numerical: Elmer [50]	1.23
Numerical: Legacy propellant sloshing model [23]	1.24
Numerical: Propellant sloshing model at hand	1.24

To analyze the frequency spectrum of the force in x-direction exerted by the fluid on the tank structure, the single-sided power spectral density of the force signal is plotted in figure 4.7 in blue. Please note that the single-sided power spectral densities in this work are calculated using a digital Fourier transform based on a Fast Fourier Transform algorithm as described in [49]. The first major peak of the plot, i.e. the first antisymmetric sloshing mode at 1.24 Hz can be clearly identified. Zooming into values of the power spectral density between 0 and $3 \cdot 10^{-3}$ N^2/Hz, also the second antisymmetric sloshing frequency at 2.85 Hz becomes visible as well as several other modes up to 6 Hz. Between 20 and 40 Hz, again multiple peaks can be seen that stem from the sound wave oscillations resulting from the weakly-compressible numerical approach (see section 9.2.3 for more details). Please note that the previously discussed higher sloshing modes as well as the sound wave oscillations have a power spectral density value that is 3 orders of magnitudes smaller than the first antisymmetric sloshing mode. Anyhow, by applying the Butterworth low-pass filter as described before, these frequencies are easily eliminated, what can be seen in figure 4.7 in red.

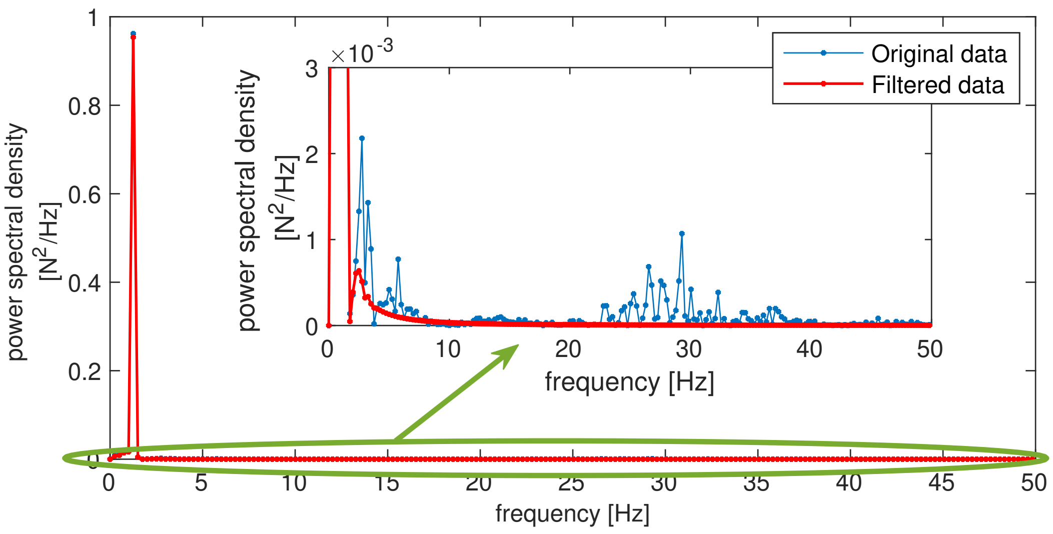


Figure 4.7: High-gravity validation: Hydrodynamic test: Power spectral density of the forces in x-direction exerted by the fluid on the tank structure. Blue line represents the power spectral density of the original data, red line the power spectral density of the low-pass filtered version of this data using a 5th order Butterworth filter with a cutoff frequency of 2 Hz.

The damping ratio obtained from the fitted linear damped oscillator model is calculated to be $\gamma = 2.1$ %. Even though this damping ratio is higher than an experimentally determined value of 0.53 %, which has been obtained by experimental tests with circular cylindrical tanks [13], it is still well below the damping ratio of 3.5 % obtained in previous work [23]. Please also note that in comparison to the legacy propellant sloshing code used in [23], the simulations performed in this work were done with a resolution of 14,580 fluid particles instead of 229,500. As will be discussed in the next section, the spatial resolution plays an essential role for the damping behavior, i.e. a higher resolution implies less numerical damping and vice versa. Hence the new propellant sloshing code is clearly superior to the legacy one from [23], inducing 40 % less damping with a resolution over 15 times rougher, which is due to the implemented transport-velocity formulation from section 2.3 in combination with the discretized continuity equation from section 2.4. This approach stabilizes the simulation to such an extent that no artificial viscosity is needed.

4.4 System properties under reduced spatial resolution

During AOCS verification campaigns a very large number of high-fidelity simulations are performed in order to predict the system behavior considering all relevant disturbances and their possible variations. When the necessity of including a computational propellant sloshing model in these AOCS verification campaign simulations is given, for example due to varying non-linearities in the propellant's dynamics during high-agility spacecraft maneuvers including high-velocity impacts, then it is important to use a computational model that runs within a reasonable amount of time. Despite highly optimized algorithms of course, the most important performance gain can be achieved by reducing the spatial resolution. As such, it is of interest how system properties like the center of mass or the force exerted by the fluid on the tank structure behave under reduced spatial resolution.

In the following, the dynamic sloshing test of section 4.3 is repeated twice, i.e. one simulation with higher resolution and one simulation with lower resolution than the one already done is performed. This yields three simulations of the dynamic sloshing test with particle diameters of 0.002 m, 0.005 m and 0.01 m, resulting in 229,500, 14,580 and 1,890 fluid particles, respectively.

Figure 4.8 shows the resulting center of mass for the three different spatial resolutions. It can be observed that qualitatively the three simulations show very similar results. Looking at the x-component of the center of mass, the frequency of the three signals are really close, even though a small tendency of lower resolutions to lower frequencies can be seen. The damping behavior of the three simulations on the other hand differs significantly, showing a stronger damping for lower resolutions.

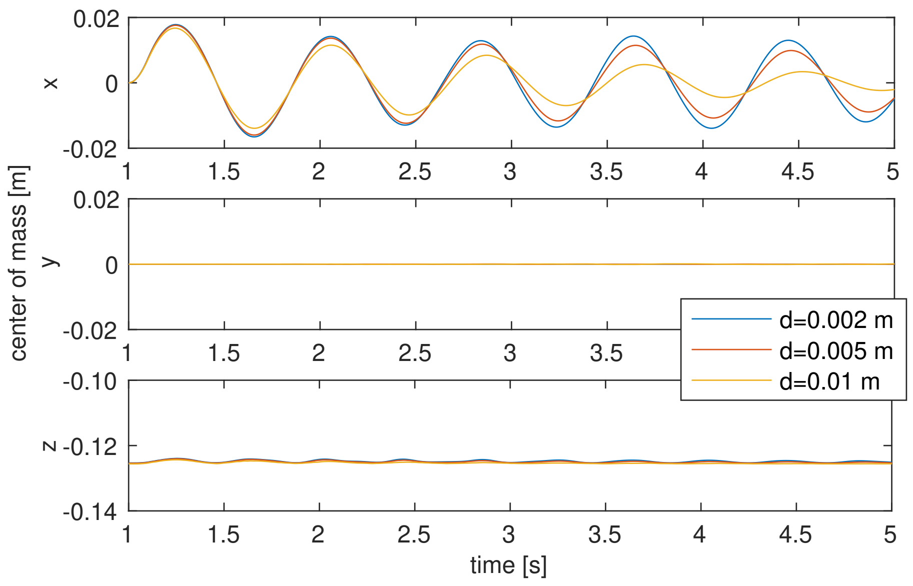


Figure 4.8: High-gravity validation: Resolution study: Center of mass of the fluid for three different resolutions. Blue line: particle diameter of $d = 0.002$ m, red line: particle diameter of $d = 0.005$ m, yellow line: particle diameter of $d = 0.01$ m.

As mentioned before in section 4.3, the important physical parameter needed for describing the rigid-body / fluid dynamics interaction properly is the force exerted by the propellant on the tank structure. Thus again this force data is used for the following sloshing analysis, i.e. the determination of sloshing frequency and damping. The force exerted by the propellant on the tank structure for the three different resolutions can be found in figure 4.9. The forces for the different spatial resolutions show qualitatively similar results like for the center of mass before. Evidently the noise on the signal due to the sound wave oscillations is present as well. A fundamental difference is given again due to the differing damping ratios for the three resolutions, having the highest damping for the lowest resolution.

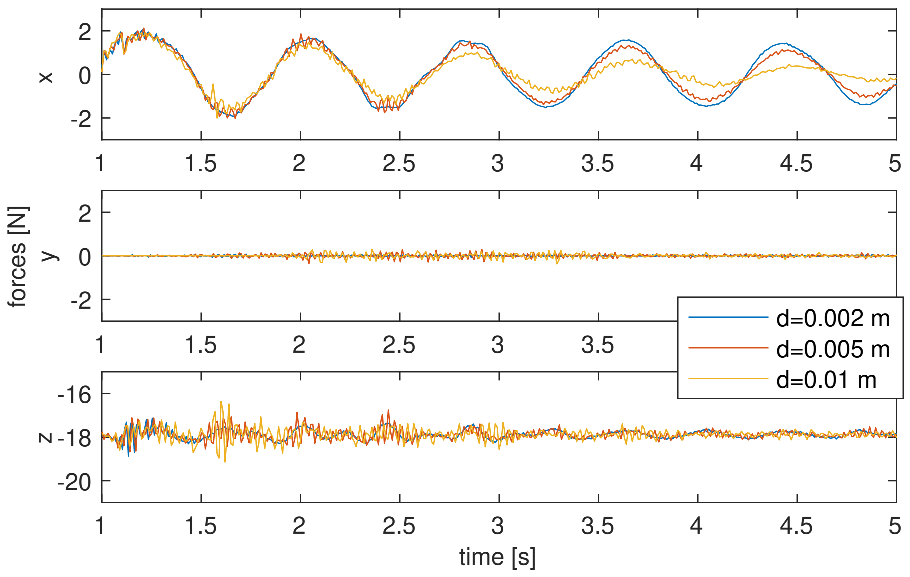


Figure 4.9: High-gravity validation: Resolution study: Forces exerted by the fluid on the tank structure for three different resolutions. Blue line: particle diameter of $d = 0.002$ m, red line: particle diameter of $d = 0.005$ m, yellow line: particle diameter of $d = 0.01$ m.

To analyze the sloshing in detail on a quantitative level, figure 4.10 shows the force in x-direction exerted by the fluid on the tank structure for the three different spatial resolutions. The original data is plotted here along with a filtered version of it using a 5th order Butterworth filter with a cutoff frequency of 2 Hz. Additionally, a linear damped oscillator model is fitted to the local maxima of the low-pass filtered data and the resulting curve is shown in figure 4.10. The first antisymmetric sloshing mode frequencies for the three simulations are calculated as 1.24 ± 0.02 Hz, 1.24 ± 0.02 Hz and 1.22 ± 0.02 Hz for particle diameters of 0.002 m, 0.005 m and 0.01 m, respectively. Even though the lowest resolution shows a slightly lower frequency than the other two simulations, the relative difference in sloshing frequency between the lowest and highest resolution is only 1.6 %, thus the spatial resolution of the proposed computational propellant sloshing model can be considered as having no big influence on the sloshing frequency.

The maximum forces in x-direction exerted by the fluid on the tank structure are given by 1.81 N, 1.80 N and 1.75 N for particle diameters of 0.002 m, 0.005 m and 0.01 m, respectively. These force values lie closely together and have a maximum relative error of 3.3 %, which is totally acceptable for being used within AOCS verification campaign simulations. The damping ratios obtained from the fitted linear damped oscillator model are calculated to be 0.9 %, 2.1 % and 6.8 % for particle diameters of 0.002 m, 0.005 m and 0.01 m, respectively. Here again the in-

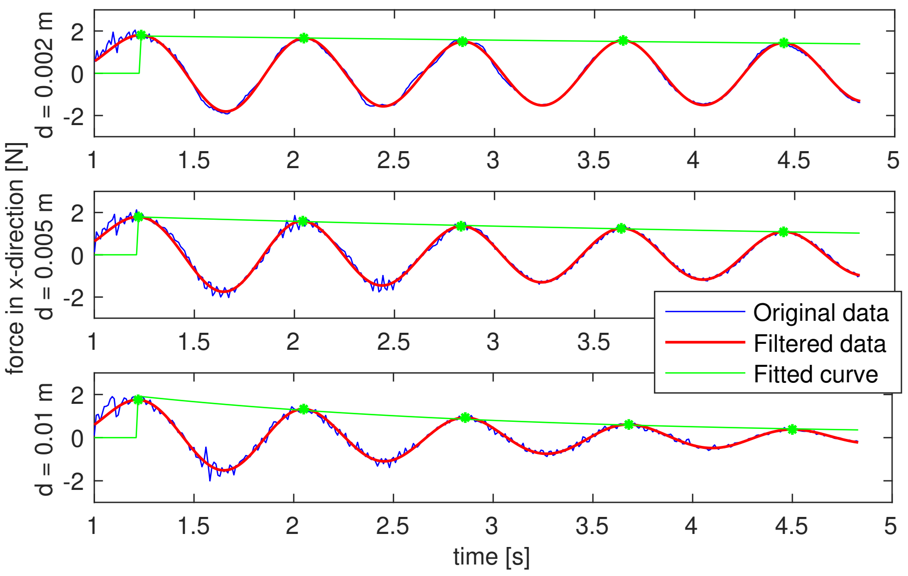


Figure 4.10: High-gravity validation: Resolution study: Analysis of forces in x-direction exerted by the fluid on the tank structure for three different resolutions. Top figure: particle diameter of $d = 0.002$ m, middle figure: particle diameter of $d = 0.005$ m, bottom figure: particle diameter of $d = 0.01$ m. Blue line represents the original data, red line the low-pass filtered version of this data using a 5th order Butterworth filter with a cutoff frequency of 2 Hz, green line the fitted curve as output from the estimation of sloshing frequency and damping using a linear damped oscillator.

creasing damping ratio for decreasing spatial resolution is clearly visible. On the other hand, the simulation with the highest resolution, i.e. a particle diameter of 0.002 m, gives a damping ratio of only 0.9 %, which is pretty close to the experimentally determined value of 0.53 % and highly superior to the legacy propellant sloshing code used in [23] with a damping ratio of 3.5 % for the same resolution.

As already stated in [23], the runtime of a simulation based on a specific implementation of the proposed computational propellant sloshing model depends on multiple factors like the used programming language and algorithms, the hardware characteristics including CPUs and GPUs as well as parallelization techniques and the optimization level of the code. Since in this section the system properties for different spatial resolutions are analyzed for one dedicated code implementation and hardware system, relative differences in the simulation runtimes for the respective spatial resolutions are given. Taking the simulation with highest spatial resolution, i.e. a particle diameter of 0.002 m, as reference, the simulation with a particle diameter of 0.005 m is performed 40 times faster and the simulation with 0.01 m particle diameter is even 500 times faster. Given the fact that the sloshing frequencies calculated from the forces exerted by the fluid on the tank structure differ by only 1.6 % between the highest and lowest spatial resolution with 229,500 and 1,890 fluid particles, respectively, for a 3D simulation, it can be concluded that the proposed computational propellant sloshing model is well-suited to be used in AOCS verification campaign simulations for obtaining reasonable results in a short amount of time.

Chapter 5

Validation of propellant sloshing model in zero-gravity conditions

5.1 Introduction

In the following, the computational propellant sloshing model presented in chapter 2 and its software implementation presented in chapter 3 are validated for zero-gravity environments. As discussed in section 2.1, in the absence of gravity and inertia forces, the major driver for the propellant's sloshing dynamics are the interfacial forces, which fundamentally consist of short-range cohesion forces (attraction of like molecules) and adhesion forces (attraction of dissimilar molecules) between the involved gas, propellant and tank molecules. The most important interfacial effects relevant for isothermal propellant sloshing in spacecraft are surface tension, i.e. the stronger cohesion forces inside the propellant than adhesion forces between propellant and gas causing the liquid free-surface to go into its minimum state, as well as the wetting effects occurring in a gas-propellant-tank system like the contact angle at the triple-points of the gas-liquid-solid interface, which is driven by the interplay of the involved cohesion and adhesion forces.

It is important to note that for describing all the interfacial effects seen in this work, no dedicated modeling for specific effects like static and dynamic contact angles with contact angle hysteresis has been done, as it is common for computational fluid dynamics models based on the finite element method (as e.g. in [51]), the finite volume method (as e.g. in [66]) or the boundary element method (as e.g. in [9]). The developed propellant sloshing model at hand is solely based on generic molecular-like cohesion and adhesion forces between like and dissimilar particles (see section 2.5), which are able to describe effects like surface tension, static and dynamic contact angles, contact angle hysteresis etc. without any specific configuration, adjustments or model changes.

Since in real-world applications the gas-propellant-tank systems are characterized by experimentally determined parameters like static contact angle and dynamic surface tension, during the numerical setup these values are used to derive the appropriate molecular-like cohesion and adhesion interaction strengths for the involved gas, propellant and tank particles. This means that at the beginning of a simulation the cohesion and adhesion interaction strengths present in the system are calculated based on these experimentally determined system parameters available to the user and then during the simulation the interplay of the involved cohesion and adhesion forces will produce the desired interfacial effects.

To validate the proposed computational propellant sloshing model and its software implementation for zero-gravity conditions, simulations will be performed showing the proper description of surface tension for liquid droplets as well as the proper description of wetting effects, showing wetting and non-wetting droplets on a horizontal solid surface. For all these simulations, experimentally available parameters will be used to automatically determine the involved cohesion and adhesion force strengths of the system.

Please note that in [24], simulations of the same test setup have been performed using a different computational propellant sloshing model, i.e. a less advanced predecessor of the computational model presented in chapter 2.

5.2 Surface tension

The successful modeling of surface tension effects is demonstrated by simulating liquid spherical droplets of different size in a zero-gravity environment and comparing the resulting pressure difference across the fluid-fluid interface due to surface tension with the analytical solution given by the Young-Laplace equation.

The droplets have a diameter d of 2 mm, 4 mm, 6 mm and 8 mm, respectively, and consist of a liquid with density $\rho = 1000$ kg/m^3, a dynamic viscosity of $\mu = 10$ mPa·s and a surface tension of $\sigma_{\alpha\beta} = 70$ mN/m between the liquid phase α and a surrounding gas phase β.

Regarding the numerical setup, the droplets are discretized with a constant particle number of 17,256, resulting from a particle spacing of $\Delta x = d/32$. The simulation particles are geometrically distributed using a Cartesian grid around the center of the droplet, creating a fluid particle in case it is inside the radius $r_f = d/2$ of the droplet origin. Figure 5.1 shows for the droplet the geometrical particle distribution after initialization at time $t = 0$ s. The maximum density variation of the fluid is set to 0.1 % and the time step size to the maximum value allowed according to section 2.6, giving values between $\Delta t = 2$ μs for $d = 2$ mm and $\Delta t = 20$ μs for $d = 8$ mm. A Wendland C^2 kernel as described in section 2.2 is used with a compact support chosen such that the kernel comprises approximately 200 particles.

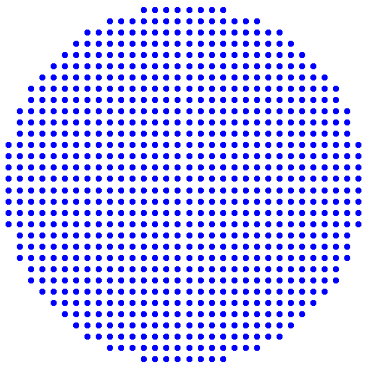


Figure 5.1: Low-gravity validation: Surface tension: Initial geometrical particle distribution, showing a slice of thickness $\Delta y = [-0.1, +0.1]$ mm (i.e. a layer of 4 particles in y-direction) through the fluid

The simulations are started without any prescribed pressure and using an easy to produce, but rather unphysical distribution of fluid particles at the free-surface with edges and a non-smooth shape as explained before and shown in figure 5.1. During a short settling time, the pressure inside the droplet due to surface tension is building up, the free-surface is converging to a smooth state accomplished through the surface tension minimizing the fluid surface and the droplet oscillations are being damped. Finally the droplet reaches an equilibrium state as can be seen in figure 5.2 for time $t = 0.2$ s.

In order to validate the droplet's pressure difference across the fluid free-surface coming from the simulation results Δp^{sim}, first its theoretical value Δp^{th} needs to be determined. As shown in section 2.1, the stress balance at the interface of the two fluid phases α and β reads [8]

$$\vec{\mathbf{n}}(p_\alpha - p_\beta) = \vec{\mathbf{n}}(\boldsymbol{\tau}_\alpha - \boldsymbol{\tau}_\beta) + \vec{\mathbf{n}}\sigma_{\alpha\beta}\kappa - \nabla\sigma_{\alpha\beta} \tag{5.2.1}$$

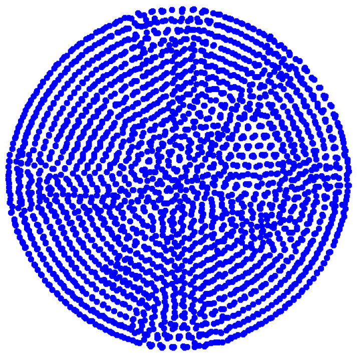


Figure 5.2: Low-gravity validation: Surface tension: Converged particle distribution at time $t = 0.2$ s, showing a slice of thickness $\Delta y = [-0.1, +0.1]$ mm (i.e. a layer of 4 particles in y-direction) through the fluid

with $\vec{\mathbf{n}}$ being the normal vector pointing away from the α phase, $\boldsymbol{\tau}_{\alpha/\beta} = \left[\mu_{\alpha/\beta}(\nabla\vec{\mathbf{u}}_{\alpha/\beta} + \nabla\vec{\mathbf{u}}_{\alpha/\beta}{}^T)\right]$ the deviatoric stress tensor and $\mu_{\alpha/\beta}$ the dynamic viscosity, $\sigma_{\alpha\beta}$ the surface tension between the two phases α and β, $\kappa = \nabla \cdot \vec{\mathbf{n}}$ the curvature of the interface and $\nabla\sigma_{\alpha\beta}$ the tangential stress due to the gradients in the surface tension.

In the simulation at hand, after the droplet converged to its final state, the fluid is at rest, leading to zero viscous stress $\boldsymbol{\tau}_\alpha = \boldsymbol{\tau}_\beta = 0$. Since all other contributions in eq. 5.2.1 act in normal direction, it follows that the tangential stress is also zero, $\nabla\sigma_{\alpha\beta} = 0$. Additionally, a spherical droplet of radius r is used, yielding a curvature of $\kappa = 2/r$ [8]. This yields the final equation for the pressure difference Δp^{th} across the fluid-fluid interface of a droplet at rest, formally known as Young-Laplace equation, as

$$\Delta p^{\text{th}} = p_\alpha - p_\beta = \frac{2}{r}\sigma_{\alpha\beta} \tag{5.2.2}$$

Regarding the simulation results, the total pressure p_α inside the droplet cannot be calculated solely based on the equation of state, eq. (2.3.7), but must also account for the particle-particle interactions of eq. (2.5.5) [58]. Hence the total pressure p_α inside a volume V of the fluid phase α is calculated using the virial theorem [57, 31] as

$$p_\alpha = \frac{1}{3V}\frac{1}{2}\sum_i\sum_j \vec{r}_{ij}(\vec{F}^p_{ij} + \vec{F}^m_{ij}) \tag{5.2.3}$$

with $\vec{r}_{ij} = \vec{r}_i - \vec{r}_j$ being the relative position regarding particles i and j, $\vec{F}^p_{ij}$ the pressure force exerted on particle i due to particle j (see eq. 2.5.2) and $\vec{F}^m_{ij}$ the molecular-like pairwise interaction force exerted on particle i due to particle j (see eq. 2.5.5). The index i in the double sum of eq. 5.2.3 is running over all particles in the volume V for which the total pressure shall be calculated and index j is running over all particles in the kernel of the respective particle i.

For the simulations at hand, the volume for which the total pressure in the fluid phase α is calculated is chosen to be a sphere around the droplet center as $V = \frac{4}{3}\pi r_v^3$ with virial radius $r_v = H$ and H being the compact support radius of the kernel used. Since the pressure of the gas phase is implicitly set to zero in the simulations, it is $p_\beta = 0$. This finally yields the computationally determined pressure difference Δp^{sim} across the fluid-fluid interface of the droplet at rest as

$$\Delta p^{\text{sim}} = p_\alpha - p_\beta = \frac{1}{8\pi H^3}\sum_i\sum_j \vec{r}_{ij}(\vec{F}^p_{ij} + \vec{F}^m_{ij}) \tag{5.2.4}$$

Since the index i in the double sum of eq. 5.2.4 is running over all particles in the volume V and index j is running over all particles in the kernel of the respective particle i, every particle is considered for the calculation of the total pressure in the liquid phase if it is within a radius of $2 \cdot H$ around the droplet center. Additionally, in order to avoid boundary effects at the fluid-fluid interface, it is required to ensure that particles within a distance of H around the fluid free-surface are not included in the calculation. This yields the necessary condition for a proper calculation of the total pressure in the liquid phase as $d/2 > 3 \cdot H$. Considering the droplet with smallest diameter simulated in this work, $d = 2$ mm, an upper limit for the compact support radius is thus given by

$$H < 0.\overline{3}\ \text{mm} \tag{5.2.5}$$

The numerical setup described above using a particle spacing of $\Delta x = d/32$ and a Wendland C^2 kernel comprising approximately 200 particles yields a compact support radius of $H = 0.23$ mm, which is below the upper limit derived in inequation 5.2.5.

Table 5.1 shows the pressure differences across the fluid-fluid interface of the droplets for different diameters derived from theory, Δp^{th}, coming from simulation, Δp^{sim}, and their relative error $\delta\Delta p = \Delta p^{\text{sim}}/\Delta p^{\text{th}} - 1$. It can be observed that the simulation results are reproducing theory pretty well with an average relative percentage error of 4.1 % and a maximum relative percentage error of less than 6 %.

Table 5.1: Low-gravity validation: Surface tension: Comparison of expected pressure difference across the fluid-fluid interface derived from theory, Δp^{th}, and simulated pressure difference Δp^{sim} for different droplet diameters d. The relative error is given by $\delta\Delta p = \Delta p^{\text{sim}}/\Delta p^{\text{th}} - 1$.

d [mm]	Δp^{th} [Pa]	Δp^{sim} [Pa]	$\delta\Delta p$ [%]
2	140.0	131.8	-5.9
4	70.0	73.2	4.6
6	46.7	47.9	2.6
8	35.0	36.1	3.1

Please note that the prescribed surface tension $\sigma_{\alpha\beta}$ between the two fluid phases α and β is not directly incorporated in the computational model, but only used to calculate the cohesion force strengths as explained in section 2.5. As such it would be possible to increase the accuracy of the computational approximation by fine-tuning the interaction strengths in eq. 2.5.6 manually. Considering the ease of just using available experimentally-determined parameters like the surface tension instead of performing manual fine-tuning as well as the fact that for AOCS verification purposes a surface tension approximation within 6 % is more than sufficient, no further calibration or fine-tuning is being done.

As analyzed in detail for the high-gravity environment in section 4.4, the dependency of the computational approximation on the spatial resolution is also assessed for the zero-gravity regime. Therefore the simulation of droplets with different diameters is repeated for a lower spatial resolution as well as for a higher spatial resolution. The higher spatial resolution uses a particle spacing of $\Delta x = d/40$, yielding 33,552 fluid particles, and the resulting compact support radius of $H = 0.18$ mm fulfills inequation 5.2.5. The lower spatial resolution uses a particle spacing of $\Delta x = d/25$, yielding only 8,217 fluid particles. Also here, the resulting compact support radius of $H = 0.29$ mm is below the derived upper limit given in inequation 5.2.5.

Table 5.2 shows the results for the pressure differences across the fluid-fluid interface of the droplets for different diameters d derived from theory, Δp^{th}, as well as calculated from the simulations, Δp^{sim}, in form of the relative error $\delta\Delta p = \Delta p^{\text{sim}}/\Delta p^{\text{th}} - 1$ for the three different spatial resolutions using subscript 'low' for 8,217 particles, 'mid' for 17,256 particles and 'high' for 33,552 particles.

It can be seen that the lower spatial resolution produces an average relative percentage error of 10.4 % and a maximum relative percentage error of less than 14 %, whereas the higher spatial resolution gives an average relative percentage error of 1.2 % and a maximum relative percentage error of 2.3 %. On the one hand this clearly shows the expected convergence to the theoretical value with increasing spatial resolution. On the other hand, it can be noted that already the low-resolution simulation is reproducing theory remarkably well, especially considering the fact that only 8,217 fluid particles are being used for a full 3D simulation incorporating interface phenomena.

Table 5.2: Low-gravity validation: Surface tension: Comparison of expected pressure difference across the fluid-fluid interface derived from theory, Δp^{th}, and simulated pressure difference Δp^{sim} for different droplet diameters d in form of the relative error $\delta\Delta p = \Delta p^{\text{sim}}/\Delta p^{\text{th}} - 1$ for three different spatial resolutions. The subscript denotes the used spatial resolution, i.e. 'low' for 8,217 particles, 'mid' for 17,256 particles and 'high' for 33,552 particles.

d [mm]	Δp^{th} [Pa]	$\delta\Delta p_{\text{low}}$ [%]	$\delta\Delta p_{\text{mid}}$ [%]	$\delta\Delta p_{\text{high}}$ [%]
2	140.0	-7.4	-5.9	-0.6
4	70.0	-9.3	4.6	-2.3
6	46.7	-13.9	2.6	-0.4
8	35.0	-11.1	3.1	-1.4

5.3 Wetting effects

In order to demonstrate that the proposed computational propellant sloshing model and its software implementation are able to properly reproduce wetting effects, in the following a liquid spherical droplet is placed on top of a horizontal solid surface in a zero-gravity environment and different static contact angles are prescribed, leading to different positions of the triple-points of the tank-propellant-gas interface and thus to different droplet shapes. The geometrically-determined contact angles resulting from the simulation are then compared to the prescribed contact angles.

The droplet has a diameter of $d = 2$ mm and consists of a liquid with a density of $\rho = 1000$ kg/m^3, a dynamic viscosity of $\mu = 10$ mPa·s and a surface tension of $\sigma_{\alpha\beta} = 70$ mN/m between the liquid phase α and a surrounding gas phase β. Furthermore, no gravitational acceleration is prescribed.

Regarding the numerical setup, the liquid droplet is discretized with a particle number of 8,217, resulting from a particle spacing of $\Delta x = d/25$. The liquid particles are geometrically distributed using a Cartesian grid around the center of the droplet, creating a liquid particle in case it is inside the radius $r_f = d/2$ of the droplet origin. The horizontal solid surface is also discretized with a particle spacing of $\Delta x = d/25$ and comprises 4 layers of wall particles in order to provide full kernel support for liquid particles close to the horizontal solid surface. Figure 5.3 shows for the droplet on the horizontal solid surface the geometrical particle distribution after initialization at time $t = 0$ s. The maximum density variation of the fluid is set to 0.1 % and the time step size to the maximum value allowed according to section 2.6, $\Delta t = 3$ μs. A Wendland C^2 kernel as described in section 2.2 is used with a compact support chosen such that the kernel comprises approximately 230 particles.

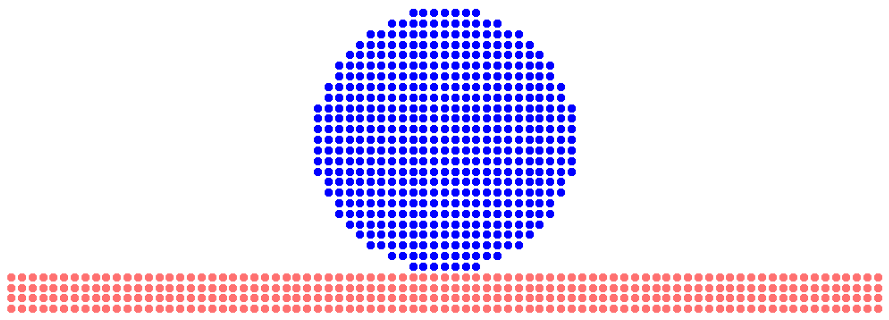
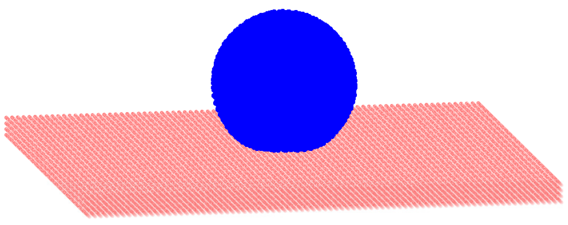
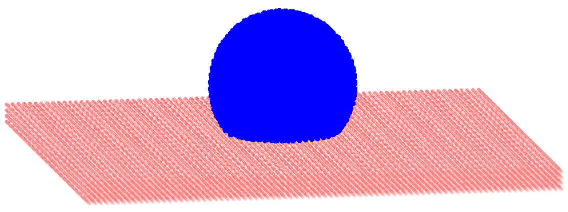


Figure 5.3: Low-gravity validation: Wetting effects: Initial geometrical particle distribution, showing a slice of thickness $\Delta y = [-0.1, +0.1]$ mm (i.e. a layer of 4 particles in y-direction) through the fluid

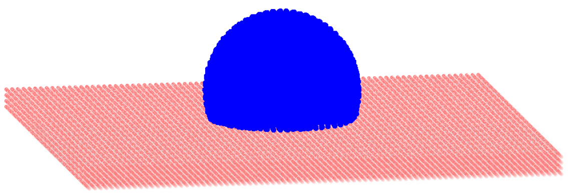
The simulation is started without any prescribed pressure and using an easy to produce, but rather unphysical distribution of fluid particles at the free-surface with edges and a non-smooth shape as shown in figure 5.3. Initially a prescribed static contact angle of $\theta_0 = 150°$ is used and during a short settling time, the interplay of cohesive and adhesive forces is causing the pressure inside the droplet to build up, the free-surface to converge to a smooth state and finally the static contact angle at the triple-points of the tank-propellant-gas interface to be formed when the droplet oscillations are damped. The final converged state of the system can be seen in figure 5.4(a).



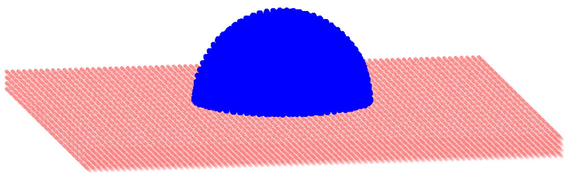
(a) $\theta_0^{\text{pre}} = 150°$



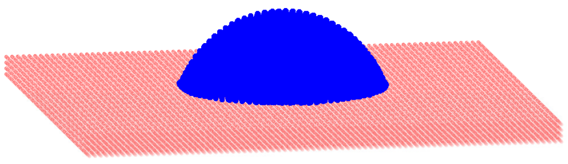
(b) $\theta_0^{\text{pre}} = 130°$



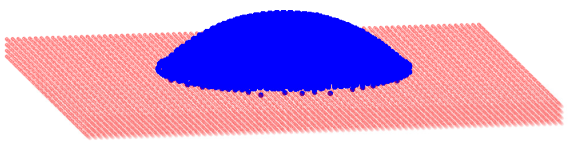
(c) $\theta_0^{\text{pre}} = 110°$



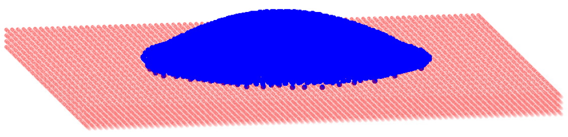
(d) $\theta_0^{\text{pre}} = 90°$



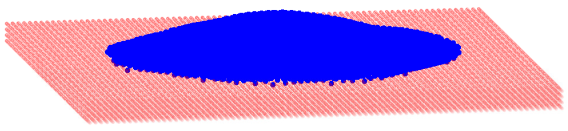
(e) $\theta_0^{\text{pre}} = 70°$



(f) $\theta_0^{\text{pre}} = 50°$



(g) $\theta_0^{\text{pre}} = 30°$



(h) $\theta_0^{\text{pre}} = 10°$

Figure 5.4: Low-gravity validation: Wetting effects: Resulting droplets (in blue) on a horizontal solid surface (in red) for prescribed static contact angles θ_0^{pre}

Starting from that initial setup, the prescribed static contact angle is reduced in 20° steps to a final static contact angle of $\theta_0 = 10°$. After a new static contact angle is prescribed, a settling time of 0.1 s is applied in order to let the liquid droplet reach an equilibrium state again. Figure 5.4 shows the resulting droplets for the prescribed static contact angles.

Please note that the prescribed surface tension $\sigma_{\alpha\beta}$ between the liquid phase α and the gas phase β as well as the prescribed static contact angle θ_0 forming at the triple-points of the solid-liquid-gas interface are not directly incorporated in the computational model, but only used to calculate the cohesive and adhesive force strengths as explained in section 2.5. As such, these molecular-like pairwise interaction forces that are used in the computational model at hand are not only able to describe static contact angles (i.e. for liquids at rest), but may also be used for simulating dynamic interface phenomena as for example contact angle hysteresis (as was demonstrated e.g. in [31]).

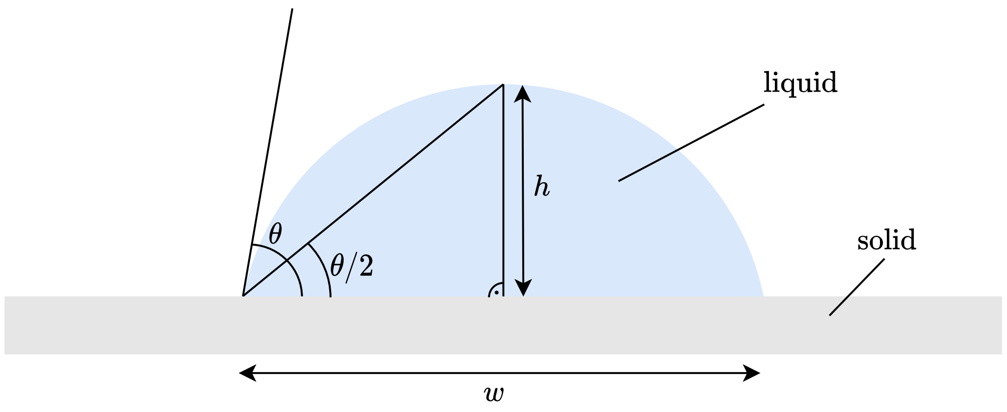


Figure 5.5: Low-gravity validation: Wetting effects: Geometrical determination of simulated static contact angles, θ_0^{sim}

To assess the quality of the computational model at hand with respect to the simulation of wetting effects, the simulated static contact angles θ_0^{sim} are determined geometrically from the simulation results and subsequently compared to the prescribed static contact angles θ_0^{pre}. Following [71], a contact angle can be measured geometrically by assuming that the droplet is part of a sphere, which is a reasonable assumption especially for a zero-gravity environment. For each droplet in figure 5.4 its width in contact with the solid, w, and height h is measured as sketched in figure 5.5. Then the corresponding simulated static contact angle θ_0^{sim} is calculated geometrically as

$$\frac{\theta_0^{\text{sim}}}{2} = \tan^{-1}\left(\frac{h}{w/2}\right) \tag{5.3.1}$$

A comparison of the prescribed static contact angles θ_0^{pre} and measured static contact angles coming from simulation results θ_0^{sim} as well as their relative errors $\delta\theta_0 = \theta_0^{\text{sim}}/\theta_0^{\text{pre}} - 1$ can be found in table 5.3. It can be observed that the static contact angles coming from simulation are reproducing the desired prescribed static contact angles very well with an average relative percentage error of 2.8 %. For the wetting contact angles (i.e. $\theta_0 < 90$ deg), the chosen numerical resolution causes the results to degrade for decreasing contact angles due to the fact that the form of the droplet, i.e. prolongated and quite thin, cannot be properly resolved anymore by a few particles only. The chosen not so fine numerical resolution of only 8,217 fluid particles for a full 3D simulation thus deteriorates the solution for low contact angles. Hence no calculated results are provided for static contact angles of $\theta_0 \leq 30°$, where the insufficient numerical resolution causes a droplet shape that cannot be treated anymore as spherical cap, as can be seen in figure 5.4(h), thus making it impossible to use the geometric contact angle measurement method as described above.

As already shown in [31] using also a molecular-like pairwise interaction force model for the description of surface tension and wetting effects, there is no systematic deviation of a simulated static contact angle using lower resolutions. To assess the dependency of the numerical approximation on the spatial resolution for the computational model at hand, the simulation of different static contact angles as explained above is repeated for a lower spatial resolution using a particle spacing of $\Delta x = d/20$, yielding 4,224 fluid particles, as well as a higher spatial resolution using a particle spacing of $\Delta x = d/32$, yielding 17,256 fluid particles. Table 5.4 shows the resulting comparison of the prescribed static contact angles θ_0^{pre} and measured static contact angles coming from simulation results θ_0^{sim} in form of the relative errors $\delta\theta_0 = \theta_0^{\text{sim}}/\theta_0^{\text{pre}} - 1$ for the three different spatial resolutions using subscript 'low' for 4,224 particles, 'mid' for 8,217 particles and 'high' for 17,256 particles.

Table 5.3: Low-gravity validation: Wetting effects: Comparison of prescribed static contact angles θ_0^{pre} and simulated static contact angles θ_0^{sim}. The relative error is given by $\delta\theta_0 = \theta_0^{\text{sim}}/\theta_0^{\text{pre}} - 1$.

θ_0^{pre} [°]	θ_0^{sim} [°]	$\delta\theta_0$ [%]
150	148.1	1.3
130	127.1	2.3
110	108.3	1.6
90	87.0	3.4
70	66.9	4.7
50	48.3	3.5

It can be noted that the static contact angles coming from simulation are reproducing the desired prescribed static contact angles very well, with an average relative percentage error of 6.1 % for the lower spatial resolution and an average relative percentage error of 0.6 % for the higher spatial resolution, clearly showing the expected convergence to the prescribed value for increasing spatial resolutions. Additionally it can be said that the results for the lower spatial resolution simulation are remarkably good especially considering the fact that only 4,224 fluid particles are being used for a full 3D simulation incorporating interface phenomena. This makes the computational model at hand especially useful for AOCS verification campaign simulations, where low-gravity environmental effects can be simulated using lower resolutions and thus in a smaller amount of time.

Table 5.4: Low-gravity validation: Wetting effects: Comparison of prescribed static contact angles θ_0^{pre} and simulated static contact angles θ_0^{sim} in form of the relative error $\delta\theta_0 = \theta_0^{\text{sim}}/\theta_0^{\text{pre}} - 1$ for three different spatial resolutions. The subscript denotes the used spatial resolution, i.e. 'low' for 4,224 particles, 'mid' for 8,217 particles and 'high' for 17,256 particles.

θ_0^{pre} [°]	$\delta\theta_{0,\text{low}}$ [%]	$\delta\theta_{0,\text{mid}}$ [%]	$\delta\theta_{0,\text{high}}$ [%]
150	2.5	1.3	-0.3
130	2.2	2.3	-1.1
110	4.6	1.6	-0.4
90	6.5	3.4	0.6
70	11.1	4.7	0.3
50	9.6	3.5	1.0

Chapter 6

Computational modeling of spacecraft attitude & orbit dynamics and control

6.1 Physics of spacecraft maneuvers

The physics relevant for spacecraft attitude maneuvers, as e.g. during an image take for an Earth observation satellite mission, and orbit maneuvers, as e.g. the de-orbiting of a satellite at the end of his lifetime, is typically given by rigid-body dynamics. Here the spacecraft structure is considered to be rigid, meaning that it doesn't deform when external forces are applied, which is a reasonable approximation of the overall system behavior excluding flexible structures like solar panels for example or liquids like the carried propellant. Flexible structures have been extensively covered in literature as e.g. in [70] and are not of primary interest for the topic discussed in this work, thus without loss of generality they are not further considered in the following. The effects of the propellant dynamics on the spacecraft and the Attitude and Orbit Control System (AOCS) including the propellant's coupling to the rigid-body dynamics is discussed in chapter 8.

The equations of motion for a rigidly assumed spacecraft are given by Euler's laws of motion for rigid bodies [70]. Rotational maneuvers are here described by use of a body reference frame B that is fixed to the spacecraft structure and aligned with its principal axes of the moment of inertia, having the spacecraft center of mass as origin, O_B. The body frame's attitude with respect to an inertial reference frame I is denoted as $\mathbf{q}_{BI}$ and its angular velocity $\vec{\boldsymbol{\omega}}_B$. The body frame is dislocated from the inertial reference frame I by position $\vec{\mathbf{r}}_{s/c}$ and moves with velocity $\vec{\mathbf{u}}_{s/c}$. A visualization of the two reference frames and its respective variables can be seen in figure 6.1.

Considering a mass element dm at position $\vec{\mathbf{R}}$ from the inertial reference frame's origin O_I and at position $\vec{\mathbf{r}}$ from the body frame's origin O_B, the total angular momentum of the spacecraft about its center of mass can be expressed in the body frame as [70]

$$\vec{\mathbf{H}}_{\text{total},B} = \left[\int \vec{\mathbf{r}} \times \frac{d\vec{\mathbf{R}}}{dt}\, dm \right]_B = \mathbf{I}_{\text{s/c}} \vec{\boldsymbol{\omega}}_B + \vec{\mathbf{h}}_{\text{int},B} \tag{6.1.1}$$

with $\mathbf{I}_{\text{s/c}}$ being the spacecraft's moment of inertia and $\vec{\mathbf{h}}_{\text{int},B}$ the internal angular momentum expressed in the body frame, as for example coming from reaction/momentum wheels or control moment gyros [22].

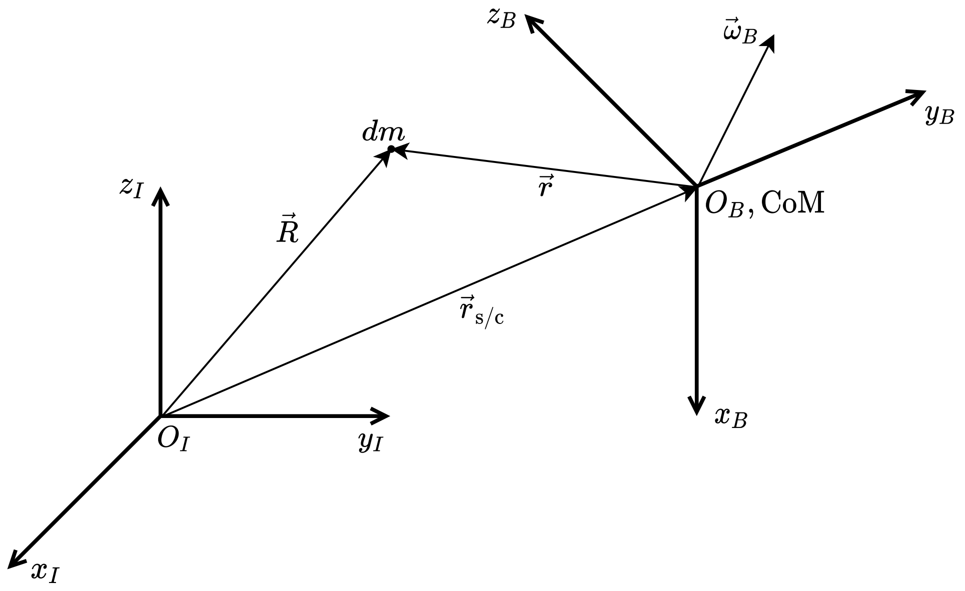


Figure 6.1: Computational spacecraft maneuver modeling: Visualization of the body reference frame B, which is fixed to the spacecraft structure and has the spacecraft center of mass as origin, dislocated with respect to an inertial reference frame I by position $\vec{\mathbf{r}}_{s/c}$ and attitude $\mathbf{q}_{BI}$ and moving with respect to inertial reference frame I by velocity $\vec{\mathbf{u}}_{s/c}$ and angular velocity $\vec{\boldsymbol{\omega}}_B$.

Euler's second law expressed in the inertial reference frame I is given as [68]

$$\vec{\mathbf{T}}_I = \frac{d\vec{\mathbf{H}}_{\mathrm{total},I}}{dt} \tag{6.1.2}$$

with $\vec{\mathbf{T}}_I$ being the external torque acting on the spacecraft and $\frac{d\vec{\mathbf{H}}_{\mathrm{total},I}}{dt}$ the time derivative of the total angular momentum of the spacecraft about its center of mass, both being expressed in the inertial reference frame I.

The time derivative of a vector $\vec{\mathbf{a}}$ in the body frame is expressed through [68]

$$\frac{d\vec{\mathbf{a}}_B}{dt} = \left[\frac{d\vec{\mathbf{a}}_I}{dt}\right]_B - \vec{\boldsymbol{\omega}}_B \times \vec{\mathbf{a}}_B \tag{6.1.3}$$

with $\left[\frac{d\vec{\mathbf{a}}_I}{dt}\right]_B$ being the time derivative of variable $\vec{\mathbf{a}}$ in the inertial frame as seen in the body frame and $\vec{\boldsymbol{\omega}}_B$ the angular velocity of the body frame with respect to the inertial reference frame, given in the body frame.

Using equation 6.1.3 with $\vec{\mathbf{a}} = \vec{\mathbf{H}}_{\mathrm{total}}$ and inserting equations 6.1.2 and 6.1.1 finally yields Euler's dynamic equation of motion for a body reference frame B fixed to a rigidly-assumed spacecraft that rotates with respect to an inertial reference frame I, expressed in the body frame, as

$$\mathbf{I}_{\mathrm{s/c}}\frac{d\vec{\boldsymbol{\omega}}_B}{dt} = \vec{\mathbf{T}}_B - \frac{d\vec{\mathbf{h}}_{\mathrm{int},B}}{dt} - \vec{\boldsymbol{\omega}}_B \times \left[\mathbf{I}_{\mathrm{s/c}}\vec{\boldsymbol{\omega}}_B + \vec{\mathbf{h}}_{\mathrm{int},B}\right] \tag{6.1.4}$$

The rotational kinematic equation of motion can be derived using a small angle approximation valid for sufficiently small attitude changes as [68, 47]

$$\frac{d\mathbf{q}_{BI}}{dt} = \frac{1}{2}\boldsymbol{\Omega} \cdot \mathbf{q}_{BI} \tag{6.1.5}$$

with $\mathbf{q}_{BI}$ being the attitude quaternion transforming from the inertial frame I to the body frame B and $\boldsymbol{\Omega}$ a skew-symmetric matrix comprising components of the angular rate vector given in the body frame $\vec{\boldsymbol{\omega}}_B = (\omega_x, \omega_y, \omega_z)$ as

$$\boldsymbol{\Omega} = \begin{pmatrix} 0 & \omega_x & \omega_y & \omega_z \\ -\omega_x & 0 & \omega_z & -\omega_y \\ -\omega_y & -\omega_z & 0 & \omega_x \\ -\omega_z & \omega_y & -\omega_x & 0 \end{pmatrix} \tag{6.1.6}$$

Please note that in this work the convention of [47] is followed by defining an attitude quaternion $\mathbf{q} = (q_0, q_1, q_2, q_3)$ for a rotation about an axis with unit vector $\vec{\mathbf{a}} = (a_1, a_2, a_3)$ by θ degrees as

$$\mathbf{q} = \begin{pmatrix} \cos\frac{\theta}{2} \\ a_1 \sin\frac{\theta}{2} \\ a_2 \sin\frac{\theta}{2} \\ a_3 \sin\frac{\theta}{2} \end{pmatrix} \tag{6.1.7}$$

The use of quaternions for describing and propagating orientations in three-dimensional space is preferred due to their lack of singularities as compared to Euler angles as well as due to their computational efficiency as compared to direction cosine matrices [70, 47].

Translational maneuvers of a rigidly assumed spacecraft are described in an inertial reference frame I by Euler's dynamic equation of motion for rigid bodies [70]

$$m_{s/c}\frac{d\vec{\mathbf{u}}_{s/c}}{dt} = \vec{\mathbf{F}} \tag{6.1.8}$$

with $m_{s/c}$ being the spacecraft's total mass, $\vec{\mathbf{u}}_{s/c}$ the velocity of the spacecraft's center of mass and $\vec{\mathbf{F}}$ the resultant of the external forces acting on the spacecraft.

The translational kinematic equation of motion is finally given by

$$\frac{d\vec{\mathbf{r}}_{s/c}}{dt} = \vec{\mathbf{u}}_{s/c} \tag{6.1.9}$$

with $\vec{\mathbf{r}}_{s/c}$ being the position of the spacecraft's center of mass. Please note that all vectors in equations 6.1.8 and 6.1.9 are given in the inertial reference frame I.

6.2 Temporal discretization

The set of equations describing rotational maneuvers, eqs. 6.1.4 and 6.1.5, and translational maneuvers, eqs. 6.1.8 and 6.1.9, are coupled first-order ordinary differential equations, which can be generalized to the vector form

$$\frac{d\vec{\mathbf{y}}(t)}{dt} = \vec{\mathbf{f}}(t, \vec{\mathbf{y}}(t)) \tag{6.2.1}$$

with t being the independent variable of time, $\vec{\mathbf{y}}(t)$ the dependent unknown variable, also called state variable, given as

$$\vec{\mathbf{y}}(t) = \begin{pmatrix} \vec{\boldsymbol{\omega}}_B(t) \\ \mathbf{q}_{BI}(t) \\ \vec{\mathbf{u}}_{s/c}(t) \\ \vec{\mathbf{r}}_{s/c}(t) \end{pmatrix} \tag{6.2.2}$$

and $\vec{\mathbf{f}}(t, \vec{\mathbf{y}}(t))$ a known function of t and $\vec{\mathbf{y}}(t)$ expressed as

$$\vec{\mathbf{f}}(t, \vec{\mathbf{y}}(t)) = \begin{pmatrix} \mathbf{I}_{\mathrm{s/c}}^{-1} \cdot \left(\vec{\mathbf{T}}_B - \frac{d\vec{\mathbf{h}}_{\mathrm{int},B}}{dt} - \vec{\boldsymbol{\omega}}_B \times \left[\mathbf{I}_{\mathrm{s/c}} \vec{\boldsymbol{\omega}}_B + \vec{\mathbf{h}}_{\mathrm{int},B} \right] \right) \\ \frac{1}{2} \boldsymbol{\Omega} \cdot \mathbf{q}_{BI} \\ \frac{1}{m_{s/c}} \vec{\mathbf{F}} \\ \vec{\mathbf{u}}_{s/c} \end{pmatrix} \tag{6.2.3}$$

The initial condition is given by $\vec{\mathbf{y}}(0) = \vec{\mathbf{y}}^0 = \left(\vec{\boldsymbol{\omega}}_B^0,\ \mathbf{q}_{BI}^0,\ \vec{\mathbf{u}}_{s/c}^0,\ \vec{\mathbf{r}}_{s/c}^0 \right)$.

A wide variety of time integration methods exists for solving equation 6.2.1, as is detailed for example in general in [49] or space engineering specific in [47]. Due to the problems to be simulated in this work and the usually chosen small enough time step size, especially when the rigid body dynamics is coupled to the propellant dynamics as explained in chapter 8, the stiffness of the differential equations is not considered to be a problem, thus in the following only explicit time integration schemes are used, which have the advantage of reduced computational cost and ease of implementation as compared to implicit methods.

A traditionally used time integration scheme for solving the equations of motion for a rigidly assumed spacecraft is the Fourth-order Runge-Kutta method (RK4). Here, equation 6.2.1 is discretized in time as [49]

$$\vec{\mathbf{y}}^{t+\Delta t} = \vec{\mathbf{y}}^t + \Delta t \frac{1}{6} \left(\vec{\mathbf{k}}_1 + 2\vec{\mathbf{k}}_2 + 2\vec{\mathbf{k}}_3 + \vec{\mathbf{k}}_4 \right) + \mathcal{O}(\Delta t^5) \tag{6.2.4}$$

$$\vec{\mathbf{k}}_1 = \vec{\mathbf{f}}\left(t, \quad \vec{\mathbf{y}}^t \right) \tag{6.2.5}$$

$$\vec{\mathbf{k}}_2 = \vec{\mathbf{f}}\left(t + \frac{\Delta t}{2},\ \vec{\mathbf{y}}^t + \frac{\Delta t}{2} \vec{\mathbf{k}}_1 \right) \tag{6.2.6}$$

$$\vec{\mathbf{k}}_3 = \vec{\mathbf{f}}\left(t + \frac{\Delta t}{2},\ \vec{\mathbf{y}}^t + \frac{\Delta t}{2} \vec{\mathbf{k}}_2 \right) \tag{6.2.7}$$

$$\vec{\mathbf{k}}_4 = \vec{\mathbf{f}}\left(t + \Delta t,\ \vec{\mathbf{y}}^t + \Delta t \vec{\mathbf{k}}_3 \right) \tag{6.2.8}$$

with $\vec{\mathbf{y}}^{t+\Delta t}$ and $\vec{\mathbf{y}}^t$ being the state vectors of the new and old time step, respectively, when advancing the solution from time t to $t + \Delta t$, and $\mathcal{O}(\Delta t^5)$ the local truncation error.

When coupling the rigid body dynamics describing the rotational and translational motion of the spacecraft, as presented in section 6.1, with the propellant dynamics describing the sloshing motion inside a partially filled tank onboard the spacecraft, as presented in section 2.1, it is important to use a unified numerical integration method in order to minimize discretization errors. Since a Leapfrog integration scheme has been used for the temporal discretization of the propellant's equations of motion as presented in section 2.6, for the spacecraft's rigid body equations of motion also a Leapfrog integration scheme is applied.

The second-order Leapfrog time integration scheme presented in the following is based on a predictor–corrector method for rigid body motion from [52] and starts by integrating the position of the rigid body to the new time step by use of the midpoint velocity as

$$\vec{\mathbf{r}}_{s/c}^{t+\Delta t} = \vec{\mathbf{r}}_{s/c}^{t} + \Delta t\ \vec{\mathbf{u}}_{s/c}^{t+\frac{1}{2}\Delta t} \tag{6.2.9}$$

Then in the predictor step the remaining state variables are calculated for the new time step. The translational and angular velocities for the new time step are predicted using the translational and angular accelerations from the previous time step and the translational and angular velocities from the midpoint time step as

$$\vec{\mathbf{u}}_{s/c}^{t+\Delta t} = \vec{\mathbf{u}}_{s/c}^{t+\frac{1}{2}\Delta t} + \frac{1}{2}\Delta t \left(\frac{d\vec{\mathbf{u}}_{s/c}}{dt}\right)^{t} \tag{6.2.10}$$

$$\vec{\boldsymbol{\omega}}_{B}^{t+\Delta t} = \vec{\boldsymbol{\omega}}_{B}^{t+\frac{1}{2}\Delta t} + \frac{1}{2}\Delta t \left(\frac{d\vec{\boldsymbol{\omega}}_{B}}{dt}\right)^{t} \tag{6.2.11}$$

The attitude for the new time step is predicted using the attitude at the midpoint time step and the angular velocity at the three-quarter time step as

$$\vec{\boldsymbol{\omega}}_{B}^{t+\frac{3}{4}\Delta t} = \vec{\boldsymbol{\omega}}_{B}^{t+\frac{1}{2}\Delta t} + \frac{1}{4}\Delta t \left(\frac{d\vec{\boldsymbol{\omega}}_{B}}{dt}\right)^{t} \tag{6.2.12}$$

$$\mathbf{q}_{BI}^{t+\Delta t} = \mathbf{q}_{BI}^{t+\frac{1}{2}\Delta t} + \frac{1}{2}\Delta t \left[\frac{1}{2}\boldsymbol{\Omega}\left(\vec{\boldsymbol{\omega}}_{B}^{t+\frac{3}{4}\Delta t}\right)\cdot \mathbf{q}_{BI}^{t+\frac{1}{2}\Delta t}\right] \tag{6.2.13}$$

With the state vector at the new time step, $\vec{\mathbf{y}}^{t+\Delta t} = \left(\vec{\boldsymbol{\omega}}_{B}^{t+\Delta t},\ \mathbf{q}_{BI}^{t+\Delta t},\ \vec{\mathbf{u}}_{s/c}^{t+\Delta t},\ \vec{\mathbf{r}}_{s/c}^{t+\Delta t}\right)$, the equations of motion in eq. 6.2.1 are integrated in time to find the corrected new state vector at time $t + \frac{3}{2}\Delta t$ as

$$\vec{\mathbf{y}}^{t+\frac{3}{2}\Delta t} = \vec{\mathbf{y}}^{t+\frac{1}{2}\Delta t} + \Delta t\ \vec{\mathbf{f}}\left(t+\Delta t,\ \vec{\mathbf{y}}^{t+\Delta t}\right) + \mathcal{O}(\Delta t^{3}) \tag{6.2.14}$$

This completes the time integration cycle and the newly calculated translational and rotational velocities as well as the attitude at time $t+\frac{3}{2}\Delta t$ are in the next time step used as midpoint values.

Please note that the forces and torques inside the function $\vec{\mathbf{f}}$ as described in equation 6.2.3 are only evaluated once per time step, which is specifically important when incorporating propellant sloshing effects as described in chapter 8.

Additionally, for comparison of the two previously described time integration schemes and for testing purposes, the first-order forward Euler method is introduced. Here, equation 6.2.1 is discretized in time as [49]

$$\vec{\mathbf{y}}^{t+\Delta t} = \vec{\mathbf{y}}^{t} + \Delta t\ \vec{\mathbf{f}}\left(t,\ \vec{\mathbf{y}}^{t}\right) + \mathcal{O}(\Delta t^{2}) \tag{6.2.15}$$

Please note that the forward Euler method is only used for validation and testing purposes in this work and that it is recommended to not apply this method to any real-world application due to its low accuracy and weak stability properties as compared to other integration schemes as for example the two methods introduced before.

6.3 Spacecraft attitude & orbit control

The task of an Attitude & Orbit Control System (AOCS) is to bring the spacecraft into a desired rotational and translational state, which is derived by requirements coming from the spacecraft's payload (e.g. pointing requirements), from other spacecraft sub-systems (e.g. thermal, power or communication requirements) or from the spacecraft's health (e.g. safe mode requirements).

In order to successfully complete this task, the AOCS requires sensors like star sensors, sun sensors or magnetometers that provide information needed to determine the current state of the spacecraft as well as actuators like thrusters, reaction wheels or control moment gyros that maneuver the spacecraft in its desired state.

The AOCS hardware comprising sensors and actuators is managed by the AOCS software running on the spacecraft's On-board computer (OBC), which generally performs the tasks of state determination, guidance, control and command as visualized in figure 6.2, and is designed to provide the required stability and performance.

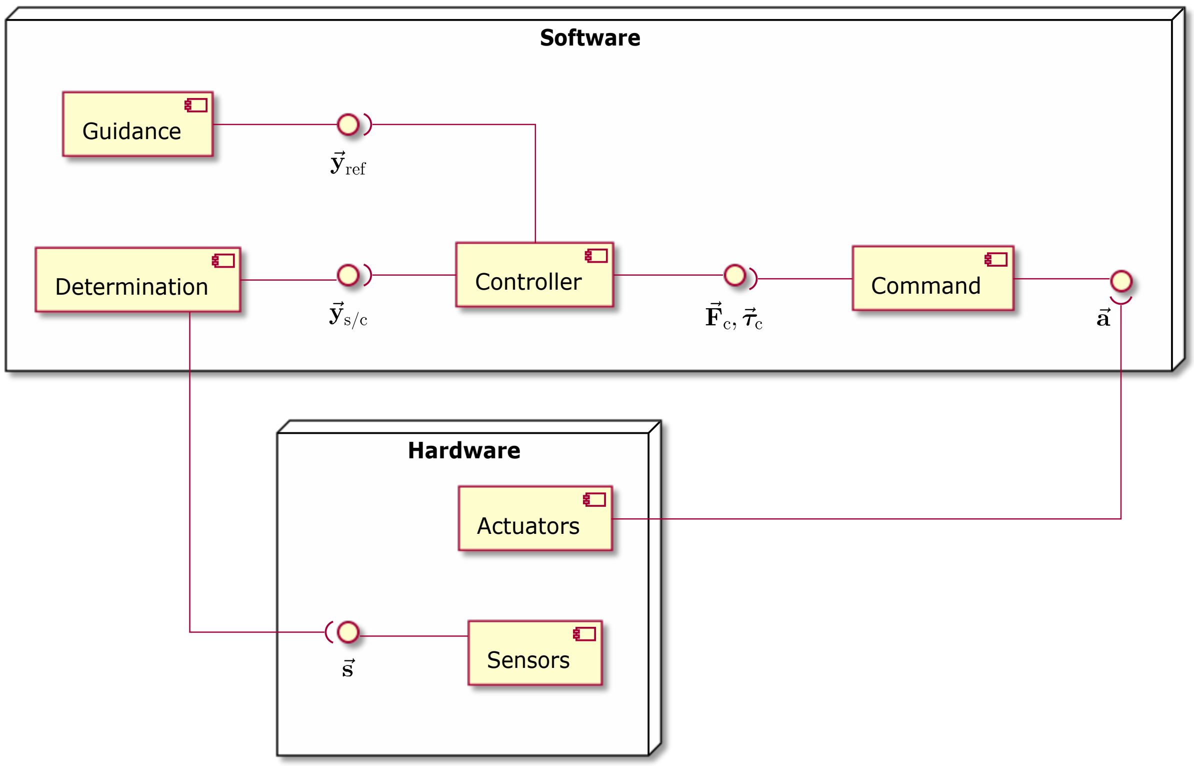

Software
Guidance
$\vec{\mathbf{y}}_{\text{ref}}$
Determination
$\vec{\mathbf{y}}_{\text{s/c}}$
Controller
$\vec{\mathbf{F}}_{\text{c}}, \vec{\boldsymbol{\tau}}_{\text{c}}$
Command
$\vec{\mathbf{a}}$
Hardware
Actuators
$\vec{\mathbf{s}}$
Sensors

Figure 6.2: Computational spacecraft maneuver modeling: AOCS components overview: Hardware comprising sensors and actuators, software running on the OBC comprising the modules state determination, guidance, control and command

The duty cycle of a typical AOCS software starts with the *determination* module, which takes as input the sensor data coming from all available sensors, denoted as $\vec{\mathbf{s}}$ in figure 6.2. This sensor data is used to calculate the current spacecraft's state $\vec{\mathbf{y}}_{s/c} = \left(\vec{\boldsymbol{\omega}}_B,\ \mathbf{q}_{BI},\ \vec{\mathbf{u}}_{s/c},\ \vec{\mathbf{r}}_{s/c}\right)$ as introduced in section 6.2. The AOCS *guidance* module determines the desired state $\vec{\mathbf{y}}_{\text{ref}}$, which the spacecraft is supposed to take to fulfill the mission requirements at the current instance of time. The current spacecraft state $\vec{\mathbf{y}}_{s/c}$ as well as the desired state $\vec{\mathbf{y}}_{\text{ref}}$ are fed into the AOCS *control* module, which calculates the needed forces $\vec{\mathbf{F}}_c$ and torques $\vec{\boldsymbol{\tau}}_c$ to maneuver the spacecraft in such a way that the error between these two states vanishes. The control forces $\vec{\mathbf{F}}_c$ and torques $\vec{\boldsymbol{\tau}}_c$ are subsequently passed to the AOCS *command* module, which determines actuator commands $\vec{\mathbf{a}}$ comprising for every actuator the needed input (e.g. a specific voltage for the reaction wheels) to produce in sum the desired forces and torques. The actuators are then activated and produce in sum a certain total force and torque, which causes the spacecraft to change its rotational and translational state. This closes the duty cycle of the AOCS software and the process starts from scratch with the AOCS *determination* module.

More details on the architecture, design and operations of an AOCS can be found for example in [64] and [68]. Important for the work at hand is an attitude controller capable of bringing the spacecraft into a desired rotational state based on given attitude maneuvers. Hence the constant-gain quaternion-error feedback control logic with variable limiter from [70] is described in the following that is used as feedback part inside the AOCS *control* module while performing the coupled spacecraft-propellant sloshing simulations in chapter 9.

The three-axis proportional-integral-derivative (PID) attitude controller presented in [70] is designed for agile spacecraft that shall perform large-angle reorientation maneuvers as fast as possible without exceeding a given maximum angular velocity, without excessive transient overshoot and without loosing its accuracy when only small deviations from the desired attitude are present.

In a first step, the attitude error Euler vector $\vec{\mathbf{e}}_q$ is computed, which is a vector representing the Euler axis and angle needed to rotate from the current attitude to the desired reference attitude given by the AOCS *guidance* module. This measure of the attitude error is determined by first calculating the attitude error quaternion $\mathbf{q}_e = (q_{e1}, q_{e2}, q_{e3}, q_{e4})$ by

$$\mathbf{q}_e = \begin{pmatrix} q_{\text{ref},1} & -q_{\text{ref},2} & -q_{\text{ref},3} & -q_{\text{ref},4} \\ q_{\text{ref},2} & q_{\text{ref},1} & -q_{\text{ref},4} & q_{\text{ref},3} \\ q_{\text{ref},3} & q_{\text{ref},4} & q_{\text{ref},1} & -q_{\text{ref},2} \\ q_{\text{ref},4} & -q_{\text{ref},3} & q_{\text{ref},2} & q_{\text{ref},1} \end{pmatrix} \begin{pmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{pmatrix} \tag{6.3.1}$$

using the desired reference attitude $\mathbf{q}_{BI}^{\text{ref}} = (q_{\text{ref},1}, q_{\text{ref},2}, q_{\text{ref},3}, q_{\text{ref},4})$ and the current attitude $\mathbf{q}_{BI}^{\text{s/c}} = (q_1, q_2, q_3, q_4)$.

Then, based on the general definition of the quaternion $\mathbf{q}_e = (\cos\theta/2, u_1 \cdot \sin\theta/2, u_2 \cdot \sin\theta/2, u_3 \cdot \sin\theta/2)$ with $\vec{\mathbf{u}} = (u_1, u_2, u_3)$ being the unit vector describing the Euler axis and θ the rotation angle about the Euler axis, the Euler vector representation of the attitude error is determined by

$$\vec{\mathbf{e}}_q = \theta \cdot \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} \tag{6.3.2}$$

Subsequently, the angular velocity error vector $\vec{\mathbf{e}}_\omega$ is calculated using the current angular velocity $\vec{\boldsymbol{\omega}}_B^{\text{s/c}}$ and the desired reference angular velocity $\vec{\boldsymbol{\omega}}_B^{\text{ref}}$ given by the AOCS *guidance* module as

$$\vec{\mathbf{e}}_\omega = \vec{\boldsymbol{\omega}}_B^{\text{s/c}} - \vec{\boldsymbol{\omega}}_B^{\text{ref}} \tag{6.3.3}$$

In the next step, the feedback control torque $\vec{\boldsymbol{\tau}}_{\text{fb}}$ is determined by [70]

$$\vec{\boldsymbol{\tau}}_{\text{fb}} = -\mathbf{K} \underset{L_i}{\text{sat}} \left(\vec{\mathbf{e}}_q + \frac{1}{T} \int \vec{\mathbf{e}}_q \right) - \mathbf{C}\, \vec{\mathbf{e}}_\omega \tag{6.3.4}$$

with T being the time constant of the integral control, $\mathbf{K}$ the proportional gain regarding attitude and $\mathbf{C}$ the derivative gain regarding angular velocity, the latter two given as

$$\mathbf{K} = 2k\, \mathbf{I}_{\mathrm{s/c}} \qquad k = \omega_n^2 + \frac{2\zeta\omega_n}{T} \tag{6.3.5}$$

$$\mathbf{C} = c\, \mathbf{I}_{\mathrm{s/c}} \qquad c = 2\zeta\omega_n + \frac{1}{T} \tag{6.3.6}$$

with $\mathbf{I}_{\mathrm{s/c}}$ being the spacecraft's moment of inertia, ω_n the control bandwidth and ζ the damping ratio.

The variable limit L_i in eq. 6.3.4 is calculated for each control axis $i = \{1, 2, 3\}$ in 3D space and is given as

$$L_i = \frac{c}{2k} \min \left\{ \sqrt{4\ |a_i|_{\max}\ |e_{q,i}|},\ |\omega_i|_{\max} \right\} \tag{6.3.7}$$

with $|a_i|_{\max}$ being the maximum control acceleration about the i-th axis, $e_{q,i}$ the i-th component of the attitude error Euler vector $\vec{\mathbf{e}}_q$ and $|\omega_i|_{\max}$ the maximum angular velocity about the i-th axis.

This variable limiter ensures that a given maximum angular velocity is not exceeded. Additionally it allows to use a higher control torque in case the attitude error is big, hence yielding faster reorientation times. When the attitude error becomes smaller, also the control torque is decreasing, thus allowing to accurately control small attitude deviations and also preventing excessive transient overshoots.

Please note that in order to eliminate the integrator windup effect in the presented PID controller, the integral part of the attitude error Euler vector in eq. 6.3.4 is reset whenever the control saturation is reached.

Finally, the total control torque $\vec{\boldsymbol{\tau}}_c$ fed into the AOCS *command* module is calculated by

$$\vec{\boldsymbol{\tau}}_c = \vec{\boldsymbol{\tau}}_{\mathrm{ff}} + \vec{\boldsymbol{\tau}}_{\mathrm{fb}} \tag{6.3.8}$$

with $\vec{\boldsymbol{\tau}}_{\mathrm{ff}}$ being the feedforward control torque coming from the AOCS *guidance* module and $\vec{\boldsymbol{\tau}}_{\mathrm{fb}}$ the previously determined feedback control torque.

For the simulations performed in this work, the following values are set for the parameters of the attitude controller as described above:

$$\begin{aligned} T &= 10 \text{ s} \\ \omega_n &= 1 \text{ rad/s} \\ \zeta &= 0.9 \\ \omega_{\max,i} &= 10 \text{ deg/s} \\ a_{\max,i} &= 40\% \cdot \tau_{\max,i} \ / \ \left(\mathbf{I}_{\mathrm{s/c}}\right)_{ii} \end{aligned}$$

with $\tau_{\max,i} = 1000$ Nm being the maximum control torque available.

Chapter 7

Software implementation of the spacecraft attitude & orbit dynamics and control model

7.1 Requirements specification

The computational spacecraft attitude & orbit dynamics and control model described in chapter 6 has been developed to support the simulation of fast large-angle reorientation maneuvers as are required for example for agile Earth observation satellite missions that are capturing multiple targets along their orbital path and thus need fast retargeting capabilities. The overall goal is to create a simulation environment, with which attitude maneuvers including propellant sloshing effects through coupling with the computational model developed in chapter 2 can be simulated. As such, the following requirements are derived for the software implementation of the computational spacecraft attitude & orbit dynamics and control model:

RA The software shall provide a generic way of simulating the rotational and translational rigid body dynamics of a spacecraft.
Rationale: In order to support the full range of possible attitude and orbit maneuvers of a spacecraft, the software shall not be restricted to a certain use case, but offer functionality that allows for example to simulate spacecraft reorientation maneuvers in a low-Earth orbit, spacecraft interplanetary trajectories to Mars or deep-space scanning retargeting in the L_2 point of the Sun-Earth system using the same software but different configuration.

RB The software shall provide a generic way of simulating the Attitude & Orbit Control System (AOCS) of a spacecraft.
Rationale: The software shall be designed in a way that the AOCS can be simulated using different implementation levels of the involved AOCS modules determination, guidance, control and command. This means that for example a simulation can be performed using a full-blown AOCS control module, but with an AOCS command module that is just passing through the control forces and torques. This allows to focus on dedicated AOCS modules individually while having an initial idealistic implementation of other AOCS modules that can be implemented in a more realistic way at a later stage whenever required.

RC It shall be possible to include arbitrary AOCS hardware, i.e. sensors and actuators, in the simulation environment.
Rationale: All actuators used by the AOCS have in common that they produce forces and/or torques given an input signal like e.g. voltage. All sensors have in common that they measure some part of the physical environment and deliver their measured signal in

form of e.g. a current to the AOCS. This information can be used to create a generic interface so that arbitrary sensors and actuators can be simulated, thus offering flexibility in the choice of used AOCS hardware.

RD It shall be possible to couple the software comprising the spacecraft AOCS simulator to a computational propellant sloshing model.
Rationale: During an AOCS verification campaign it is desired to couple the spacecraft through its rigid-body dynamics with the solid tank structure of the propellant sloshing model described in this work. This offers the possibility to analyze the impact of the sloshing propellant on the spacecraft structure and thus on the AOCS.

RE The software shall offer a convenient way of configuring the simulation to be performed.
Rationale: In order to avoid changes of the software's source code when performing different simulations, the entire functionality of the spacecraft AOCS simulator shall be controlled and set solely by configuration parameters that reside outside of the software's source code in a dedicated config file for each simulation.

RF There shall be an easy to use post-processing tool for data visualization.
Rationale: It shall be possible to conveniently visualize, analyze and export the simulation results coming from the spacecraft AOCS simulator.

RG The software shall run on a Linux operating system.
Rationale: AOCS verification campaigns, which comprise a large amount of simulations that can be performed mostly in parallel, are the main target of the software to be developed. Hence it is straightforward to run the software in a cloud infrastructure making use of its high scalability and on-demand availability. Multiple cloud providers and cloud computing service models exist, which all support the Linux operating system.

7.2 Design overview

Following the requirements in section 7.1, a software is written that implements the computational spacecraft attitude & orbit dynamics and control model described in chapter 6. The software implementation basically consists of a spacecraft AOCS simulator and a post-processing tool, which can be seen in figure 7.1 along with their interfaces.

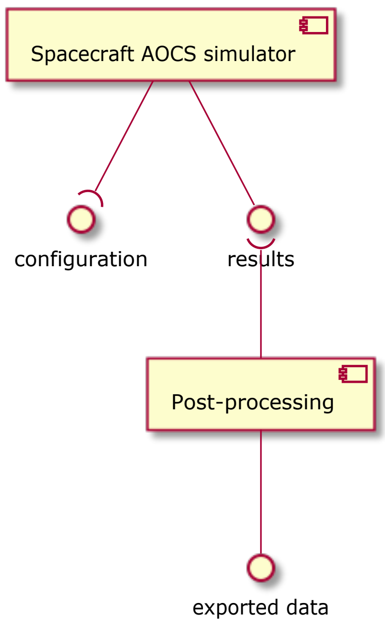


Figure 7.1: Components of the software implementation of the spacecraft AOCS simulator including interfaces.

The spacecraft AOCS simulator is a newly developed software written in object-oriented Matlab [62] and takes as input a configuration file. The configuration file comprises a structure array that contains all parameters needed for the simulation to run. An excerpt of such a configuration file can be found in figure 7.2. There, the entire functionality of the spacecraft AOCS simulator can be controlled through parameters in a convenient way, closing requirement [RE], as for example the number and configuration of spacecraft (`par.sc{i}` with `i` being the number of the spacecraft), the hardware components including their configuration used within each spacecraft (`par.sc{i}.hardware{j}` with `j` being the number of the hardware component) or the type of controller used within the spacecraft's AOCS (`par.sc{i}.aocs.control.type`).

The spacecraft AOCS simulator is written in Matlab, since it offers a feature-rich set of tools for solving numerical problems, allows to use convenient high-level abstractions through its object-oriented programming (OOP) capabilities and is also typically well known to AOCS engineers. A general overview of the spacecraft AOCS simulator developed in this work can be found in figure 7.3, where all classes and their relationships are drawn.

The main class instantiated at the beginning of a simulation using the previously described configuration parameters structure array is the *Simulation* class. It stores the configuration provided as input and sets up one or multiple *Spacecraft* class instances based on the configuration. Additionally it instantiates a *Data* class object that is used to store and manage the simulation results. Finally it creates a *Solver* class instance using the desired integration scheme

```
% ...

% Initial attitude and angular rate at simulation start
par.sc{1}.att_q_B_I = [1; 0; 0; 0];
par.sc{1}.angRate_B = [0; 0; 0]; % [rad/s]

% Declare S/C hardware
par.sc{1}.hardware{1}.type = HARDWARE_TYPES.PERFECT_ACTUATOR;

par.sc{1}.hardware{2}.type = HARDWARE_TYPES.TANK_CFD;
par.sc{1}.hardware{2}.boundaryParticleId = 11;
par.sc{1}.hardware{2}.timeStepSize = par.sim.timeStepSize;

% AOCS parameters
par.sc{1}.aocs.obcTimeStepSize = par.sim.timeStepSize; % [s]

% Guidance: Calculate guidance profile for first slew maneuver
% - Define parameters
maneuverParams = [];
maneuverParams.slewAngle = 50 * pi/180; % [rad]
maneuverParams.slewUnitVector_B = [0 1 0];
maneuverParams.slewDuration = 5; % [s]
maneuverParams.timeStepSize = par.sim.timeStepSize; % [s]
maneuverParams.initial_att_q_B_I = [1; 0; 0; 0];
maneuverParams.initial_timeOffset = 10; % [s]
% - Obtain the guidance profile state
profileState_man1 = createGuidanceProfile(MANEUVER_TYPES.BANG_BANG, maneuverParams);

% Torque profile (time, torque_x, torque_y, torque_z)
par.sc{1}.aocs.guidance.mode = GUIDANCE_MODES.FULL_ROTATIONAL;
par.sc{1}.aocs.guidance.profileRotationalState = profileState_man1;

% Control
par.sc{1}.aocs.control.type = CONTROL_TYPES.PID;
```

Figure 7.2: Spacecraft AOCS simulator: Excerpt from a configuration file comprising a subset of parameters needed to setup a spacecraft AOCS simulation. The structure array `par.sc{1}` contains the configuration for the first spacecraft in the simulation.

from the configuration, which then provides a function to start the time integration.

The *Data* class comprises the data structures to store the simulation results. It provides functionality to analyze this data and can also be used for basic data visualization by plotting (also to file) selected variables in an automated way. Finally it offers the functionality to export selected variables to a comma-separated value (CSV) file and/or to export the entire data set in Matlab's HDF5 based binary compressed file format.

The *Solver* class contains the numerical integration schemes presented in section 6.2, i.e. the fourth-order Runge-Kutta method (RK4), the second-order Leapfrog method as well the first-order forward Euler method. The time integration method to use for the simulation is selected in the configuration file. During the time integration itself the rigid body dynamics are evaluated in a generic way, offering the possibility to simulate all kind of rotational and translational maneuvers, thus closing requirement [RA]. Finally, the class provides a function to start the time integration for all defined spacecraft of the simulation.

All spacecraft relevant information is kept in the *Spacecraft* class, which comprises the spacecraft's hardware (like thrusters or reaction/momentum wheels) through a *HardwareCollection* class instance, the spacecraft's software contained in the AOCS through a *AOCS* class instance as well as the physics relevant for the spacecraft through a *Physics* class instance. The *Spacecraft*

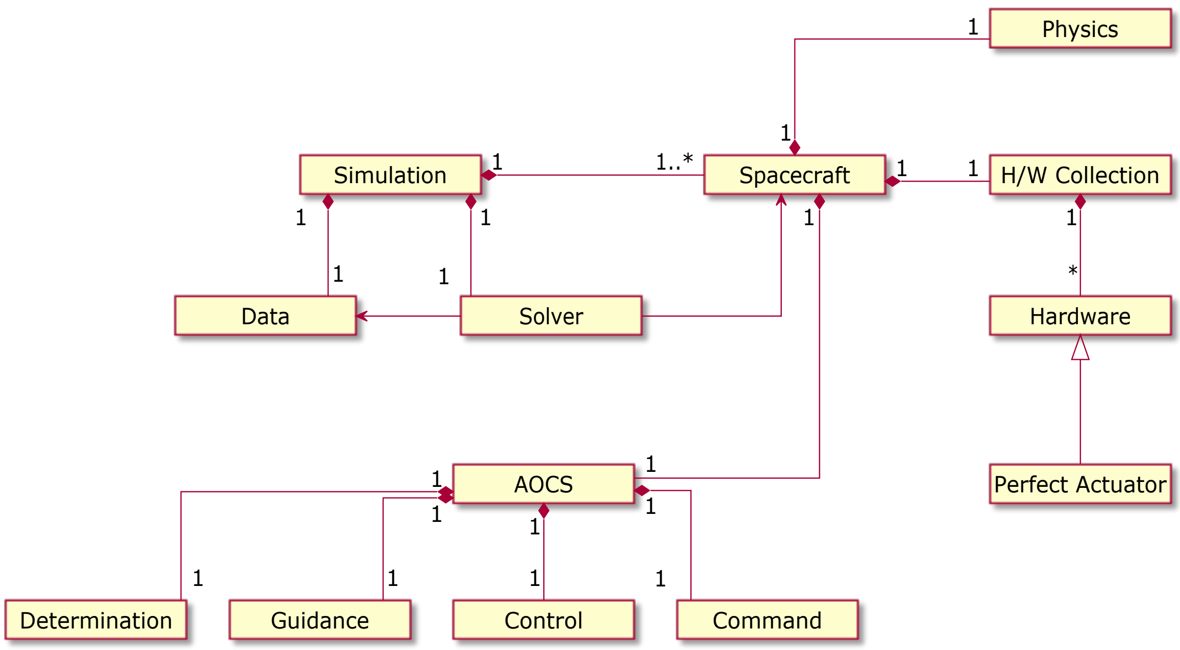


Figure 7.3: Spacecraft AOCS simulator: Overview of classes of the simulation environment including their relationships

class manages the state of the spacecraft as defined in equation 6.2.2 and is also used to obtain the total forces and torques acting on the spacecraft, coming from physics as well as from the hardware in the hardware collection.

The *HardwareCollection* class contains zero or more instances of the *Hardware* class and offers a functionality to obtain the states of the assigned hardware as for example the current angular momentum of a reaction wheel. Additionally it is also used to obtain the total forces and torques acting on the spacecraft, coming from the assigned hardware in the hardware collection. The *Hardware* class itself is an abstract base class that defines a basic structure common to all hardware types. The class *PerfectActuator* is a trivial example of an implementation of the *Hardware* class, offering an actuator that produces the forces and torques exactly as commanded. A more sophisticated hardware implementation, i.e. a partially filled spacecraft propellant tank, will be introduced in section 8.2. This flexible way of including arbitrary AOCS hardware like sensors or actuators closes requirement [RC].

The *Physics* class implements the physics relevant for the spacecraft, as for example gravity forces with respect to one or multiple bodies. It is also used to obtain physically based forces and torques acting on the spacecraft.

The *AOCS* class implements the AOCS software relevant for the simulations in this work. It comprises instances of the generic AOCS sub-classes *Determination*, *Guidance*, *Control* and *Command*. Additionally, it takes care of the OBC time and calls all AOCS sub-classes instances to evaluate them for the respective instance in time.

The current state of the spacecraft is predicted based on available sensor data in the *Determination* class. The *Guidance* class provides reference spacecraft states based on the selected guidance mode in the configuration file. The *Control* class offers functionality to calculate the control forces and torques based on the current spacecraft state coming form the *Determination* class instance and the desired spacecraft state coming from the *Guidance* class instance. As example for a PID control law, the constant-gain quaternion-error feedback control logic with variable limiter described in section 6.3 is implemented in the *Control* class. The desired control forces and torques coming from the *Control* class instance are translated into input signals for the respective hardware (like a certain voltage for a reaction wheel for example) inside the *Command* class.

Through this approach of implementing the AOCS software in a generic object-oriented way, the individual AOCS modules can be implemented in a detail level corresponding to the requirements of the spacecraft mission to be simulated, thus closing requirement [RB]. This also completes the overview of the classes of the spacecraft AOCS simulator.

Finalizing the discussion of figure 7.1, during a simulation, the spacecraft AOCS simulator generates output data in form of a comma-separated value (CSV) file, storing information of selected simulation parameters like the angular momentum of the spacecraft, and/or a Matlab specific binary compressed file based on HDF5, storing all simulation data available.

The simulation results coming from the spacecraft AOCS simulator are fed into a newly developed post-processing tool written in Matlab that has been explained in detail already in section 3.2, where it has been used for the post-processing of data coming from the propellant sloshing simulator. In the same way the post-processing tool is used for propellant sloshing system states, it is also used for the display, analysis and export of data coming from the spacecraft AOCS simulator, as for example the spacecraft's angular momentum or the center of mass motion of the spacecraft. This then also closes requirement [RF]. Additionally, requirement [RG] is fulfilled trivially since the Matlab environment runs on the Linux operating system.

Since the tasks described in figure 7.1 above are reoccurring, it is important to have the spacecraft AOCS simulation end-to-end process fully automated so that simulations including the analysis and export of the resulting data can be performed easily in the frame of AOCS verification campaigns, where a large number of simulations is performed.

In the current work, the main interest lies in the computational propellant slosh modeling for AOCS verification campaigns and not in the complete design of a spacecraft AOCS. As such, it is not required to provide details for all AOCS software modules mentioned before and some simplifications can be done without loss of generality. Hence for the simulations performed in this work, it is assumed that the actuators are producing the forces and torques exactly as commanded, i.e. without any error. Additionally it is also assumed that the rotational and translational state of the spacecraft is exactly known. Consequently also the AOCS sub-classes *Determination* as well as *Command* are only passing on the values they obtain.

Please note that the remaining open requirement [RD] about the coupling of the spacecraft AOCS simulator with a propellant sloshing simulator is discussed and closed in chapter 8.

7.3 Validation

In order to validate the software implementation of the spacecraft attitude dynamics & control model presented in chapter 6, in the following the rigid body example 6.1 of [47] is simulated comparing the three time integration schemes described in section 6.2 for multiple temporal resolutions.

Example 6.1 in [47] describes the dynamics of a rigid body with a moment of inertia of

$$\mathbf{I} = \begin{pmatrix} 2000 & 300 & -200 \\ 300 & 4000 & 1000 \\ -200 & -100 & 1000 \end{pmatrix} \text{ kg m}^2 \tag{7.3.1}$$

undergoing a rotation due to a constant inertial torque about the z-axis of $\vec{\boldsymbol{\tau}}_I = \begin{pmatrix} 0 \\ 0 \\ 0.1 \end{pmatrix}$ Nm.

With the rigid body being initially at rest, for each of the three time integration methods the simulation is run for 1000 s with time step sizes of $\Delta t \in \{1.0, 0.25, 0.1, 0.025, 0.01, 0.0025, 0.001\}$ s. Figure 7.4 shows as an example the plot of the resulting inertial angular momentum $\vec{\mathbf{H}}_I$ for the Leapfrog time integration scheme and a time step size of $\Delta t = 0.25$ s. The angular momentum in z-direction is building up as expected, whereas in x-/y-direction small unphysical oscillations can be seen that stem from noise due the numerical time integration.

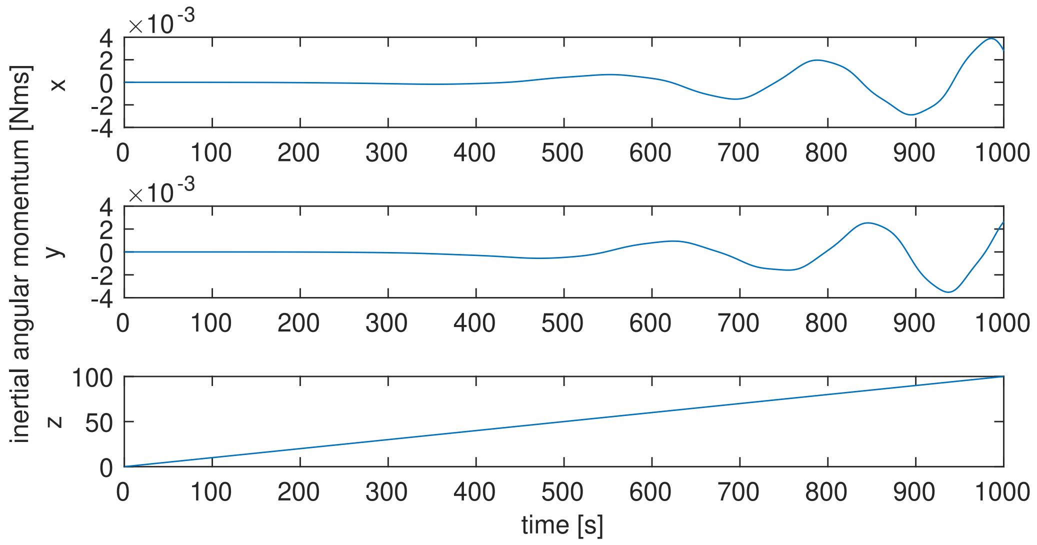


Figure 7.4: Spacecraft AOCS simulator: Inertial angular momentum $\vec{\mathbf{H}}_I$ of the rigid-body simulation applying a constant inertial torque $\vec{\boldsymbol{\tau}}_I$ using the Leapfrog time integration method and a time step size of $\Delta t = 0.25$ s

To assess the quality of the involved time integration schemes, figure 7.5 shows a plot of the maximum relative L_2 error of the inertial angular momentum for all involved time integration methods and time step sizes (asterisk symbols) along with its least-squares fitted curves (solid lines). It can be seen that the slopes of the straight lines (in log-log) fit very well to the theoretically expected convergence behavior of the numerical integration methods, i.e. a slope of $m = 1.03$ for the first-order Euler method, a slope of $m = 2.00$ for the second-order Leapfrog method and a slope of $m = 3.99$ for the fourth-order Runge-Kutta (RK4) method.

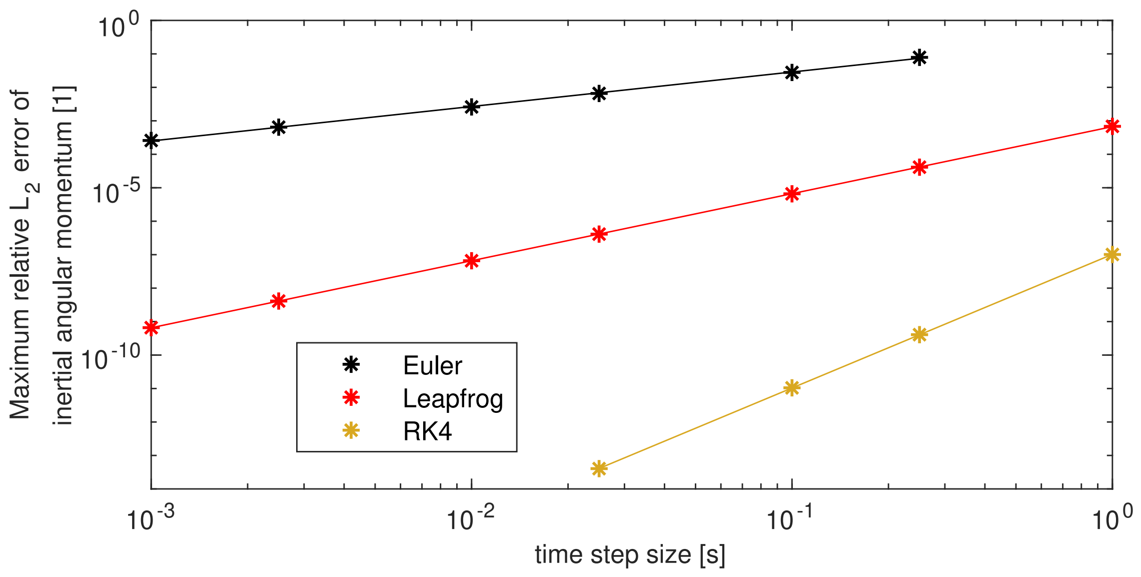


Figure 7.5: Spacecraft AOCS simulator: Maximum relative L_2 error of the inertial angular momentum for all involved time integration methods and time step sizes (asterisk symbols) along with its least-squares fitted curves (solid lines)

The plot also shows that for the RK4 method, no simulation is performed using a time step size smaller than 0.025 s. This is due to the fact that with a time step size of $\Delta t = 0.025$ s the RK4 method reaches a maximum relative L_2 error of the inertial angular momentum of 4.04e-14, which is already close to the floating-point accuracy of approx. 1e-15 for double precision numbers used in Matlab. Further reducing the time step size yields an error in the same order than the machine epsilon, thus the roundoff error becomes the driving error and no improvement in the maximum relative L_2 error can be achieved.

Summarizing it can be said that the spacecraft AOCS simulator successfully approximated the given rigid-body dynamics problem and showed the expected theoretical convergence behavior of the three implemented time integration schemes, the Euler method, the Leapfrog method as well as the Runge-Kutta (RK4) method.

Chapter 8

Coupling propellant sloshing and spacecraft attitude dynamics & control

8.1 Coupling strategy

In Attitude & Orbit Control System (AOCS) verification campaigns, it is desired to analyze the influence of sloshing propellant on the spacecraft and its AOCS. As such, the software presented in chapter 3 implementing the computational propellant sloshing model derived in chapter 2 needs to be coupled to the software presented in chapter 7 implementing the computational spacecraft attitude & orbit dynamics and control model derived in chapter 6.

A partially filled propellant tank is fixedly mounted to a spacecraft that is able to freely rotate and translate in 3D space. Following the approach presented in [23], the coupling of the propellant's and spacecraft's motion is then implicitly achieved by treating the solid tank wall particles (denoted by subscript s) from the propellant sloshing simulation as being part of the spacecraft's rigid body from the spacecraft AOCS simulation, yielding the tank wall particle positions $\vec{\mathbf{x}}_s$, velocities $\vec{\mathbf{u}}_s$ and accelerations $\vec{\mathbf{a}}_s$ in an inertial frame as

$$\vec{\mathbf{x}}_s = \vec{\mathbf{x}}_{s/c} + \mathrm{DCM}_{IB} \cdot \vec{\mathbf{x}}_s^B \tag{8.1.1}$$

$$\vec{\mathbf{u}}_s = \vec{\mathbf{u}}_{s/c} + \vec{\boldsymbol{\omega}}_B \times \vec{\mathbf{x}}_s^B \tag{8.1.2}$$

$$\vec{\mathbf{a}}_s = \vec{\mathbf{a}}_{s/c} + \dot{\vec{\boldsymbol{\omega}}}_B \times \vec{\mathbf{x}}_s^B + \vec{\boldsymbol{\omega}}_B \times \vec{\boldsymbol{\omega}}_B \times \vec{\mathbf{x}}_s^B \tag{8.1.3}$$

with $\vec{\mathbf{x}}_{s/c}$, $\vec{\mathbf{u}}_{s/c}$ and $\vec{\mathbf{a}}_{s/c}$ being the spacecraft's position, velocity and acceleration, respectively, in an inertial frame. The tank wall particle's position in the spacecraft body frame is given by $\vec{\mathbf{x}}_s^B$ and DCM_{IB} is the direction cosine matrix, transforming a given vector from the spacecraft body frame B into an inertial frame I. $\vec{\boldsymbol{\omega}}_B$ and $\dot{\vec{\boldsymbol{\omega}}}_B$ are the spacecraft's angular rate and angular acceleration.

Additionally, a dynamic equilibrium at the tank wall as interface of the propellant sloshing and spacecraft AOCS simulation is enforced. As explained in section 2.7, during the propellant sloshing simulations in this work, multiple layers of dummy particles are used for the discretization of the tank wall in order to complete the missing kernel support of the liquid particles close to the tank wall. Based on a force balance at this solid-liquid interface, the pressure of the wall particles is set in such a way that they mimic a continuous liquid phase.

In order to obtain the integrated force $\vec{\mathbf{F}}_{\text{int}}$ and torque $\vec{\mathbf{T}}_{\text{int}}$, with respect to the spacecraft center of mass, exerted by the propellant on the tank wall, first the force $\vec{\mathbf{F}}_{\alpha s}$ acting on a liquid particle α due to the surrounding wall particles s is calculated by

$$\vec{\mathbf{F}}_{\alpha s} = \sum_{s} -(V_\alpha^2 + V_s^2)\frac{\rho_s p_\alpha + \rho_\alpha p_s}{\rho_\alpha + \rho_s}\nabla_\alpha W_{\alpha s} \tag{8.1.4}$$

with V_α and V_s being the volumes, ρ_α and ρ_s the densities and p_α and p_s the pressures of the liquid particles α and solid wall particles s, respectively, and $\nabla_\alpha W_{\alpha s} = \frac{\partial}{\partial \vec{\mathbf{x}}_\alpha^B} W(|\vec{\mathbf{x}}_\alpha^B - \vec{\mathbf{x}}_s^B|, h)$ the gradient of the interpolation kernel. Please note that the sum in equation 8.1.4 is evaluated for all solid wall particles s.

Using Netwon's third law, the integrated force $\vec{\mathbf{F}}_{\text{int}}$ and torque $\vec{\mathbf{T}}_{\text{int}}$ exerted by the propellant on the tank wall can then be found by

$$\vec{\mathbf{F}}_{\text{int}} = -\sum_{\alpha} \vec{\mathbf{F}}_{\alpha s} \tag{8.1.5}$$

$$\vec{\mathbf{T}}_{\text{int}} = -\sum_{\alpha} \vec{\mathbf{x}}_\alpha^B \times \vec{\mathbf{F}}_{\alpha s} \tag{8.1.6}$$

with $\vec{\mathbf{x}}_\alpha^B$ being the position of liquid particle α in the spacecraft body frame. Please note that the sums in equations 8.1.5 and 8.1.6 are evaluated for all liquid particles α.

Finally, the dynamic equilibrium at the tank wall is achieved by considering in the spacecraft AOCS simulation the force $\vec{\mathbf{F}}_{\text{int}}$ when evaluating Euler's translational dynamic equation of motion 6.1.8 and the torque $\vec{\mathbf{T}}_{\text{int}}$ when evaluating Euler's rotational dynamic equation of motion 6.1.4.

8.2 Software implementation of the coupling

The remaining open requirements [R3] from section 3.1 and [RD] from section 7.1 are closed in the following by describing the software implementation of the coupling of propellant sloshing and spacecraft attitude dynamics & control.

As described in sections 3.2 and 7.2, the propellant sloshing simulator is written in C++ and the spacecraft AOCS simulator in object-oriented Matlab, thus an interface needs to be established to exchange the spacecraft's rotational and translational state with the integrated force and torque exerted by the propellant on the tank, as derived in the previous section 8.1.

Following [23], this is done by using Matlab's MEX API [60], which allows to compile the propellant sloshing simulator C++ code in such a way that it can be directly called from within Matlab. The propellant sloshing simulator is thus compiled and its functionality made available in a new Matlab class *SPHSloshingSimulator*, which takes care of the construction and destruction of the C++ SPH algorithm objects as well as of the time integration. As is visualized in figure 8.1, the spacecraft AOCS simulator is extended by another Matlab class *PropellantTank* inheriting from the *Hardware* class, which uses a *SPHSloshingSimulator* class instance for access to the propellant sloshing simulator functionality and implements the interface to the spacecraft AOCS simulator by trading rotational and translational spacecraft states for integrated forces and torques exerted by the propellant on the tank and thus on the spacecraft.

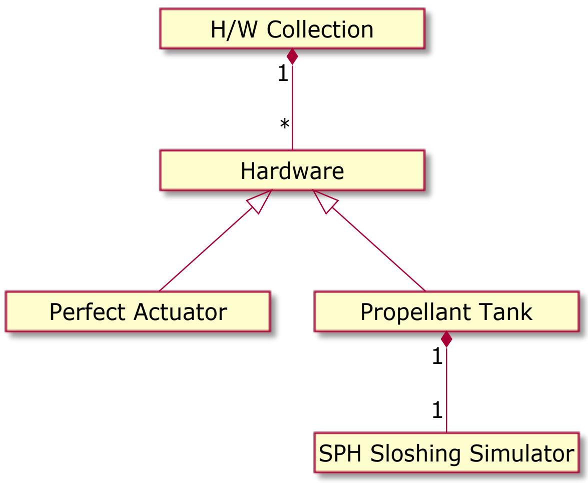


Figure 8.1: Coupling propellant sloshing simulator and spacecraft AOCS simulator: Overview of hardware related classes of the spacecraft AOCS simulation environment including their relationships, extending figure 7.3 by a hardware class *PropellantTank* that uses the previously presented computational model of a partially filled propellant tank through the class *SPHSloshingSimulator*.

The advantage of using Matlab's MEX API for the coupling is that the data exchange between the two C++ and Matlab simulators is not performed on a file basis, as e.g. done in [37], but in Random-access-memory (RAM), thus yielding faster response times. Additionally, by using Matlab's MEX API it is possible to fully control the propellant sloshing simulator from within Matlab, offering for example the possibility to use one unified time integration scheme for the simulation of both propellant sloshing and spacecraft attitude dynamics & control.

Finally, figure 8.2 shows the components including interfaces of the software implementation of the coupled simulation environment comprising propellant sloshing and spacecraft AOCS simulator. As described in sections 3.2 and 7.2 for the respective simulators, both the propellant sloshing simulator and spacecraft AOCS simulator take a configuration file as input, thereby defining the simulation completely. During the run of a coupled simulation, the two simulators are exchanging translational and rotational spacecraft states coming from the spacecraft AOCS simulator, $\vec{\mathbf{x}}_{s/c}, \vec{\mathbf{u}}_{s/c}, \vec{\mathbf{a}}_{s/c}, \mathbf{q}_{BI}, \vec{\boldsymbol{\omega}}_B, \dot{\vec{\boldsymbol{\omega}}}_B$, and integrated propellant force and torque acting on the tank coming from the propellant sloshing simulator, $\vec{\mathbf{F}}_{\text{int}}, \vec{\mathbf{T}}_{\text{int}}$. The results of both simulators are fed into the previously presented Matlab-based post-processing tool capable of visualizing, analyzing and exporting the respective propellant sloshing or spacecraft attitude dynamics & control data.

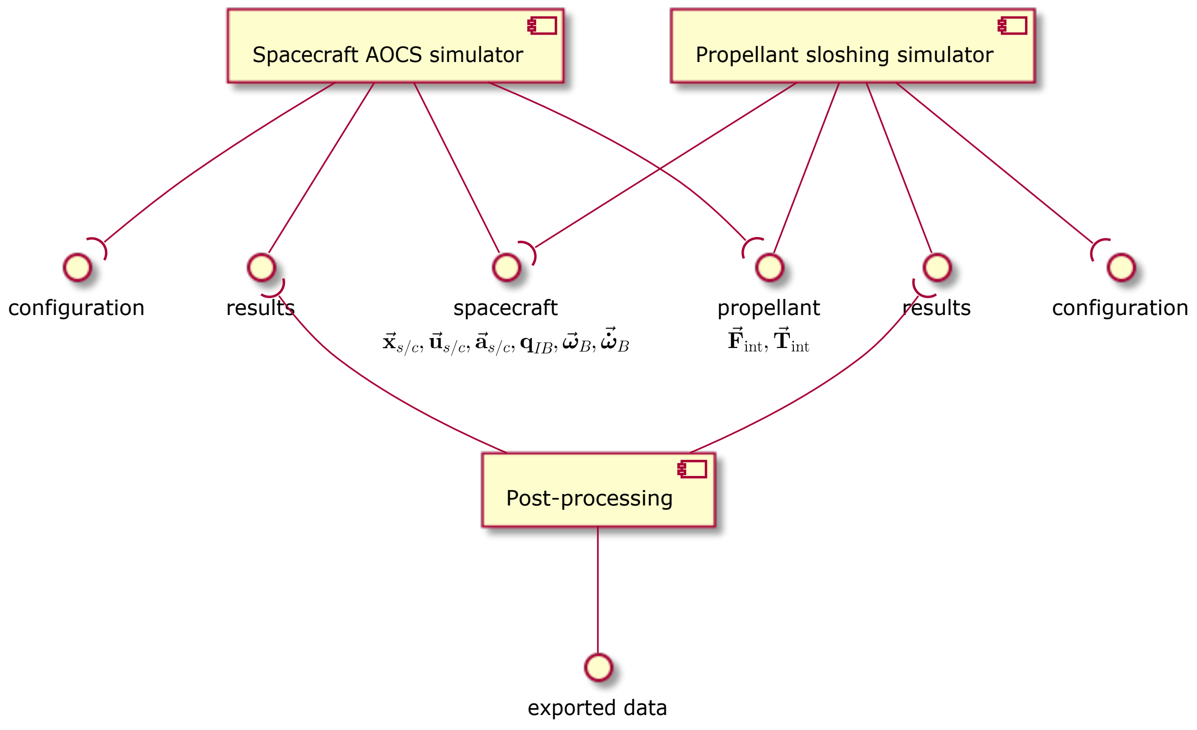

Spacecraft AOCS simulator
Propellant sloshing simulator
configuration
results
spacecraft
$\vec{\mathbf{x}}_{s/c}, \vec{\mathbf{u}}_{s/c}, \vec{\mathbf{a}}_{s/c}, \mathbf{q}_{IB}, \vec{\boldsymbol{\omega}}_B, \dot{\vec{\boldsymbol{\omega}}}_B$
propellant
$\vec{\mathbf{F}}_{\text{int}}, \vec{\mathbf{T}}_{\text{int}}$
results
configuration
Post-processing
exported data

Figure 8.2: Coupling propellant sloshing simulator and spacecraft AOCS simulator: Components of the software implementation including interfaces.

Please note that for the coupled simulations performed in chapter 9, the Leapfrog integration scheme presented in sections 2.6 and 6.2 is chosen as unified numerical time integration method for both the propellant sloshing simulator and the spacecraft AOCS simulator. This guarantees that the rotational and translational spacecraft states as well as the integrated propellant forces and torques are evaluated by the respective other simulator at the proper time instances, thus providing a consistent numerical time integration scheme.

Chapter 9

Application to AOCS verification of nonlinear propellant sloshing

9.1 Forced roll-motion in high-gravity conditions

9.1.1 Experimental & numerical setup

After having validated the computational propellant sloshing model and its software implementation in chapter 4 for high-gravity environments and in chapter 5 for zero-gravity environments, the coupled propellant sloshing and spacecraft attitude dynamics & control environment as described in chapter 8 is now used for a first application in high-gravity conditions, i.e. the simulation of a nonlinear forced roll-motion of a partially filled rectangular tank in Earth's gravity field. Please note that a simulation of the same experiment using a different computational sloshing model, i.e. a less advanced predecessor of the computational model presented in chapter 2, has been performed in [23].

In the following, the "Sloshing wave impact problem", i.e. test case 10 [12, 54, 53, 7] from the SPHERIC[1] community is simulated and the results are compared to the experiment. This experiment comprises a lateral sloshing wave impact in a partially filled rectangular tank of width W = 900 mm, depth D = 62 mm and height H = 508 mm. The tank is filled up to a height of h = 93 mm with colored water of density ρ = 998 kg/m^3 at 19°C and a dynamic viscosity of μ = 8.94e-4 Pa·s, yielding a filling ratio of 18 %. At the left tank wall at a height of 93 mm, a pressure sensor is located that measures the pressure of the water wave during sloshing.

Figure 9.1 shows the basic experimental setup and the initial state of the system, i.e. the tank being in a horizontal position and the water at rest. Since the experiment is performed on Earth, a vertical gravitational acceleration of $g = -9.81$ m/s^2 is acting on the system. At time $t = 0$ s, a forced roll motion is put in place about the center of the bottom line of the tank with a rotation frequency of $f \approx 0.6$ Hz. During the experiment, the pressure sensor values as well as the roll angles are recorded.

Regarding the numerical setup, a particle diameter of 0.0155 m is used for the spatial discretization, resulting in 1,392 fluid particles. Please note that every numerical simulation performed in this work is done in 3D, also the simulation presented in the following where only 1,392 particles are used in order to fully discretize the three-dimensional fluid volume inside the tank (see also section 4.4). As can be seen in figure 9.2, the simulation particles are geometrically distributed

[1]SPHERIC is the international organization representing the community of researchers and industrial users of Smoothed Particle Hydrodynamics.

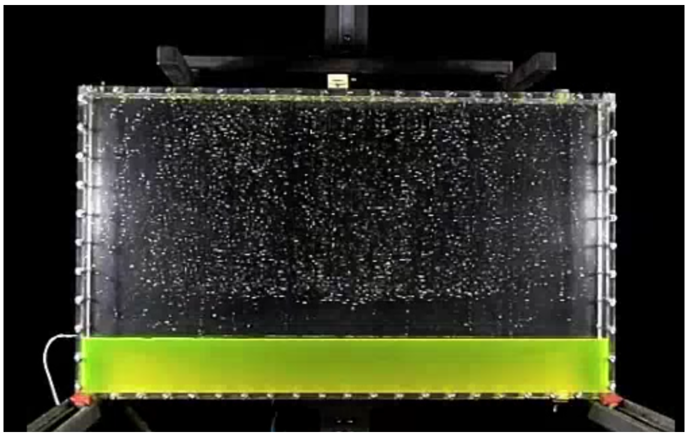


Figure 9.1: Forced roll-motion in high-gravity: Experimental setup [12, 54, 53, 7]

using a Cartesian grid around the center of the tank with dimensions of the tank plus 2 layers of wall particles around in order to provide full kernel support for fluid particles close to the tank wall. The time step size is set to 2 ms and the maximum density variation of the fluid to 1 %. A cubic spline kernel as described in section 2.2 is used with a compact support chosen such that the kernel comprises approximately 30 particles. The spacecraft AOCS simulator is run in feedforward mode, i.e. without a controller in the loop, and the roll angles obtained from the experiment as well as the derived angular rates and angular accelerations are directly applied to the rigid tank structure consisting of the wall particles.

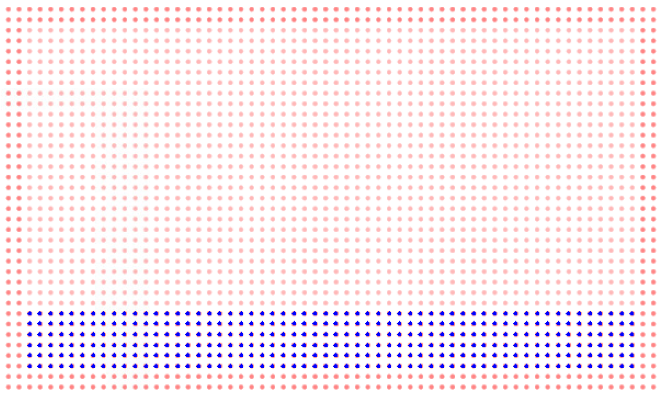


Figure 9.2: Forced roll-motion in high-gravity: Numerical setup: z/x axis view of a cut through tank and fluid at y = 0 m

9.1.2 Simulation of a forced roll-motion

A simulation of the "Sloshing wave impact problem", i.e. SPHERIC test case 10, is performed using the setup as described in the previous section. Figure 9.3 shows the simulated and experimentally measured pressure values at the sensor location at the left tank wall.

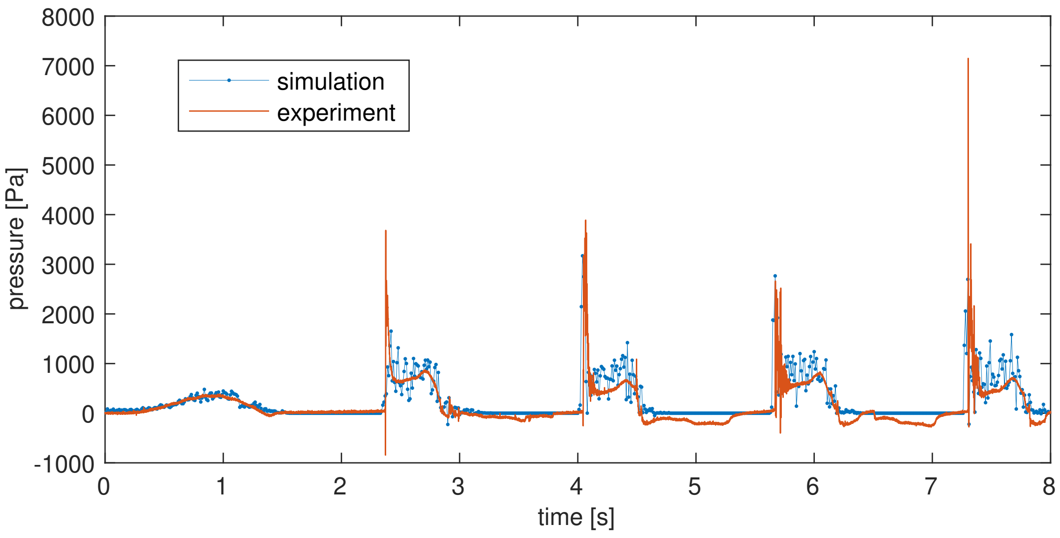


Figure 9.3: Forced roll-motion in high-gravity: Simulated (blue) vs. experimental (red) pressure values at the location of the sensor at the left tank wall

Although high-frequency oscillations are present in the computationally obtained pressure values due to the chosen weakly-compressible numerical approach (see also section 9.2.3), it can be seen that the pressure time series coming from the experiment can be reproduced quite well. The impacts of the sloshing water hitting the tank wall and thus the pressure sensor are well determined in time, whereas the magnitudes of these pressure peaks fit only qualitatively with those coming from simulation.

As discussed in [12, 48], the random character of the pressure peaks during an impact makes these maximum pressure values unsuitable for comparison of experimental and numerical results. Instead, the pressure impulse is used, which integrates the pressure over the time span of the impact from t_0 to t_1 as

$$P(\vec{\mathbf{x}}) = \int_{t_0}^{t_1} p(\vec{\mathbf{x}}, t)\ \mathrm{dt} \tag{9.1.1}$$

with $p(\vec{\mathbf{x}}, t)$ being the pressure at location $\vec{\mathbf{x}}$ and time t.

A comparison of the experimentally and computationally determined pressure impulses for the four impacts as shown in figure 9.3 is presented in table 9.1. Here it can be seen that the pressure impulses of the four impacts calculated from numerical results have a relative error ranging from +2 % up to +40 % compared to the experiment. Hence the computational model tends to overestimate the pressure during a sloshing impact.

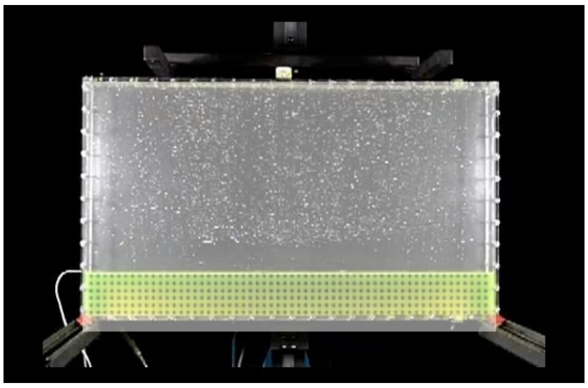
Table 9.1: Forced roll-motion in high-gravity: Comparison of the experimentally and computationally determined pressure impulses at the sensor location for the four sloshing impacts

	1st impact	2nd impact	3rd impact	4th impact
P_{exp} [Pa·s]	363.7	310.8	326.7	334.5
P_{sim} [Pa·s]	371.0	433.9	448.9	431.1
Rel. error [%]	2.0	39.6	37.4	28.9

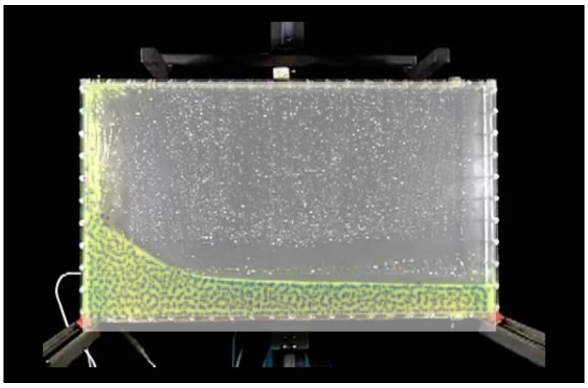
In order to further assess the quality of the computational model, a 3D visualization of the numerical results is created and overlaid to a video captured during the experiment, showing the roll-motion of the tank structure with colored water sloshing inside. Photos of the result can be seen in figure 9.4, where the sloshing motion is depicted for 6 different times instances. The particles from the numerical simulation are colored in blue, the water from the experiment is colored in green.

The first picture shows the starting point at $t = 0$ s, where the overlay of the simulation particles and the colored water can be seen in their initial configuration. The second picture at $t = 2.54$ s is taken at the point in time where the first sloshing impact on the left tank wall occurs. The bulk motion of the water in the entire tank is reproduced very well by the simulation, whereas the splashes at the left tank wall are not properly captured by the simulation particles. This is also expected due to the fact that the resolution of this 3D simulation with 1,392 fluid particles is too rough to be able to account for these fine-grained structures. The third picture at $t = 2.95$ s shows the sloshing wave moving in direction of the right tank wall after the first impact. Here the particles do not reproduce well the crest of the breaking wave, which is again mainly due to the low resolution. The remainder of the water shape however is again fitting in an acceptable way. At $t = 3.39$ s, the fourth picture depicts the sloshing impact on the right tank wall. As before, the bulk motion of the water is reproduced very well and the splashes at the right top are not reflected. The fifth picture at $t = 3.86$ s shows the sloshing wave on its way to the left tank wall. Here the entire water shape is replicated quite well including the crest of the breaking wave. The sixth picture at $t = 4.15$ s finally depicts the second sloshing impact on the left tank wall. Also here, excluding again the splashes at the top left, the simulation particles recreate the water motion pretty well.

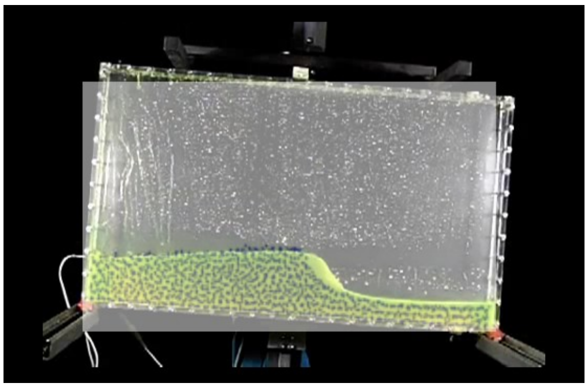
Summarizing it can be noted that the bulk motion of the sloshing water is reproduced very well throughout the entire simulation time. The fact that certain details like fine-grained splashes during the sloshing impacts cannot be resolved properly is just of minor importance, since the involved mass of these water splashes is negligible compared to the mass of the water bulk hitting the tank wall. The computational propellant sloshing model at hand is developed to be used within AOCS verification simulations, hence the calculation of forces exerted by the fluid on the tank walls and thus on the spacecraft is the main goal. This means that the low mass of the fluid splashes and accordingly its small force acting on the spacecraft can be neglected without loss of information, thus allowing to run the simulations with low resolutions yielding faster simulation runtimes.



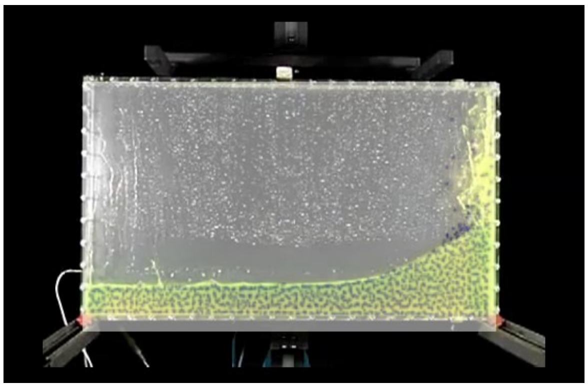
(a) $t = 0$ s



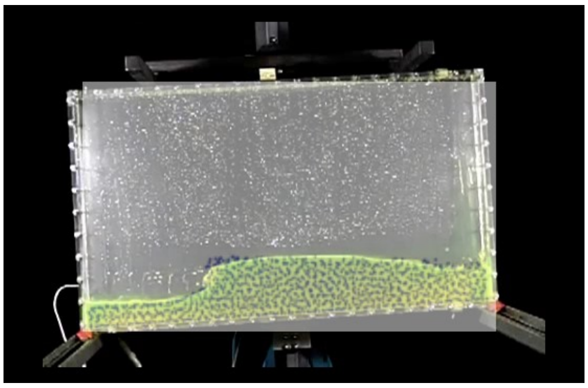
(b) $t = 2.54$ s



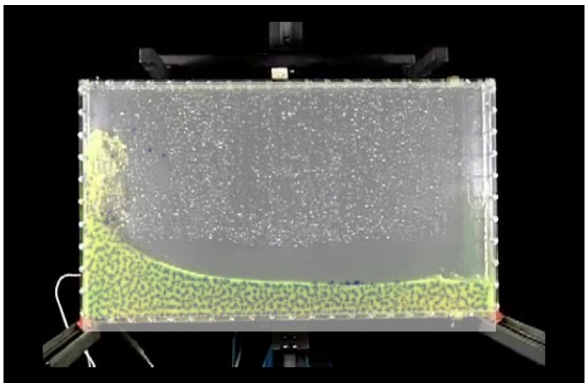
(c) $t = 2.95$ s



(d) $t = 3.39$ s



(e) $t = 3.86$ s



(f) $t = 4.15$ s

Figure 9.4: Forced roll-motion in high-gravity: Time series of screenshots from the overlay of the sloshing motion from experiment [12, 54, 53, 7] (water in green) and a visualization of the simulation results (particles in blue).

9.2 Earth observation satellite mission scenario in zero-gravity conditions

9.2.1 Mission scenario description

In the following, a typical Earth observation satellite mission scenario [22] in zero-gravity conditions is simulated. As recent Earth observation satellites have their optical instrument fixed to the body, in order to take images from Earth the entire satellite needs to be rotated accordingly. A typical satellite of this kind is on a low Earth orbit passing above or near the Earth poles with an attitude such that the optical instrument is Nadir-pointing. At its current position in orbit, in order to take images from a location on Earth which is not directly below the satellite, it needs to perform a slew maneuver for targeting the desired area. After the satellite is stabilized on its new attitude, one or multiple images are taken. Afterwards, the satellite usually performs a second slew maneuver in order to return to its initial attitude, i.e. Nadir-pointing.

The exemplary Earth observation satellite mission scenario chosen for this work is performing such an off-nadir image scan. First, the satellite is slewing 30 deg around its y-axis within a 5 s time span using a bang-bang maneuver. In the next 10 s, the satellite is stabilizing and taking the images from the desired location. Afterwards, a second bang-bang slew maneuver is performed by 30 deg in 5 s around the negative y-axis in order to bring the satellite back in its initial Nadir-pointing attitude. The resulting slew rates for the two maneuvers between 0 s and 5 s as well as between 15 s and 20 s can be seen in figure 9.5.

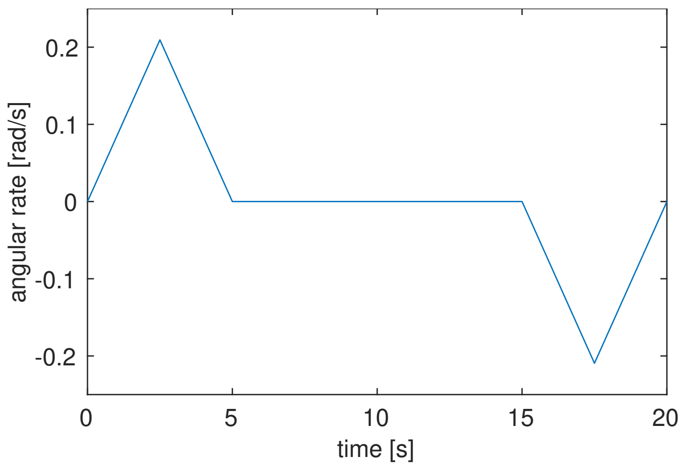


Figure 9.5: Earth observation satellite mission scenario: Angular rate profile of the two successive bang-bang slew maneuvers around the y-axis

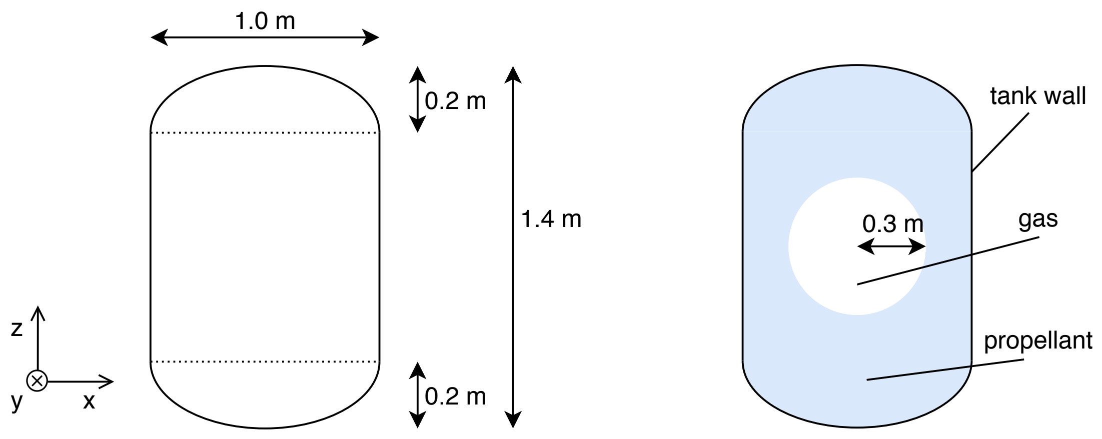
The chosen satellite dry mass, i.e. including tank structure but without propellant, is $m_{s/c} = 3000$ kg. The moment of inertia of the satellite is given by

$$\mathbf{I}_{s/c} = \begin{pmatrix} 1000 & 0 & 0 \\ 0 & 1000 & 0 \\ 0 & 0 & 1000 \end{pmatrix} \text{kg m}^2 \tag{9.2.1}$$

The initial attitude quaternion from inertial frame to body frame $\mathbf{q}_{BI}$ as well as the initial angular rate $\vec{\boldsymbol{\omega}}_B$ are selected to be

$$\mathbf{q}_{BI} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \qquad \vec{\boldsymbol{\omega}}_B = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \text{ rad/s} \tag{9.2.2}$$

The satellite carries as a cylindrical tank with spherical domes as can be seen in figure 9.6(a). The tank has a width of 1.0 m, a height of 1.4 m and a depth of 1.0 m, and the height of the spherical caps is given by 0.2 m, yielding a total tank volume of 0.95 m^3.



(a) Cylindrical tank with spherical domes (b) Initial fluid distribution in tank

Figure 9.6: Earth observation satellite mission scenario: Experimental setup

The tank is filled with a propellant and in the center of the tank there is a gas bubble in form of a sphere with radius 0.3 m as depicted in figure 9.6(b). The propellant has a density of $\rho = 1000.0$ kg/m^3, a dynamic viscosity of $\mu = 0.01$ Pa·s, a surface tension coefficient regarding the propellant-gas interface of $\gamma = 70.0\text{e-}3$ N/m as well as a static contact angle regarding the propellant-gas-tank interface of $\theta_0 = 60.0$ deg. The fluid volume amounts to 0.84 m^3, thus giving a filling ratio of 88 % and a total fluid mass of 840 kg.

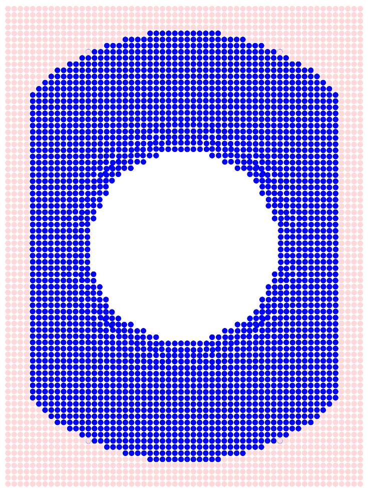
As is obvious from the experimental setup, i.e. a rotational attitude maneuver around the center using a cylindrical tank with spherical domes and the fluid being filled to 88 % with a gas bubble in the tank center comprising the remaining volume, it is not expected that the zero-gravity propellant-gas interface will be destabilized, but only small deformations of the gas bubble are foreseen. Even though no big distortions are induced like during a high-velocity propellant impact, the error on the desired attitude due to the varying mass distribution of the propellant during the slew maneuvers is already significant. Since especially for an Earth observation satellite mission the time needed to stabilize the satellite with its propellant cannot be used for its main purpose, i.e. image takes, getting insights into the propellant dynamics and its impact on the AOCS may help to reduce this stabilization time and thus allows to take more images per orbit.

9.2.2 Numerical setup

In the remainder of this chapter, the Earth observation satellite mission scenario as described in section 9.2.1 is simulated using the coupled propellant sloshing and spacecraft attitude dynamics & control environment from chapter 8 and the numerical setup as follows.

A particle diameter of 0.02 m is used for the spatial discretization, resulting in 105,104 fluid particles. The time step size is set to 0.5 ms and the maximum density variation of the fluid to 0.1%. A Wendland C^2 kernel as described in section 2.2 is used with a compact support chosen such that the kernel comprises approximately 230 particles.

The simulation particles are geometrically distributed using a Cartesian grid around the origin with dimensions of the tank plus 4 layers of wall particles around in order to provide full kernel support for fluid particles close to the tank wall. Inside the tank, the fluid particles are placed on a vertex of the Cartesian grid, if the distance of the vertex to the origin is greater or equal than the inner radius of the spherical fluid shell as described in 9.2.1. This yields a rather unphysical distribution of fluid particles at the free-surface with edges and a non-smooth shape as can be seen in figure 9.7(a). Hence the simulation is run for 20 s in zero-gravity conditions without any external disturbances in order to let the surface tension regularize the fluid particles at the propellant-gas interface. Figure 9.7(b) shows the resulting particle distribution at $t = 20$ s, where the fluid free-surface is not yet converged to its spherical minimum area, but already shows a smooth and regular contour as desired.


(a) Initial geometrical particle distribution

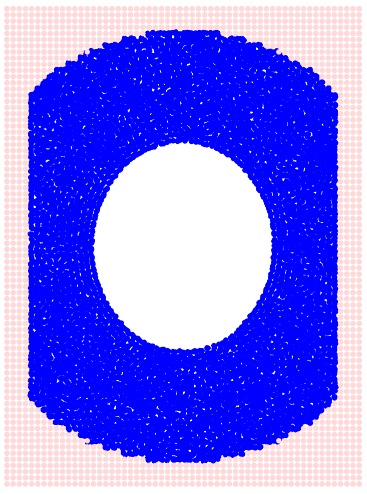

(b) Smooth fluid free-surface at $t = 20$ s

Figure 9.7: Earth observation satellite mission scenario: Setting up an initial particle distribution. The figures are showing slices of thickness $\Delta y = [-0.04, +0.04]$ m (i.e. a layer of 4 particles in y-direction) through tank and fluid

9.2.3 Artificial sound wave analysis

As explained in chapter 2, the computational propellant sloshing model at hand uses a weakly compressible approach in order to approximate the propellant dynamics. This weakly compressible approach implies artificial sound wave oscillations, which may influence the Attitude and Orbit Control System (AOCS) of a satellite. In the following, the sound wave oscillations generated by the proposed computational propellant sloshing model are analyzed and the impact on the AOCS is discussed.

The weakly-compressible approach incorporated for simulating fluids using the Smoothed Particle Hydrodynamics method allows the fluid a certain degree of compressibility κ in order to use a simple state equation to close the equations of motion (see chapter 2), resulting in a simpler set of equations to solve through an explicit algorithm, which can be easily parallelized [10].

This approach comes with the cost of sound wave oscillations, since the allowed artificial compressibility of the fluid causes fluctuations in the density and thus pressure field. These density and thus pressure field fluctuations then cause volumetric deformations leading to sound waves that form an oscillatory system comprising a spectrum of frequencies.

The generated spurious sound waves are characterized by the speed of sound, which is in the simulations performed in this work set as (see chapter 2)

$$c_s = \sqrt{\frac{\max(p_{\mathrm{b,max}}, p_{\mathrm{m,max}})}{\rho_0 \cdot \kappa}} \tag{9.2.3}$$

with $p_{\mathrm{b,max}}$ and $p_{\mathrm{m,max}}$ being the maximum expected pressure due to body forces and molecular forces, respectively, ρ_0 the density of the fluid and κ the compressibility factor used for the weakly-compressible approach.

An AOCS is typically equipped with a low-pass filter that removes frequencies above a certain cutoff frequency. Hence determining the lowest frequency that the artificial sound waves produce and ensuring that this frequency is way above the cutoff frequency is of main interest for ensuring that an AOCS is not affected by this artificial disturbance. When taking a look at the geometry of the problem at hand with a gas bubble in the middle of a cylindrical tank with spherical domes filled with propellant (see figure 9.7), it is clear that reflections of the sound waves may occur on different paths with different traveling lengths. Since for an AOCS the lowest frequency is most relevant, in the following an estimate is being done on the minimum frequency present in the system due to sound wave oscillations based on the maximum traveling length available inside the tank. Hence for the x/y-direction 1 m is assumed and for the z-direction 1.4 m (see dimensions of the tank in section 9.2.1) and the frequencies for these paths are estimated.

In general, a sound wave propagating inside a tank from one end to the other and back including reflection at the tank wall takes a time of $T = {}^{2l}/_{c_s}$ with l being the dimension of the tank and c_s the speed of sound. Considering this fundamental mode $n = 1$ as well as higher modes $n \in \mathbb{N}, n > 1$ yields the series of eigenfrequencies f_n for the sound waves as [16]

$$f_n = \frac{n \cdot c_s}{2l} \tag{9.2.4}$$

Looking at three different compressibilites $\kappa = \{0.1\%, 0.2\%, 0.4\%\}$, table 9.2 finally shows the estimated lowest frequencies, i.e. fundamental frequencies of the sound waves inside the tank for the x/y-direction, $f_1^{\mathrm{x,y}}$, and z-direction, f_1^{z}.

Table 9.2: Earth observation satellite mission scenario: Sound wave analysis: Estimated fundamental frequencies of sound wave oscillations to be expected in the numerical system due to the weakly compressible numerical approach

κ [%]	$f_1^{x,y}$ [Hz]	f_1^{z} [Hz]
0.1	4.33	3.09
0.2	3.06	2.19
0.4	2.17	1.55

From a theoretical point of view it can be concluded that for compressibilities of $\kappa = \{0.1\%, 0.2\%, 0.4\%\}$, the minimum frequency to be expected in the system due to traveling sound waves is given by the z-direction tank dimension as 3.09 Hz, 2.19 Hz and 1.55 Hz, respectively.

Since the geometry of the tank analyzed in this work is cylindrical with spherical domes and there is a gas bubble in the middle of the tank with the fluid being located at the tank wall, i.e. that the traveling distances are usually lower than the estimated upper bound, the majority of energy of the sound wave oscillations should be distributed among multiple frequencies and in higher regions.

In the following, the numerical setup as described above is used in order to perform three feed-forward simulations of 20 s in zero-gravity conditions without any external disturbances, varying the compressibility between $\kappa = \{0.1\%, 0.2\%, 0.4\%\}$. In those 20 s, the surface tension will regularize the fluid particles at the free-surface due to the initial unphysical geometric particle distribution as described above. Figure 9.8 shows the accelerations in z-direction during $t = 10$ s and $t = 15$ s as well as their corresponding Power Spectral Density plots for compressibilities of $\kappa = \{0.1\%, 0.2\%, 0.4\%\}$. The single-sided Power Spectral Densities in this work are calculated using a digital Fourier transform based on a Fast Fourier Transform algorithm as described in [49].

As expected, a spectrum of modes can be seen in the plots. When looking at the minimum frequency of sound wave oscillations available in z-direction, i.e. the dimension with the greatest possible traveling distance, one observes for a compressibility of 0.1 % major modes present from approx. 4.2 Hz onwards as well as a minor mode at 3.6 Hz, all above the theoretically estimated minimum frequency of $f_1^z = 3.09$ Hz. For a compressibility of 0.2 % there are multiple major modes at approx. 2.6 Hz, 3.2 Hz, 3.6 Hz and 4.4 Hz, which are also above the theoretically estimated minimum frequency of $f_1^z = 2.19$ Hz. The same applies for a compressibility of 0.4 % with major modes being present at 2.2 Hz, 2.6 Hz and 3 Hz as well as a minor mode at 1.2 Hz. Despite the minor mode, which can be neglected taking into account the total energy of the other modes, the theoretically estimated minimum frequency of $f_1^z = 1.55$ Hz for that case is also a proper lower border for occurring sound wave oscillations.

Since the sound wave oscillations discussed here are of artificial nature due to the weakly-compressible numerical approach, it is important to assure that they don't interfere with the AOCS in order to provide a realistic simulation of the satellite's behavior. Following the sampling theorem by Whittaker, Nyquist, Kotelnikov & Shannon [35], every signal that has a limited bandwidth of f_N, called Nyquist frequency, can be perfectly reconstructed using a sampling frequency of $f_s = 2f_N$. A typical modern AOCS is based on discrete time or digital control,

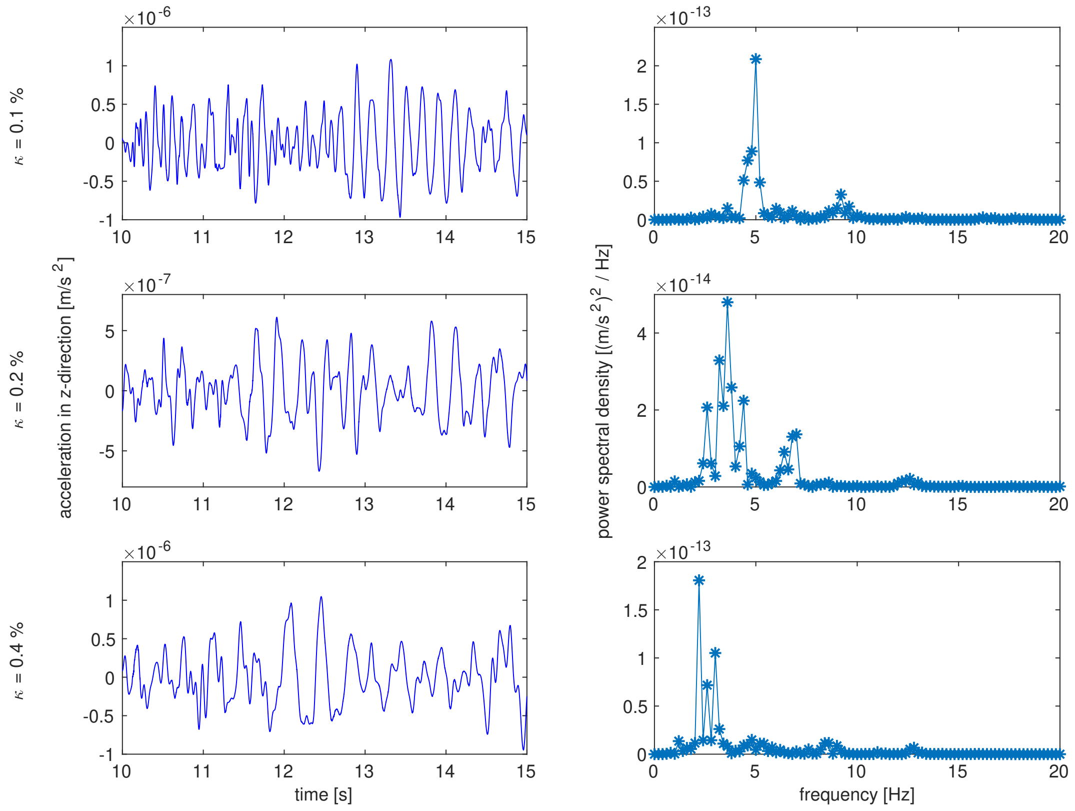


Figure 9.8: Earth observation satellite mission scenario: Sound wave analysis: Accelerations in z-direction for timespan $t = 10$ s to $t = 15$ s [left column] and their corresponding Power Spectral Densities [right column] for the compressibilities $\kappa = 0.1\%$ [top row], $\kappa = 0.2\%$ [middle row] and $\kappa = 0.4\%$ [bottom row]

so by assuming an AOCS sampling rate of f_s, this implies an inherent robustness against noise signals with a frequency higher than the Nyquist frequency f_N [59]. In order to minimize the effect of the numerically induced sound wave oscillations on the AOCS, it thus needs to be assured that the minimum frequency of the artificial sound waves is significantly higher than the AOCS Nyquist frequency f_N. Using the previously derived fundamental frequency of sound wave oscillations in the system, f_1, it is stipulated

$$f_1 \gg f_N \tag{9.2.5}$$

Using the relation in equation 9.2.4 as well as the sampling rate $f_s = 2f_N$ yields

$$\frac{c_s}{l_{\max}} \gg f_s \tag{9.2.6}$$

with $l_{\max}$ being the longest dimension of the tank and c_s the speed of sound. Inserting equation 9.2.3 finally gives the relation between the compressibility κ and the sampling rate of the AOCS f_s as well as other system parameters as

$$\kappa \ll \frac{\max(p_{\mathrm{b,max}}, p_{\mathrm{m,max}})}{\rho_0 \cdot f_s^2 \cdot l_{\max}^2} \tag{9.2.7}$$

It can thus be concluded that for a coupled sloshing and rigid body dynamics simulation using the numerical method introduced in this work, for a given fluid, tank geometry, attitude maneuver and AOCS sampling frequency, a compressibility κ needs to be chosen according to equation 9.2.7 as prerequisite for a simulation where the AOCS is not being affected by the artificial sound wave oscillations.

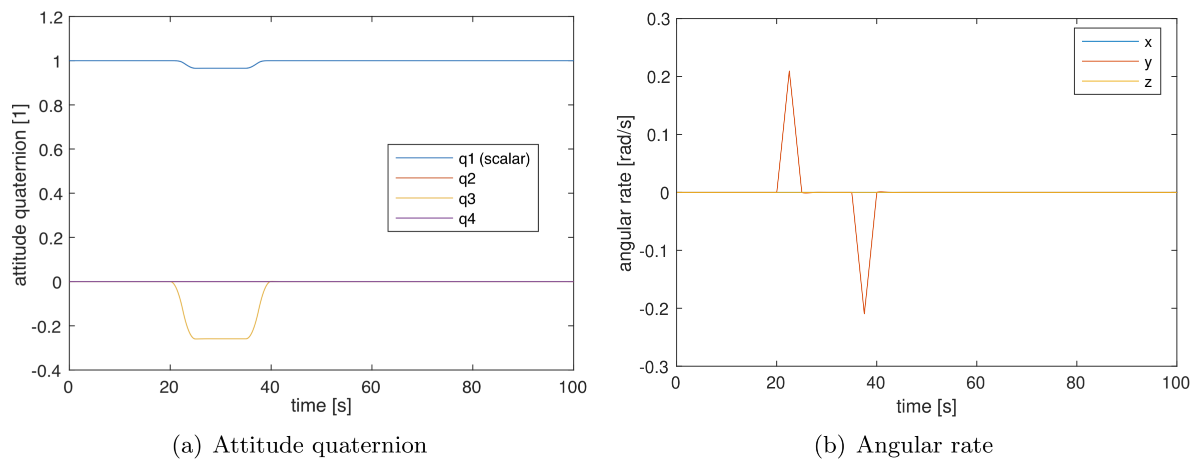
Choosing a lower compressibility implies a higher speed of sound and hence increases the minimum frequency of the sound wave oscillations in the system, thus having a smaller influence on the AOCS. On the other hand, a lower compressibility and thus a higher speed of sound also implies a lower simulation time step size needed as given through the CFL condition (see chapter 2). This leads to an increased time needed to perform the simulation. Selecting a proper value for the compressibility is thus a trade-off between numerical stability and simulation runtime.

A typical AOCS has a sampling frequency of a few Hz [69, 14], for example CryoSat-1 with a sampling rate of $f_s = 1$ Hz, what would yield for the simulated test case at hand a maximum compressibility of $\kappa_{\text{max}} = 3.8$ % due to equation 9.2.7. Usually, attitude and orbit control systems come in combination with a low-pass filter, suppressing input signals above the desired cutoff frequency before sampling and thus avoiding aliasing effects. In that case, the condition in inequation 9.2.7 can be relaxed by considering this cutoff frequency instead of the sampling frequency.

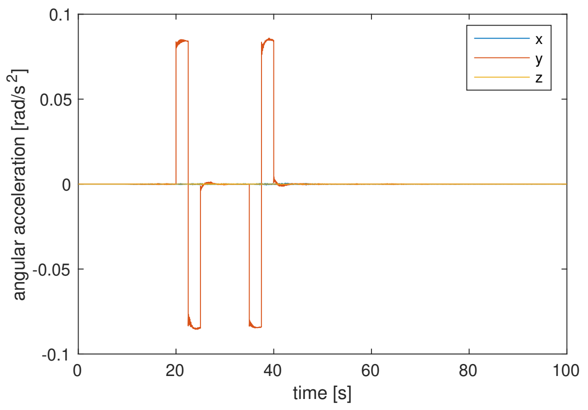
A further relaxation of inequation 9.2.7 can be performed by using the AOCS control bandwidth instead of the AOCS sampling frequency. For the computational simulations performed in the remainder of this chapter, the controller bandwidth of the AOCS is set to $\omega_n = 1$ rad/s as described in section 6.3, meaning that the reactions of the controller are limited to a frequency of up to $f_{\text{bw}} \approx 0.2$ Hz. Using this frequency f_{bw} with inequation 9.2.7 yields a theoretical maximum compressibility of $\kappa_{\text{max}} = 95$ %, which is trivially always fulfilled for any realistic SPH simulation involving liquid propellant. Summarizing it can be said that due to a high sampling frequency the AOCS may sense the distortions of the artificial sound waves caused by the weakly compressible numerical approach, but due to the low AOCS control bandwidth it is not able to react to it by use of the actuators.

9.2.4 Simulation of an Earth observation satellite mission scenario

After having set up an initial particle distribution as described in section 9.2.2, starting at $t = 20$ s the maneuvers of the Earth observation satellite mission scenario are now executed as described in section 9.2.1 using the feedback controller as presented in section 6.3. Figure 9.9 shows the resulting attitude quaternion, the angular rate as well as the angular acceleration for the simulation run. Here the maneuvers taking place at $t = 20$ s and $t = 35$ s with the resulting change in angular acceleration, rate and attitude can be clearly identified.



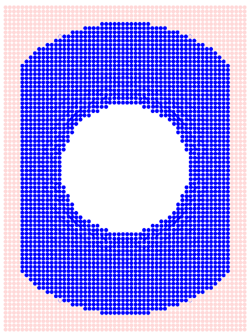
(a) Attitude quaternion (b) Angular rate



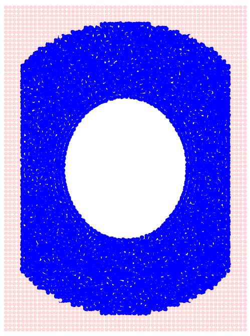
(c) Angular acceleration

Figure 9.9: Earth observation satellite mission scenario: Simulation results

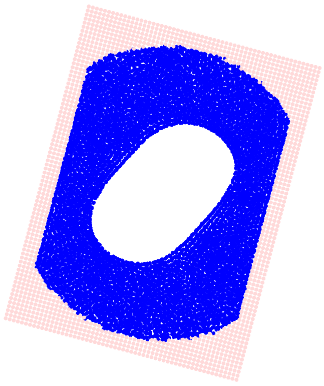
In order to visualize the fluid dynamics inside the tank during the slew maneuvers, figure 9.10 shows an image series of slices of thickness $\Delta y = [-0.04, +0.04]$ m (i.e. a layer of 4 particles in y-direction) through tank and fluid. Starting from the smooth particle distribution at $t = 20$ s (figure 9.10(b)), the first slew maneuver of 30° around the y-axis starts. The gas bubble gets elongated (figure 9.10(c)) and at the end of the slew maneuver, when the tank movement is stopped (figure 9.10(d)), the fluid continues its rotation and deforms the gas bubble into a cross-like form (figure 9.10(e)). Since the surface tension drives the free-surface to a minimal state, the gas bubble starts turning into a more sphere-like shape (figure 9.10(f)). Then, at $t = 35$ s, the second slew maneuver in opposite direction starts, elongating the gas bubble again (figures 9.10(g) and 9.10(h)). After the second slew maneuver is complete at $t = 40$ s and the tank again in its initial attitude, the fluid continues its movement, deforming the gas bubble again into a cross-like form (figure 9.10(i)). Then the surface tension again drives the free-surface into a minimal state, resulting in an almost spherical shape at $t = 100$ s (figure 9.10(l)).



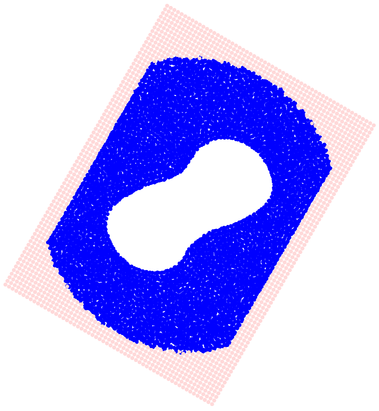
(a) $t = 0.1$ s



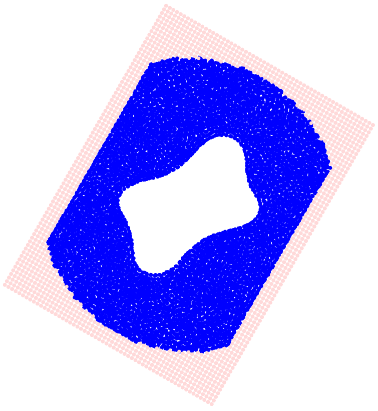
(b) $t = 20.0$ s



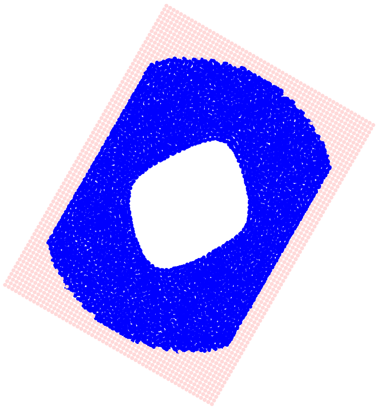
(c) $t = 22.5$ s



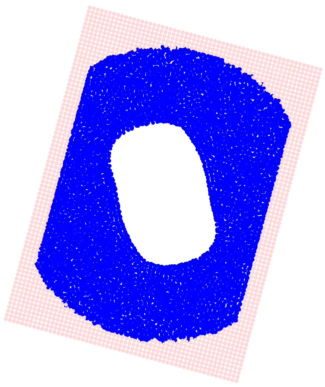
(d) $t = 25.0$ s



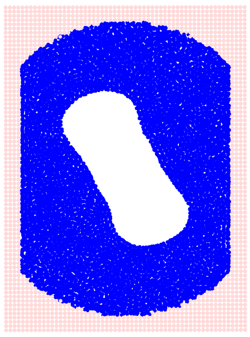
(e) $t = 30.0$ s



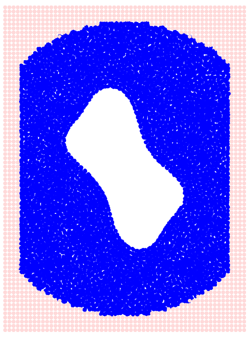
(f) $t = 35.0$ s



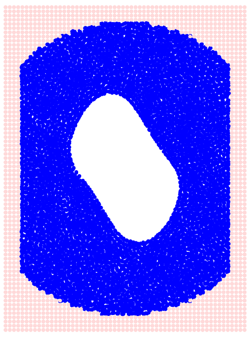
(g) $t = 37.5$ s



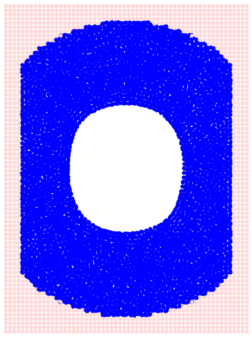
(h) $t = 40.0$ s



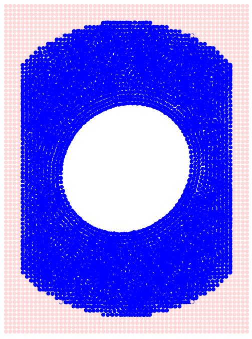
(i) $t = 45.0$ s



(j) $t = 50.0$ s



(k) $t = 60.0$ s



(l) $t = 100.0$ s

Figure 9.10: Earth observation satellite mission scenario: Image series showing slices of thickness $\Delta y = [-0.04, +0.04]$ m (i.e. a layer of 4 particles in y-direction) through tank and fluid

Figure 9.11 top shows the total torque (in blue) in y-direction, zoomed into a region between 40 and 60 s, i.e. the time directly after the second maneuver is completed. The total torque is the sum of the control torque produced by the actuators as well as the torque induced by the sloshing propellant. It can be seen that the total torque signal has a high-frequency part overlaid and is converging to zero. When looking at the power spectral density of the total torque (in blue) in figure 9.11 bottom, one can see a major mode at $\approx$ 0.2 Hz and two minor modes at $\approx$ 4.6 Hz as well as around $\approx$ 6.6 Hz.

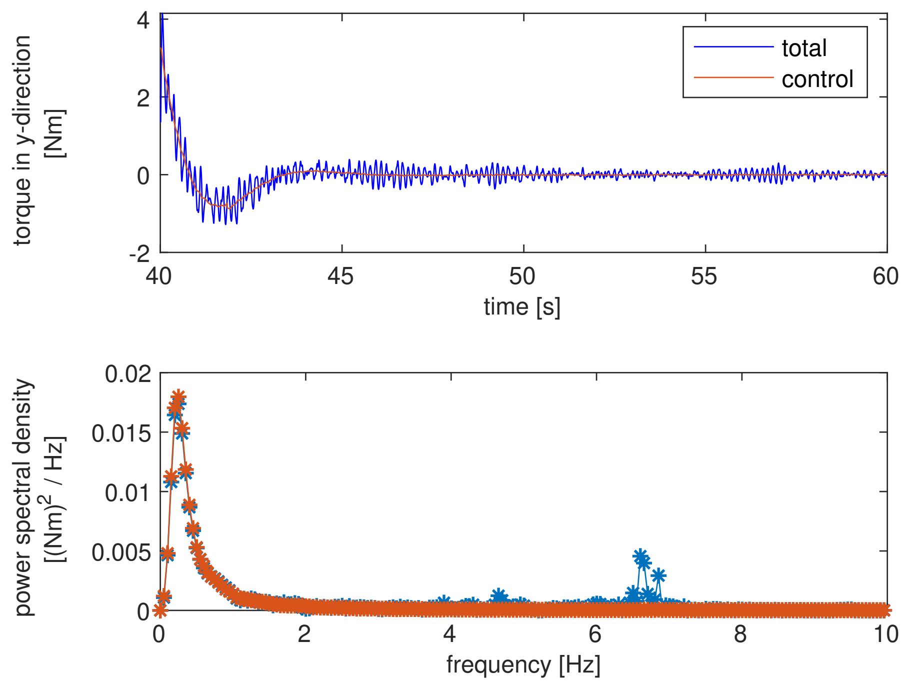


Figure 9.11: Earth observation satellite mission scenario: Total torque (in blue) as well as control torque (in red) in y-direction, zoomed into a region between 40 and 60 s [top] as well as their corresponding power spectral densities [bottom]

Due to the nature of the experimental setup, i.e. a rotational attitude maneuver around the origin using a cylindrical tank with spherical domes and the fluid being settled at the tank wall with a gas bubble in the middle (see figure 9.7), no propellant-gas interface destabilization is expected but rather a deformation of the gas bubble in the middle and thus a changing fluid mass distribution around it, as can be seen also in figure 9.10. This means that no bigger sloshing modes are anticipated in the power spectral density plot and that the major mode around 0.2 Hz stems from the implemented attitude controller and its prescribed control bandwidth as given in section 6.3. Figure 9.12 shows the power spectral density of the total torque in y-direction for a feedback simulation without considering any propellant, where the same peak at $\approx$ 0.2 Hz can be seen, but of course with a much lower total energy due to the fact that no propellant dynamics needs to be compensated by the attitude controller.

When comparing the power spectral densities of the total torque in y-direction for the two simulations with (figure 9.11 bottom in blue) and without (figure 9.12) propellant dynamics taken into account, it can further be seen that the minor modes at $\approx$ 4.6 Hz and around $\approx$ 6.6 Hz stem from the propellant, to be precise those are the artificial sound wave oscillations as analyzed in section 9.2.3. Although these artificial sound wave oscillations are present in the system, it can be seen in figure 9.11 bottom and top, that the control torque produced by the actuators (in red) is not affected by this numerical noise due to the small attitude control bandwidth.

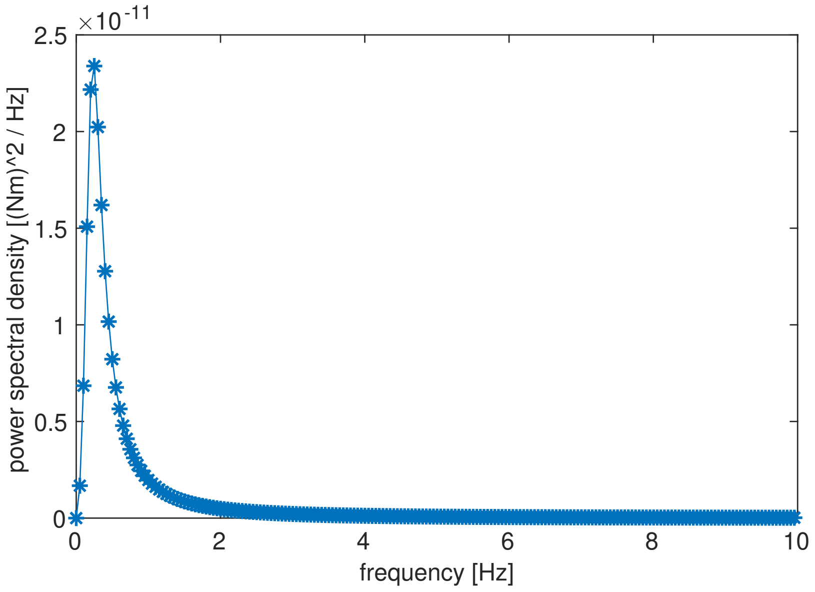


Figure 9.12: Earth observation satellite mission scenario: Simulation without considering the propellant dynamics: Power spectral density of the total torque in y-direction for a time between 40 and 60 s

In order to obtain a better understanding of the quality of the simulated attitude, the attitude error Euler vector $\vec{\mathbf{e}}_q$ is computed, which is a vector representing the Euler axis and angle needed to rotate from the current attitude to the desired attitude given by the guidance profile. This measure of the attitude error is determined by first calculating the attitude error quaternion $\mathbf{q}_e = (q_{e1}, q_{e2}, q_{e3}, q_{e4})$ by

$$\mathbf{q}_e = \begin{pmatrix} q_{d1} & -q_{d2} & -q_{d3} & -q_{d4} \\ q_{d2} & q_{d1} & -q_{d4} & q_{d3} \\ q_{d3} & q_{d4} & q_{d1} & -q_{d2} \\ q_{d4} & -q_{d3} & q_{d2} & q_{d1} \end{pmatrix} \begin{pmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{pmatrix} \tag{9.2.8}$$

using the desired attitude $\mathbf{q}_d = (q_{d1}, q_{d2}, q_{d3}, q_{d4})$ and the current attitude $\mathbf{q} = (q_1, q_2, q_3, q_4)$.

Then, based on the general definition of the quaternion $\mathbf{q}_e = (\cos\theta/2, u_1 \cdot \sin\theta/2, u_2 \cdot \sin\theta/2, u_3 \cdot \sin\theta/2)$ with $\vec{\mathbf{u}} = (u_1, u_2, u_3)$ being the unit vector describing the Euler axis and θ the rotation angle about the Euler axis, the Euler vector representation of the attitude error is determined by

$$\vec{\mathbf{e}}_q = \theta \cdot \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} \tag{9.2.9}$$

The resulting attitude error Euler vector is plotted in figure 9.13. It clearly shows the attitude error in y-direction due to the two y-axis slew maneuvers at $t = 20$ s and $t = 35$ s. The maximum resulting error is $\theta_{\text{fb}}^{\text{max}} \approx 1.5\text{e-}3$ rad, which corresponds to $\approx 86\text{e-}3$ deg.

The root-mean-square (RMS) of the attitude error angle, θ^{rms}, is now calculated using

$$\theta^{\text{rms}} = \sqrt{\frac{\sum_{i=1}^{N} \theta_i^2}{N}} \tag{9.2.10}$$

with N being the number of temporal discretization points and θ_i the attitude error angle at time $t = i \cdot \Delta t$. The simulation at hand then yields a RMS attitude error angle of $\theta_{\text{fb}}^{\text{rms}} = 3.3\text{e-}4$ rad.

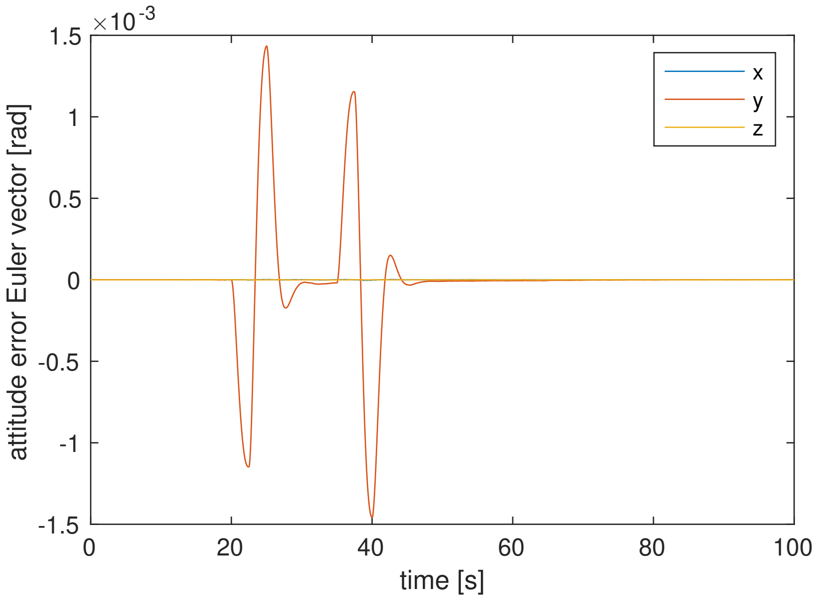


Figure 9.13: Earth observation satellite mission scenario: Attitude error Euler vector

In order to compare the results of the feedback simulation with a simulation without controller, i.e. feedforward, figure 9.14 shows the attitude error Euler vector for both cases for each axis. It can be clearly seen that the feedback simulation effectively compensates the attitude error, while the feedforward simulation causes huge and uncontrolled errors. In numbers, the feedforward simulation has a maximum error of $\theta_{\text{ff}}^{\text{max}} \approx 0.022$ rad, corresponding to ≈ 1.26 deg. The RMS attitude error angle for the feedforward simulation is calculated to be $\theta_{\text{ff}}^{\text{rms}} = 86.1\text{e-}4$ rad, meaning an error ≈ 26 times higher then for the feedback case.

When considering a typical Earth observation satellite mission like the Sentinel-2 [34] or MetOp [33] satellites, the satellite is orbiting Earth in a height of $h \approx 800$ km. An error of $\theta_{\text{fb}}^{\text{max}} \approx 1.5\text{e-}3$ rad for the feedback case would then translate into an error on ground of $e_{\text{fb}}^{\text{gnd}} \approx 1.2$ km, using the geometrical relation $e^{\text{gnd}} = h \cdot \tan\theta$ for a nadir-pointing target. The feedforward case would already yield an error on ground of $e_{\text{ff}}^{\text{gnd}} \approx 17.6$ km. The main objective of Earth observation satellites are ground scans of various types. For the WorldView-2 satellite [32] for example, which is orbiting Earth in a height of $h \approx 770$ km, these multispectral ground scans are performed using a swath width of 16.4 km at nadir. This makes it clear that an error on ground of $e_{\text{ff}}^{\text{gnd}} \approx 17.6$ km is unacceptable and even an error of $e_{\text{fb}}^{\text{gnd}} \approx 1.2$ km is already pretty large.

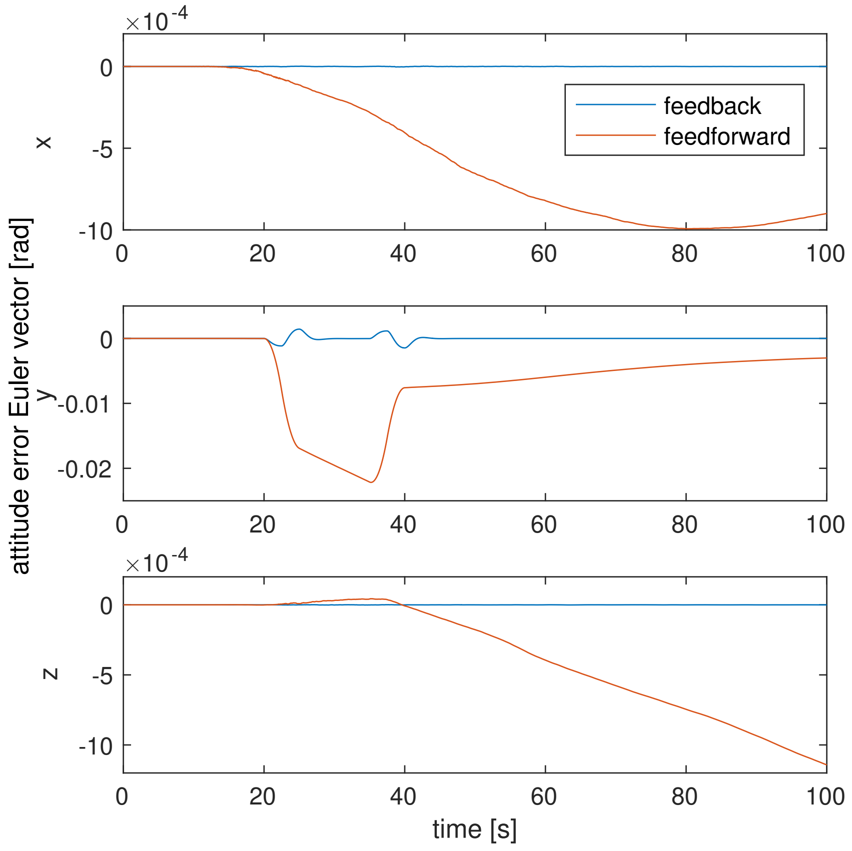


Figure 9.14: Earth observation satellite mission scenario: Attitude error Euler vector comparison between feedback (blue line) and feedforward (red line) simulations

Since attitude errors during slew maneuvers are naturally higher due to the induced angular accelerations, it is common to only perform ground scans after a slew maneuver has ended when the satellite has reached a certain stable state again (see [22]). This especially applies to the propellant, which has been excited during to the slew maneuver and which needs to settle again before the next ground scan can be performed. The time needed to perform the slew maneuver as well as the propellant settling time are main mission drivers which determine how many ground scans can be performed per orbit and thus during the lifetime of the satellite. For the WorldView-2 satellite for example, the pointing accuracy is given as $e_{\text{wv2}}^{\text{gnd}} < 500$ m at image start and stop [32]. Applying this performance parameter to the Earth observation satellite mission scenario at hand and the feedback simulation performed, a pointing accuracy $e_{\text{wv2}}^{\text{gnd}} < 500$ m corresponds to an attitude angle error of $\theta_{\text{fb}}^{\max} < 6.25\text{e-}4$ rad, which is achieved 1.02 s after each of the two slew maneuvers has ended. This means that 1.02 s after the slew maneuver has been completed, a ground scan could be performed with the desired pointing performance.

It is not the intent of this work to demonstrate a high-quality controller for performing arbitrary slew maneuvers in arbitrary gravity conditions, but to show the basic behavior of an AOCS with propellant sloshing included in the spacecraft's dynamics. Hence the controller used during the simulations hasn't been optimized for the Earth observation satellite mission scenario at hand, but standard values for the controller parameters proposed in literature (see section 6.3) have been used. Despite the fact that an optimized controller will likely result in a better performance, it can generally be said that the controller quality significantly contributes to the time needed to stabilize the spacecraft again after a slew maneuver has been performed.

Please note that for the feedforward simulation without considering the propellant, i.e. without CFD in the loop, the attitude error is approximately zero. This is due to the fact that ideal conditions regarding sensors and actuators are assumed at any time, i.e. that the state of the spacecraft is perfectly known and that the maneuvers are performed without any error. The errors found in the simulations using CFD in the loop are thus entirely induced by the propellant's dynamics and its coupling with the AOCS, showing firstly the relevance of considering propellant sloshing effects and secondly the need for a properly designed controller with respect to stability and performance.

The simulations of the Earth observation satellite mission scenario performed during this work demonstrate the ease of integrating propellant sloshing effects into AOCS simulations using the computational sloshing model presented in chapter 2. It is not needed to develop dedicated sloshing models for each the zero-gravity phase, the high-gravity phase and the transients from one to the other. The computational model developed in this work can be used for all the gravitational regimes without changing any model-specific parameters, so no switching between different models during the simulation is needed.

For setting up the computational sloshing model, no theoretical analyses and no intermediate semi-empirical models are required. There is also a common practice where firstly mechanical analog based sloshing models are derived based on extensive CFD simulations (one model for each dedicated working point) and secondly these mechanical analog models are then used for the AOCS simulations (see [63]). This also becomes obsolete, since the entire setup of the newly developed computational sloshing model is done by providing the involved physical parameters as for example the fluid properties or the tank geometry. Numerical parameters as e.g. the time step size or method-specific parameters like the speed of sound used due to the weakly compressible approach can be automatically determined based on given physical parameters as described in chapter 2.

Another advantage of the computational sloshing model at hand is the possibility to run the simulations with low resolutions as described in [23]. Since lower resolutions translate into smaller simulation durations, it becomes feasible to directly couple the computational sloshing model to the AOCS simulator having acceptable runtimes.

All this makes the developed computational propellant sloshing model a very good fit for AOCS verification campaigns, where a very large number of high-fidelity simulations are performed in order to predict the system behavior considering all relevant disturbances and their possible variations. The simulation runs during such an AOCS verification campaign are mainly independent from each other, hence these simulations can be trivially run in parallel, reducing massively the time needed to perform the entire campaign. This kind of parallelization is naturally given for the computational propellant sloshing model at hand.

Chapter 10

Conclusions

10.1 Thesis summary

The main goal of the thesis at hand was to contribute to the incorporation of nonlinear propellant slosh modeling in the AOCS design and verification in order to improve spacecraft performance and stability. This was achieved by developing and validating a computational propellant sloshing model able to reproduce nonlinear sloshing effects typically arising in spacecraft missions in high-gravity as well as low-gravity environments. The resulting software implementation was optimized for shorter simulation runtimes as well as better usability, yielding an easy to use and fast propellant sloshing simulator, which was subsequently coupled to a newly developed and validated spacecraft AOCS simulator. The coupled propellant sloshing and spacecraft attitude dynamics & control environment was finally used to showcase the applicability of the proposed computational propellant sloshing model to AOCS verification simulations. In the following please find a summary of the most important results of this thesis.

A computational propellant sloshing model was developed based on a transport-velocity formulation for the weakly-compressible smoothed particle hydrodynamics method. A significant advantage of this method is its ability to describe naturally fluid-fluid or fluid-solid interfaces by use of particles in a Lagrangian way, therefore lacking the need to reconstruct or track interfaces as it is often required in computational models available in literature. A combined density summation and discretized continuity equation approach is used that allows to stably determine the fluid's density also when simulating a single fluid phase only as done for a typical spacecraft application comprising a partially filled propellant tank containing gas that can be dynamically neglected compared to the liquid. Here, the usage of only a single fluid phase trivially yields a performance gain, since the equations of motion only need to be solved for the particles that represent the liquid phase. Surface tension as well as wetting effects were successfully included in the computational propellant sloshing model by using generic molecular-like cohesion and adhesion forces between like and dissimilar particles. Compared to existing propellant sloshing models, that approach has the advantage of not relying on a dedicated model for each individual effect occurring in the nature of fluids, but it provides a generic way of modeling molecular effects by cohesion and adhesion forces in both high-gravity and low-gravity environments. The particle interaction force strengths needed to setup this model for the specific tank-propellant-gas system of interest are derived from the experimentally available parameters surface tension and static contact angle, so no parameter calibration needs to be done. Additionally a wall boundary condition with dummy particles was used for the description of the tank wall structure that can be conveniently utilized for coupling the propellant sloshing dynamics with the spacecraft attitude & orbit dynamics and control.

The software implementation of the computational propellant sloshing model was done in C++ and focussed on the optimization of simulation runtime as well as on usability. A shared-memory parallelization technique based on OpenMP was implemented, offering the possibility to reduce the simulation runtime by using multiple concurrently-running threads for calculating independent tasks, without causing too much computational overhead when using low spatial resolutions. Convenient pre-processing and post-processing functionality was implemented around the computational propellant sloshing model, simplifying especially the setup of a simulation by creating predefined tank geometries and propellant distribution with only configuration parameters. Furthermore, as compared to other propellant sloshing simulators available in literature, the computational model and its software implementation presented in this work don't require "magical parameters", i.e. unphysical parameters like the artificial viscosity, which is been added to a simulation in order to account for numerical stability and whose value needs to be set before each simulation based on the specific application.

The computational propellant sloshing model and its software implementation described in this thesis were extensively validated for high-gravity environments through hydrostatic and dynamics sloshing simulations, comparing the simulation results to theory, experiments and other computational models. Additionally a study was performed under high-gravity conditions, showing that global system properties provide sufficient accuracy also for reduced spatial resolutions. Here, the sloshing frequencies calculated from the forces exerted by the fluid on the tank structure differed by only 1.6 % between the highest and lowest spatial resolution with 229,500 and 1,890 fluid particles, respectively, for a 3D simulation. Then the computational model was validated for zero-gravity environments through surface tension and wetting effects simulations, comparing the simulation results to theory. The wetting effects simulations were repeated using a lower spatial resolution of only 2,176 particles for a 3D simulation, still showing a remarkable good accuracy with a maximum error of 6.5 % for static contact angles greater than $50°$. The validation in high-gravity as well as low-gravity conditions showed that the computational propellant sloshing model provides sufficiently accurate results also for reduced spatial resolutions and thus for reduced simulation runtimes, making it suitable to be directly included in AOCS verification simulations.

Then a spacecraft AOCS simulator was developed in object-oriented Matlab and subsequently validated. The previously developed and validated propellant sloshing simulator was coupled to this spacecraft AOCS simulator using Matlab's MEX API, which allows to compile the propellant sloshing simulator C++ code in such a way that it can be directly called from within Matlab, thus yielding fast response times between the two simulators. Finally the coupled propellant sloshing and spacecraft attitude dynamics & control environment was used to simulate two test cases relevant for showing the suitability of the computational propellant sloshing model for AOCS verification applications. The first test case simulated a forced roll motion in high-gravity conditions. The resulting simulation results were compared to experiments and showed that the bulk motion of the sloshing liquid is reproduced very well throughout the entire simulation time. The second test case simulated a typical Earth observation satellite mission scenario in zero-gravity conditions, where a propellant inside a partially filled tank was excited by two successive, guided and controlled attitude bang-bang maneuvers. The implications of the sloshing propellant on the AOCS performance were discussed as well as the impact of the artificial sound wave oscillations, occurring due to the chosen weakly-compressible smoothed particle hydrodynamics approach, on the AOCS controller. Here a relationship was derived with which the maximum compressibility of the weakly-compressible liquid can be calculated based on the AOCS sampling frequency so that the AOCS controller is not affected by the artificial sound wave oscillations.

10.2 Future work

An area to improve the computational propellant sloshing model presented in this thesis is the fluid's damping behavior using low spatial resolutions. As could be seen during the validation of the computational model in high-gravity conditions, global system properties like the sloshing frequency due to the integrated forces exerted by the propellant on the tank structure are reproduced sufficiently well also for a reduced spatial resolution, but as a side effect, the sloshing damping ratio increases for decreasing spatial resolution. In the dynamic sloshing test performed for this analysis, the sloshing damping ratio increased from 0.9% for 229,500 particles to 6.8% for 1,890 particles for a 3D simulation. Finding a proper tradeoff between spatial resolution and thus simulation speed on the one hand and numerical dissipation on the other hand could be worth further investigation.

The developed computational propellant sloshing model including its software implementation have been optimized for shorter simulation runtimes, as is detailed in this thesis. A further reduction in time required to run a simulation could be achieved by switching the numerical integration method used for the temporal discretization of the fluid's equations of motion from a fixed time step to an adaptive time stepping approach (see e.g. [29, 40]). For example in typical Earth observation satellite missions, first a short time span occurs where a rotational maneuver takes place, involving accelerations on the spacecraft and thus on the propellant, requiring a smaller time step size. Afterwards a long stabilization and image capturing phase follows with a non-accelerated spacecraft and a settling propellant, where a higher time step size could be chosen. This adaptive time stepping could potentially lead to an even reduced simulation runtime.

A third improvement to make the proposed computational propellant sloshing model even more suitable for AOCS verification simulations is the parallelization of the coupled propellant sloshing and spacecraft attitude dynamics & control environment as a whole. During AOCS verification campaigns, a very large number of high-fidelity simulations are performed in order to predict the system behavior considering all relevant disturbances, including nonlinear propellant sloshing if deemed as relevant, and their possible variations. Since the majority of simulations performed in an AOCS verification campaign are independent, these simulations could be run also in parallel. This could be achieved by implementing a Microservice-based software architecture around both the spacecraft AOCS simulator and the propellant sloshing simulator. Using heavy automation, the coupled simulations could then be run in parallel in a cloud environment. Since public cloud environments offer a huge amount of compute resources shared among entire regions, it should be easily possible to perform hundreds or thousands of coupled AOCS verification simulations in parallel, thus limiting the time needed for the entire AOCS verification campaign to the runtime needed for one single simulation.

Bibliography

[1] Abramson, H. N.: *The Dynamic Behavior of Liquids in Moving Containers, with Applications to Space Vehicle Technology.* NASA SP-106, 1966

[2] Adami, S. ; Hu, X. ; Adams, N. : A generalized wall boundary condition for smoothed particle hydrodynamics. In: *Journal of Computational Physics* 231 (2012), aug, Nr. 21, S. 7057–7075

[3] Adami, S. ; Hu, X. ; Adams, N. A.: A transport-velocity formulation for smoothed particle hydrodynamics. In: *Journal of Computational Physics* 241 (2013), S. 292–307

[4] Adami, S. : *Modeling and Simulation of Multiphase Phenomena with Smoothed Particle Hydrodynamics.* München, Technische Universität München, Dissertation, 2014

[5] Allen, M. P. ; Tildesley, D. J.: *Computer Simulation of Liquids.* Oxford University Press, USA, 1989

[6] Behruzi, P. ; Rose, F. D. ; Cirillo, F. : Coupling sloshing, GNC and rigid body motions during ballistic flight phases. In: *52nd AIAA/SAE/ASEE Joint Propulsion Conference*, American Institute of Aeronautics and Astronautics (AIAA), jul 2016

[7] Botia-Vera, E. ; Souto-Iglesias, A. ; Bulian, G. ; Lobovský, L. : Three SPH Novel Benchmark Test Cases for free surface flows. In: *5th ERCOFTAC SPHERIC workshop on SPH applications*, 2010

[8] Bush, J. : 18.357 Interfacial Phenomena. In: *Massachusetts Institute of Technology: MIT OpenCourseWare* (Fall 2010). – https://ocw.mit.edu, License: Creative Commons BY-NC-SA

[9] Chandra, B. ; Collicott, S. : Low Gravity Propellant Slosh Prediction Using Surface Evolver. In: *38th AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit*, 2002

[10] Colagrossi, A. ; Landrini, M. : Numerical simulation of interfacial flows by smoothed particle hydrodynamics. In: *Journal of computational physics* 191 (2003), Nr. 2, S. 448–475

[11] Dehnen, W. ; Aly, H. : Improving convergence in smoothed particle hydrodynamics simulations without pairing instability. In: *Monthly Notices of the Royal Astronomical Society* 425 (2012), aug, Nr. 2, S. 1068–1082

[12] Delorme, L. ; Colagrossi, A. ; Souto-Iglesias, A. ; Zamora-Rodríguez, R. ; Botía-Vera, E. : A set of canonical problems in sloshing, Part I: Pressure field in forced roll. Comparison between experimental results and SPH. In: *Ocean Engineering* 36 (2009), feb, Nr. 2, S. 168–178

[13] Dodge, F. T.: *The new "dynamic behavior of liquids in moving containers".* Southwest Research Inst. San Antonio, TX, 2000

[14] Eickhoff, J. : *Simulating spacecraft systems.* Springer Science & Business Media, 2009

[15] Feynman, R. P. ; Leighton, R. B. ; Sands, M. : *Lectures on physics.* Bd. Vol. 2. Definitive ed. Pearson, Addison-Wesley, 2006

[16] Feynman, R. P. ; Leighton, R. B. ; Sands, M. : *Lectures on physics.* Bd. Vol. 1. Definitive ed. Pearson, Addison-Wesley, 2006

[17] Field, K. S.: Liquid Slosh Analysis Using Smoothed Particle Hydrodynamics. In: *Dissertations and Theses* Paper 159 (2014)

[18] Fries, N. ; Behruzi, P. ; Arndt, T. ; Winter, M. ; Netter, G. ; Renner, U. : Modelling of fluid motion in spacecraft propellant tanks-Sloshing. In: *Space Propulsion 2012 conference, 7th-10th May 2012, Bordeaux* (2012)

[19] Gambioli, F. ; Alegre Usach, R. ; Kirby, J. ; Wilson, T. ; Behruzi, P. : Experimental Evaluation of Fuel Sloshing Effects on Wing Dynamics. In: *International Forum on Aeroelasticity and Structural Dynamics*, 2019

[20] Gerrits, J. : *Dynamics of liquid-filled spacecraft: numerical simulation of coupled solid-liquid dynamics*, Diss., 2001

[21] Gingold, R. A. ; Monaghan, J. J.: Smoothed particle hydrodynamics: theory and application to non-spherical stars. In: *Monthly notices of the royal astronomical society* 181 (1977), Nr. 3, S. 375–389

[22] Hahn, M. ; Müller, T. ; Levenhagen, J. : An optimized end-to-end process for the analysis of agile earth observation satellite missions. In: *CEAS Space Journal* 6 (2014), Nr. 3-4, S. 145–154

[23] Hahn, M. ; Adami, S. ; Förstner, R. : Computational modeling of nonlinear propellant sloshing for spacecraft AOCS applications. In: *CEAS Space Journal* 10 (2018), Nr. 3, S. 441–451

[24] Hahn, M. ; Förstner, R. : A Nonlinear Computational Propellant Sloshing Model for Spacecraft in Micro-Gravity and Transient Regimes for AOCS Verification. In: *Deutscher Luft- und Raumfahrtkongress, München*, 2017

[25] Housner, G. W.: Dynamic pressures on accelerated fluid containers. In: *Bulletin of the seismological society of America* 47 (1957), Nr. 1, S. 15–35

[26] Hu, X. ; Adams, N. : A multi-phase SPH method for macroscopic and mesoscopic flows. In: *Journal of Computational Physics* 213 (2006), apr, Nr. 2, S. 844–861

[27] Hu, X. ; Adams, N. : An incompressible multi-phase SPH method. In: *Journal of Computational Physics* 227 (2007), nov, Nr. 1, S. 264–278

[28] Ibrahim, R. A.: *Liquid sloshing dynamics: theory and applications.* Cambridge University Press, 2005

[29] Ihmsen, M. ; Akinci, N. ; Gissler, M. ; Teschner, M. : Boundary handling and adaptive time-stepping for PCISPH. In: *Workshop on virtual reality interaction and physical simulation VRIPHYS*, 2010

[30] Jaiswal, O. ; Kulkarni, S. ; Pathak, P. : A study on sloshing frequencies of fluid-tank system. In: *Proceedings of the 14th World Conference on Earthquake Engineering*, 2008, S. 12–17

[31] Kordilla, J. ; Tartakovsky, A. M. ; Geyer, T. : A smoothed particle hydrodynamics model for droplet and film flow on smooth and rough fracture surfaces. In: *Advances in water resources* 59 (2013), S. 1–14

[32] Kramer, H. J.: *ESA Earth Observation Portal, Satellite Missions Database: WorldView-2.* https://earth.esa.int/web/eoportal/satellite-missions/v-w-x-y-z/worldview-2, accessed 06.01.2020

[33] Kramer, H. J.: *ESA Earth Observation Portal, Satellite Missions Database: MetOp.* https://earth.esa.int/web/eoportal/satellite-missions/m/metop, accessed 31.12.2019

[34] Kramer, H. J.: *ESA Earth Observation Portal, Satellite Missions Database: Sentinel-2.* https://earth.esa.int/web/eoportal/satellite-missions/c-missions/copernicus-sentinel-2, accessed 31.12.2019

[35] Laan, M. : *Signal sampling techniques for data acquisition in process control*, Diss., 1995

[36] Lucy, L. B.: A numerical approach to the testing of the fission hypothesis. In: *Astronomical Journal* 82 (1977), S. 1013–1024

[37] Marsell, B. ; Griffin, D. ; Schallhorn, P. ; Roth, J. : Integrated CFD and controls analysis interface for high accuracy liquid propellant slosh prediction. In: *AIAA paper* 308 (2012), S. 2012

[38] Monaghan, J. : Simulating Free Surface Flows with SPH. In: *Journal of Computational Physics* 110 (1994), feb, Nr. 2, S. 399–406

[39] Monaghan, J. : Smoothed Particle Hydrodynamics and Its Diverse Applications. In: *Annual Review of Fluid Mechanics* 44 (2012), Nr. 1, S. 323–346

[40] Monaghan, J. J.: Smoothed particle hydrodynamics. In: *Annual review of astronomy and astrophysics* 30 (1992), S. 543–574

[41] Monaghan, J. J.: Smoothed particle hydrodynamics. In: *Reports on progress in physics* 68 (2005), Nr. 8, S. 1703

[42] Monaghan, J. J.: SPH without a tensile instability. In: *Journal of computational physics* 159 (2000), Nr. 2, S. 290–311

[43] Nair, P. ; Pöschel, T. : Dynamic capillary phenomena using Incompressible SPH. In: *Chemical Engineering Science* 176 (2018), S. 192–204

[44] National Aeronautics and Space Administration (NASA): *NSSDCA Master Catalog: ATS 5.* accessed 28.09.2020

[45] Neer, J. ; Salvatore, J. : Fuel slosh energy dissipation on a spinning body. (1972)

[46] OpenMP Architecture Review Board: *OpenMP Application Programming Interface.* https://www.openmp.org, accessed 13.08.2020

[47] Paluszek, M. ; Razin, Y. ; Pajer, G. ; Mueller, J. ; Thomas, S. : *Spacecraft attitude and orbit control.* 3rd edition. Princeton Satellite Systems, Inc, 2012

[48] Peregrine, D. : Water-wave impact on walls. In: *Annual review of fluid mechanics* 35 (2003), Nr. 1, S. 23–43

[49] Press, W. H. ; Teukolsky, S. A. ; Vetterling, W. T. ; Flannery, B. P.: *Numerical recipes: the art of scientific computing, 3rd Edition.* Cambridge University Press, 2007

[50] Rebelo, T. ; Hahn, M. ; Cirillo, F. : Liquid Propellant Sloshing: A study about the use of Open Source CFD Software. In: *8th European Symposium on Aerothermodynamics for Space Vehicles, Lisbon, Portugal*, 2015

[51] Saksono, P. H. ; Peric, D. : On finite element modelling of surface tension: Variational formulation and applications - Part I: Quasistatic problems. In: *Computational Mechanics* 38 (2006), Nr. 3, S. 265–281

[52] Seelen, L. ; Padding, J. ; Kuipers, J. : Improved quaternion-based integration scheme for rigid body motion. In: *Acta Mechanica* 227 (2016), S. 3381–3389

[53] Souto-Iglesias, A. ; Botia-Vera, E. ; Bulian, G. : Repeatability and Two-Dimensionality of Model Scale Sloshing Impacts. In: *International Offshore and Polar Engineering Conference (ISOPE)*, 2011

[54] Souto-Iglesias, A. ; Botia-Vera, E. ; Martín, A. ; Pérez-Arribas, F. : A set of canonical problems in sloshing. Part 0: Experimental setup and data processing. In: *Ocean Engineering* 38 (2011), nov, Nr. 16, S. 1823–1830

[55] Space Exploration Technologies Corp (SpaceX): *Demo Flight 2 - Flight Review Update.* 15.06.2007

[56] Standard C++ Foundation: *C++ programming language.* https://isocpp.org, accessed 14.08.2020

[57] Tartakovsky, A. ; Meakin, P. : Modeling of surface tension and contact angles with smoothed particle hydrodynamics. In: *Physical Review E* 72 (2005), Nr. 2, S. 026301

[58] Tartakovsky, A. M. ; Panchenko, A. : Pairwise Force Smoothed Particle Hydrodynamics model for multiphase flow: Surface tension and contact line dynamics. In: *Journal of Computational Physics* 305 (2016), jan, S. 1119–1146

[59] Tewari, A. : *Advanced control of aircraft, spacecraft and rockets.* John Wiley & Sons, 2011

[60] The MathWorks, Inc.: *C++ MEX Applications.* https://www.mathworks.com/help/matlab/cpp-mex-file-applications.html, accessed 13.08.2020

[61] The MathWorks, Inc.: *Matlab R2018b.* https://de.mathworks.com/help/releases/R2018b/matlab, accessed 13.08.2020

[62] The MathWorks, Inc.: *Object-Oriented Design with Matlab.* https://de.mathworks.com/help/matlab/object-oriented-design-with-matlab.html, accessed 13.08.2020

[63] Theureau, D. ; Mignot, J. ; Tanguy, S. : Integration of low g sloshing models with spacecraft attitude control simulators. In: *Guidance, Navigation, and Control and Co-located Conferences.* American Institute of Aeronautics and Astronautics, 2013

[64] Uhlig, T. ; Sellmaier, F. ; Schmidhuber, M. : *Spacecraft Operations.* Springer Vienna, 2015

[65] Vacondio, R. ; Altomare, C. ; De Leffe, M. ; Hu, X. ; Le Touzé, D. ; Lind, S. ; Marongiu, J.-C. ; Marrone, S. ; Rogers, B. D. ; Souto-Iglesias, A. : Grand challenges for Smoothed Particle Hydrodynamics numerical schemes. In: *Computational Particle Mechanics* (2020), S. 1–14

[66] Veldman, A. ; Gerrits, J. ; Luppes, R. ; Helder, J. ; Vreeburg, J. : The numerical simulation of liquid sloshing on board spacecraft. In: *Journal of Computational Physics* 224 (2007), may, Nr. 1, S. 82–99

[67] Violeau, D. ; Rogers, B. D.: Smoothed particle hydrodynamics (SPH) for free-surface flows: past, present and future. In: *Journal of Hydraulic Research* 54 (2016), Nr. 1, S. 1–26

[68] Wertz, J. R.: *Spacecraft attitude determination and control.* 1978

[69] Wertz, J. R. ; Larson, W. J.: *Space mission analysis and design.* Microcosm Press, 1999

[70] Wie, B. : *Space vehicle dynamics and control.* 2nd edition. American Institute of Aeronautics and Astronautics, 2008

[71] Yuan, Y. ; Lee, T. R.: Contact angle and wetting properties. In: *Surface science techniques.* Springer, 2013, S. 3–34

[72] Zhang, C. ; Hu, X. Y. ; Adams, N. A.: A generalized transport-velocity formulation for smoothed particle hydrodynamics. In: *Journal of Computational Physics* 337 (2017), S. 216–232