ISRN LUTMDN/TMHP—13/5283—SE
ISSN 0282-1990

Hydrodynamic Analysis and Simulation of a Tidal Energy Converter

Marcus Jansson

Thesis for the Degree of Master of Science

Division of Fluid Mechanics
Department of Energy Sciences
Faculty of Engineering
Lund University

© Marcus Jansson
ISRN LUTMDN/TMHP–13/5283–SE
ISSN 0282–1990
Printed in Sweden
Lund 2013

Abstract

A new motion simulator has been developed to simulate Minesto AB's tidal energy converter flying in water. The motion simulator is as a part of a new computer based development environment, aiming to shorten the company's development process. The environment consists of a hydrodynamic analysis together with the new simulator to evaluate the performance of the company's tidal energy converter.

The new motion simulator is custom-made in MATLAB and Simulink for Minesto's purposes, simulating the motion of a flying tidal energy converter with six degrees of freedom in variable flow conditions. The hydrodynamic forces used in the simulator are calculated in the hydrodynamic analysis. Quaternions are used to avoid singularities in the angle representation, enabling the device to move freely without mathematical restrictions. The flow conditions are set in an external flow model with the ability to simulate variable flows. A control system is integrated in Simulink to control the device's rudder during the simulation.

The simulations are valid for normal flight conditions and are usable for development of both device and control system designs but also for educational purposes, especially if integrated with a visualization program.

Keywords: *Motion Simulation Development, Tidal Energy, Energy Converter, Flight Dynamics, Quaternions, MATLAB, Simulink*

Sammanfattning

En ny flygsimulator har utvecklats som en del i en ny datorbaserad utvecklingsmiljö, framtagen för att effektivisera Minesto AB:s produktutvecklingsprocess. Utvecklingsmiljön består av en hydrodynamisk analys och den nya simulatorn med syftet att artificiellt utvärdera prestandan hos företagets energiomvandlare.

Den nya simulatorn har sex frihetsgrader och är specialdesignad i MATLAB och Simulink för att simulera företagets flygande energiomvandlare i varierbara vattenflöden. De hydrodynamiska krafterna som används i simulatorn beräknas i den hydrodynamiska analysen. Kvaternioner är implementerade för att undvika singulariteter i vinkelrepresentationen. Flödeskarakteristiken vid simuleringen specificeras i en separat flödesmodul med möjligheterna att variera både storlek och riktning som funktion av tiden. Ett kontrollsystem är integrerat i Simulink för att styra energiomvandlarens roder under simuleringen.

Simuleringarna gäller för normala körtillstånd och är användbara till både produkt- och kontrollsystemsutveckling. Simulatorn är också lämplig för utbildningsändamål, speciellt om den integreras med ett visualiseringsprogram.

Nyckelord: *Simulatorutveckling, Tidvattenkraft, Energiomvandlare, Flygdynamik, Kvaternioner, MATLAB, Simulink*

Preface

This is a master's thesis project report for a Master of Science degree in Mechanical Engineering at the Faculty of Engineering, Lund University. The project was performed during the spring of 2013 at the company Minesto AB in Gothenburg. The project was supervised by Professor Johan Revstedt at the Department of Energy Sciences together with M.Sc. Erik Dölerud and M.Sc. Anders Movert at Minesto AB.

I want to dedicate my sincere gratitude to the employees of Minesto, making the thesis work a true positive experience, both off and on the floorball court. Special thanks to my supervisors; Anders Movert for building the foundation of the project and Erik Dölerud for the great and enjoyable collaboration in the HAMS project.

[illegible]

Marcus Jansson
Gothenburg, May 2013

Table of Contents

1 Introduction

1.1 Background

There are a few companies in the world developing power plants to extract energy from ocean currents and one of them is Minesto AB, located in Gothenburg, Sweden. Minesto AB is a pure development company with the objective to have a 3 MW power plant array in operation by 2015. The technology is very different from the competitors and proven to be both technically and economically sufficient. There is a detailed description of the technology in chapter 3 and at the company website [1].

Since Minesto AB is a development company, striving for fast results of high quality, the development process of a design needs to be short and cost efficient. Instead of building a prototype for each new design and test it in expensive test facilities, the development process could be severely shortened if the design first could be tested in a computer based environment.

1.2 Computer based development environment

The company started a project with the aim to build a computer based development environment from scratch for early designs with the ability to evaluate its performance due to changes in design, control system and flow environment.

The computer based development environment project, henceforth referred to as *HAMS – Hydrodynamic Analysis and Motion Simulation*, consists of six major subsystems, presented in Figure 1.1. A CAD-geometry is put in a hydrodynamic analysis program that works as an artificial water tunnel with the ability to calculate the hydrodynamic forces acting on the design due to various flow conditions, later on converted to dimensionless force coefficients. These coefficients are then tabulated and used in a motion simulation, together with a continuous communication with the control system and a flow model that characterizes the water flow at a certain site. The results of the simulation are then used to evaluate the performance of the design.

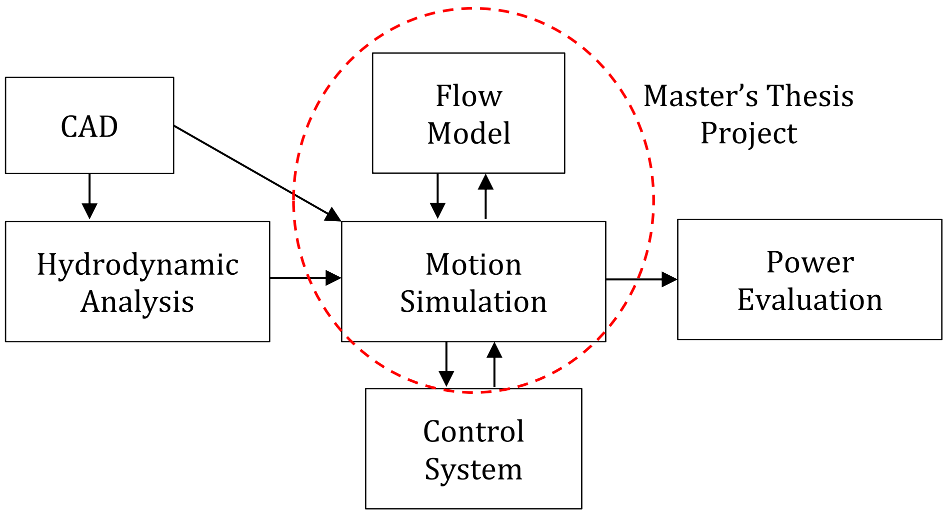

Figure 1.1 – Project overview.

The hydrodynamic analysis is made with a computer program solving the potential flow with coupled boundary layers. The potential flow method saves an impressive amount of time compared to solving the problem with a more advanced viscous method, though with a loss of accuracy in the solution.

1.3 Master's thesis project description

This master's thesis is a part of the HAMS project and, as shown in Figure 1.1, regards the motion simulation and the flow model in the system. The aim of this master's thesis project is to develop a new custom-made motion simulator with an external flow model using MATLAB and Simulink, simulating the energy converter flying in water. To be able to steer the simulation as desired, connectivity to Minesto's existing control system is to be established in Simulink. Figure 1.2 is a simplified overview of the Motion Simulation in Figure 1.1 that shows the major subsystems needed within the simulator.

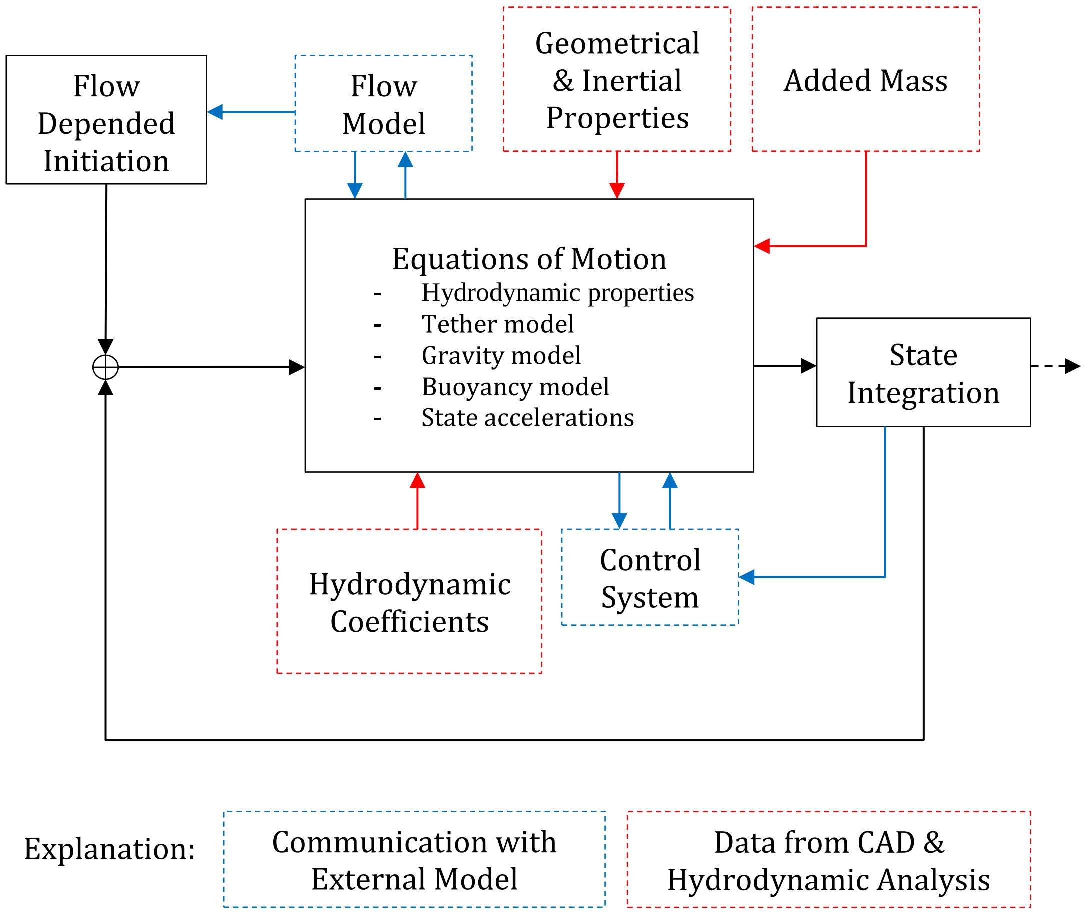

Figure 1.2 – Simplified overview of the Motion Simulation.

The simulation is desired to be initiated with respect to the prevailing flow conditions. A number of different models are needed to handle the influences of e.g. buoyancy, gravity and the tether attached to the sea floor. Information about the hydrodynamic forces for all possible conditions and the effects of added mass is gathered from the hydrodynamic analysis; information about geometric and inertial properties is gathered from a CAD program. The aim of the control system integration is to get the simulation steered in an infinity loop, a horizontal eight.

1.4 Limitations

The hydrodynamic analysis program lacks e.g. a turbulence model, and is not able to calculate the forces correctly when the designs get more advanced. Nor will it be able to catch all small design changes, due to the same reason. This is a major source of errors in the motion simulation that is not investigated in the thesis.

The variable buoyancy control system and the dynamics of the tether related to drag are excluded in the thesis work due to their complexities.

Possible seawater density variations, e.g. due to varying salt concentrations with depth, are neglected.

Working time within the frame of the thesis is a limiting factor; the company has to continue developing the simulator on its own. The effects from the turbine and the generator are excluded due to shortage of time.

2 Methodology

The design process of building a new simulator consists of at least four steps; study of literature, study of existing simulators, implementation and verification. During the work iterations between these steps are likely to be required.

2.1 Study of literature

Since the objective of this master's thesis is to simulate a rigid body flying in water one requires comprehensive knowledge about classical mechanics, flight dynamics and hydrodynamics. Besides the theory of motion this work demands knowledge about e.g. rotating reference frames, quaternions due to singularities in the Euler angle representation, wing theory to understand the effects of lift and drag, the added mass phenomenon due to motion in water and numerical issues occurring when using MATLAB. To be able to understand the intended behaviour of the simulator, the real system also needs to be thoroughly revised.

2.2 Study of existing simulators in MATLAB

There is probably no existing simulator for this particular application, but there are relevant simulators for airplanes, boats and underwater vehicles. Studying these simulators can be valuable for the understanding of how to build a similar simulator, and some of their help functions may be usable.

2.3 Design & implementation in MATLAB & Simulink

For simplicity a simulator is built in steps, introducing more advanced features along the work. This is done to easier isolate errors in the implementation as the degrees of freedom are as few as possible. Thus, a simple test simulator is built to begin with, only containing the simplest dynamics possible. When this works as desired more advanced models are implemented, e.g. tether model, buoyancy model and water flow model. In the end everything is implemented in Simulink and the control system is integrated.

2.4 Verification procedure

To ensure the simulator's accuracy it needs to be verified. This can be somewhat difficult if data is missing describing the dynamics of a certain design. One approach to get an idea of the performance is to set up a verification test using a simple geometry with an explicitly known behaviour. The simulation results are studied and characteristic quantities are plotted and related to expected behaviour.

3 The Principle of Minesto's Power Generation

3.1 A general description of the Deep Green Technology

The tidal energy device developed by Minesto AB, named Deep Green, converts kinetic energy in ocean currents to electricity. Briefly the technology is based on principles somewhat similar to kite surfing; a flying kite is maneuvered in a moving fluid, but in an ocean current instead of in the wind. A turbine and a nacelle with a generator inside are attached underneath the kite to generate electricity. Figure 3.1 shows a simplified description of the device.

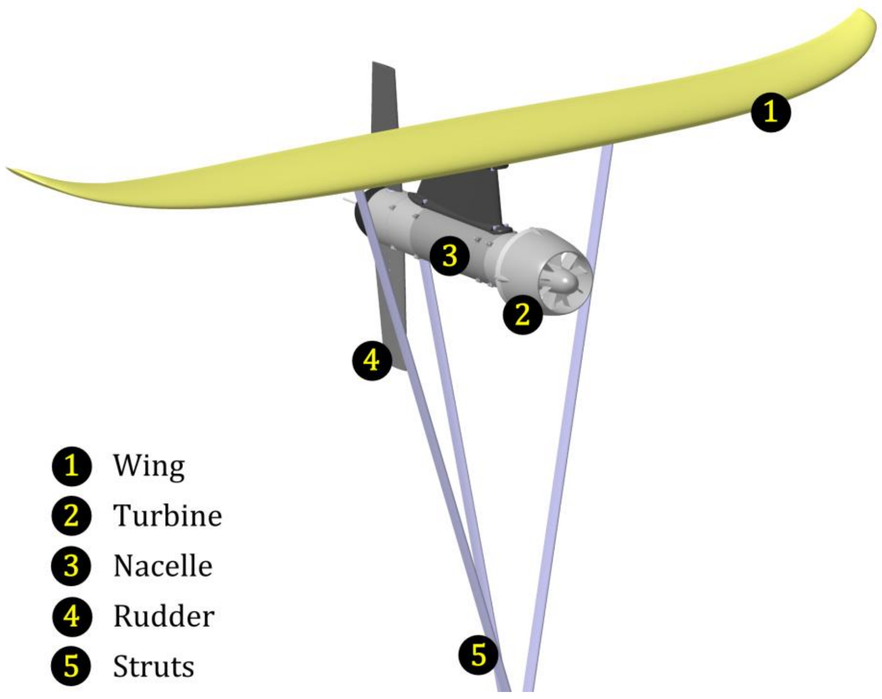

Figure 3.1 – A description of the Deep Green device.

When water flows over the hydrodynamically shaped wing, a lift force is generated that allows the device to move, the turbine starts to rotate and the generator produces electricity. The device is attached to the ocean floor with a tether and controlled by a rudder to follow an optimized trajectory. The tether is anchored in a swivel at a fixed point on the ocean floor which allows it to move and rotate freely around the fixed point. In the other end the tether is attached to the device' supporting struts. The electricity generated in the generator is transferred from the generator, through a cable inside the tether, to the onshore grid. The device also has a buoyancy control system to e.g. handle slack water between tides and to be able to operate at different depths. Figure 3.2 shows an illustration of the Deep Green device in operation, following the trajectory of an infinity loop.

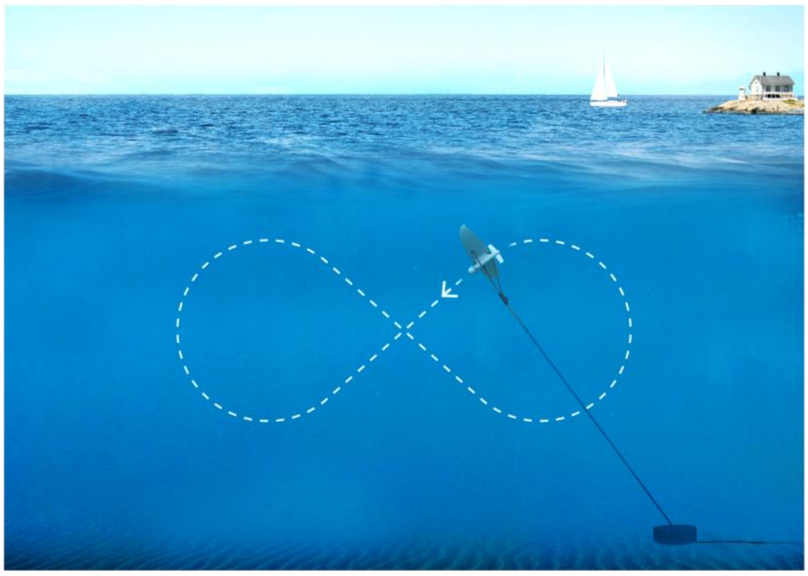

Figure 3.2 – An illustration of the Deep Green device in operation.

A similar description of the device and instruction videos showing the Deep Green Technology in operation is published on the company's website [1].

3.2 Hydrodynamic analysis and wing theory fundamentals

John D. Anderson explains in [2] that when a fluid passes over a wing profile, with an angle of attack α and a side slip angle β between the flow and the wing, the fluid is forced to be curved. The curving gives rise to a hydrodynamic force (R), often divided in three force components known as lift (L), side force (SF) and drag (D), and consequently a hydrodynamic moment (M); simply visualized in two dimensions in Figure 3.3. The hydrodynamic forces and moments on the wing are related to the pressure and shear stress distributions integrated over the wing profile, all thoroughly explained in detail in [2].

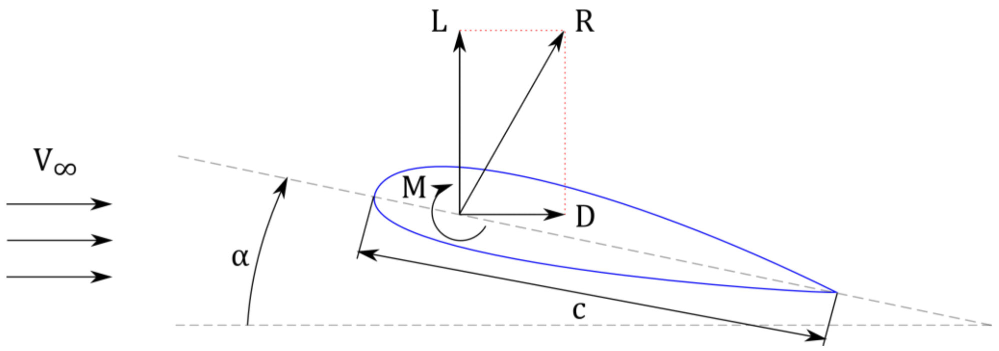

Figure 3.3 – Wing profile in 2D showing the hydrodynamic resultant force (R), divided in the two components lift (L) and drag (D), together with the hydrodynamic moment (M); all as a function of the angle of attack (α).

Anderson states in [2] that there are quantities of a more fundamental nature than the hydrodynamic forces and moments, which are the dimensionless force and moment coefficients. Equation (3.1) - (3.6) describes the general relations between the dimensionless coefficients and the hydrodynamic forces and moments in three dimensions; $\bar{q}$ is the dynamic pressure in the free stream, b is the characteristic length *wing span* for rolling and yawing moments, $\bar{c}$ is the characteristic length *mean* *aerodynamic chord length* for pitching moments and S is the characteristic *reference area*. These coefficients relate the forces and moments to the geometry and the dynamic pressure in the free stream, allowing us to easily scale the system and compare it with different designs.

$$C_D = \frac{D}{\bar{q}S} \tag{3.1}$$

$$C_L = \frac{L}{\bar{q}S} \tag{3.2}$$

$$C_{SF} = \frac{SF}{\bar{q}S} \tag{3.3}$$

$$C_{roll} = \frac{M_{roll}}{\bar{q}Sb} \tag{3.4}$$

$$C_{pitch} = \frac{M_{pitch}}{\bar{q}S\bar{c}} \tag{3.5}$$

$$C_{yaw} = \frac{M_{yaw}}{\bar{q}Sb} \tag{3.6}$$

The dynamic pressure is defined as (3.7) where ρ is the fluid density and V_∞ is the flow speed in the free stream.

$$\bar{q} = \frac{\rho V_\infty^2}{2} \tag{3.7}$$

3.3 Electric power generation

Similar to a traditional wind power plant the Deep Green device converts kinetic energy in a moving fluid to electricity with a turbine and a generator. As described in *Fluid Mechanics and Thermodynamics of Turbomachinery* [3], the kinetic energy in a flow is a function of fluid velocity in cubic, shown in (3.8) where ρ is the fluid density, A the turbine area and v the flow velocity.

$$P = \frac{1}{2}\rho A v^3 \tag{3.8}$$

Thus, an increase in flow speed is very favorable. The Deep Green device operates in ocean currents between 1 – 2.5 m/s and is designed to move more than 10 times faster than the ocean current due to the hydrodynamic wing, which increases the power available a 1000 times compared to static technologies.

4 Flight Dynamics

The theory of flight is truly cross-functional with connections to several disciplines e.g. classical mechanics, linear algebra, aerodynamics and automatic control. When operating under water the envelope expands even more, adding complexity to the problem.

In this chapter the theory used in the simulator for the governing motion equations are thoroughly reviewed. For clarity while reading equations in the report; matrices are bold and capital, vectors are bold and lowercase, and scalars are lowercase.

4.1 Frame of reference

Professor Robert F. Stengel at Princeton University states in his book *Flight Dynamics* [4] that the most convenient way to define the translational position and angular attitude of a flying rigid body, e.g. an aircraft, is by the position of its mass center together with the orientation of a body-fixed coordinate system relative to an inertial frame of reference. An appropriate inertial frame of reference for this application is the surface of the earth, even though it is not rigorously inertial due to its rotation around the sun.

The position and the velocity of the body mass center, relative to the surface of the earth, is expressed in a Cartesian coordinate system by three components x, y and z representing northerly, easterly and vertical movement from the origin. To preserve a *"right-hand-rule"* the vertical component is positive downwards, shown in Figure 4.1.

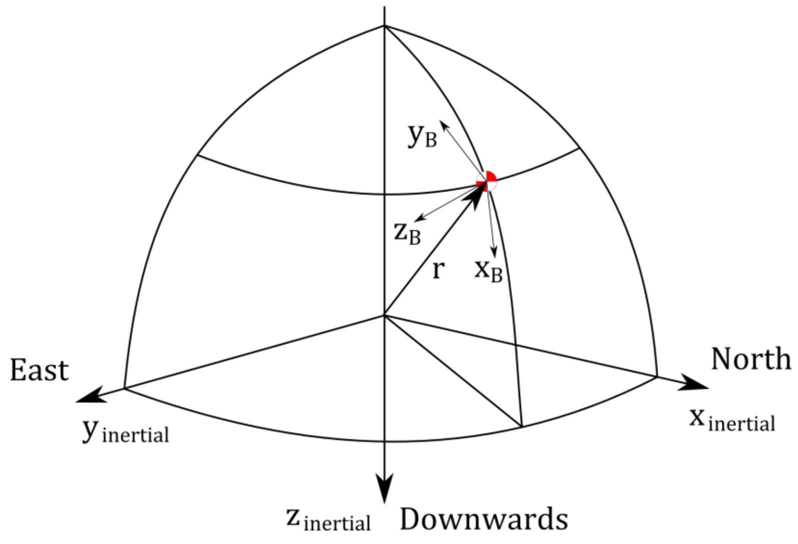

Figure 4.1 – Position of body mass center and body-fixed coordinate system in an inertial frame of reference.

The body-axis coordinate system, possibly accelerated and rotating relative to the inertial frame of reference, is fixed in the body mass center and represented by three components x_B, y_B and z_B where x_B is pointing forward, y_B to the right and z_B downwards, shown in Figure 4.2.

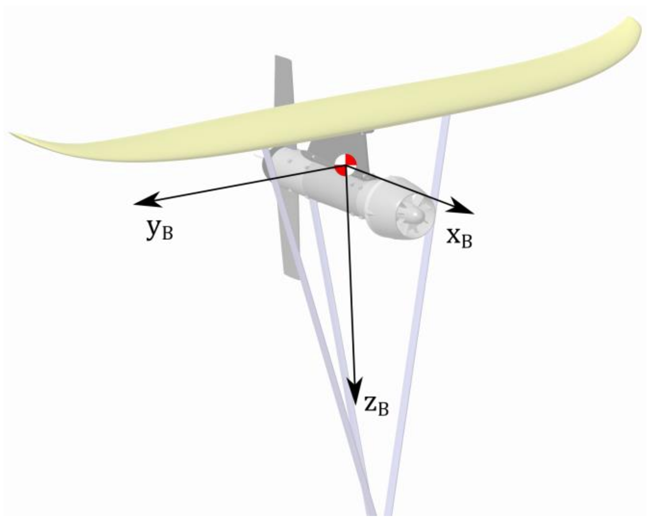

Figure 4.2 – Body-fixed coordinate system, positioned in the center of mass.

The angular orientation of the body-axis coordinate system about the inertial frame of reference is described by three Euler angles yaw (ψ), pitch (θ) and roll (φ); all of them positive for right-handed rotations about the axes, illustrated in Figure 4.3. Yaw represents a rotation about the z-axis, pitch represents a rotation about the y-axis and roll represents rotation about the x-axis.

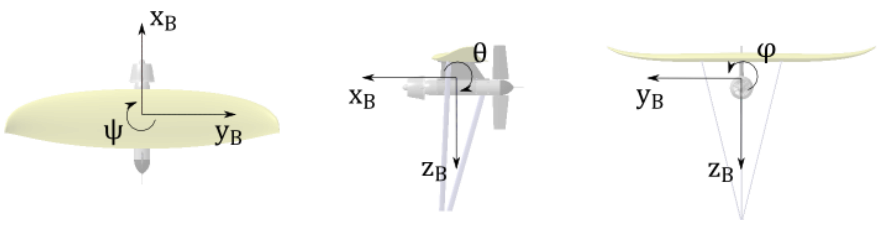

Figure 4.3 – Explanation of the angular rotations; yaw (ψ), pitch (θ), roll (φ).

4.1.1 Transformation between reference frames

Stengel explains in [4] that the Euler angles are sequential and therefore always needed to be executed in the same order. The order is arbitrary but set when once chosen; here by the sequence yaw, secondly pitch and lastly roll, often used in aerospace applications. This is significantly important when transforming quantities between two reference frames, e.g. the inertial and the body-fixed reference frame.

The 3-by-3 transformation matrix $\boldsymbol{H}_I^B(\varphi,\theta,\psi)$ transforms a translational quantity orthogonally from an inertial to a body-fixed reference frame; exemplified for a velocity transformation in (4.1). Note that the two velocity vectors $\boldsymbol{v}_B$ and $\boldsymbol{v}_I$ in (4.1) are the same vector but observed in two different axis systems.

$$\boldsymbol{v}_B = \boldsymbol{H}_I^B(\varphi,\theta,\psi)\boldsymbol{v}_I \tag{4.1}$$

The transformation matrix $\boldsymbol{H}_I^B(\varphi,\theta,\psi)$ is in fact a product of three transformation matrices for each rotation, executed with the right sequence, presented in (4.2).

$$\boldsymbol{H}_I^B(\varphi,\theta,\psi) = \boldsymbol{H}_2^B(\varphi)\boldsymbol{H}_1^2(\theta)\boldsymbol{H}_I^1(\psi)$$

$$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\varphi) & \sin(\varphi) \\ 0 & -\sin(\varphi) & \cos(\varphi) \end{bmatrix} \begin{bmatrix} \cos(\theta) & 0 & -\sin(\theta) \\ 0 & 1 & 0 \\ \sin(\theta) & 0 & \cos(\theta) \end{bmatrix} \begin{bmatrix} \cos(\psi) & \sin(\psi) & 0 \\ -\sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} \cos(\theta)\cos(\psi) & \cos(\theta)\sin(\psi) & -\sin(\theta) \\ \sin(\varphi)\sin(\theta)\cos(\psi) - \cos(\varphi)\sin(\psi) & \cos(\varphi)\cos(\psi) + \sin(\varphi)\sin(\theta)\sin(\psi) & \sin(\varphi)\cos(\theta) \\ \sin(\varphi)\sin(\psi) + \cos(\varphi)\sin(\theta)\cos(\psi) & \cos(\varphi)\sin(\theta)\sin(\psi) - \sin(\varphi)\cos(\psi) & \cos(\varphi)\cos(\theta) \end{bmatrix} \quad \textbf{(4.2)}$$

The transformation matrix $\boldsymbol{H}_I^B(\varphi,\theta,\psi)$ has a convenient quality; since it is a square, orthonormal matrix its inverse is its transpose. This makes the back-transformation easy; $\boldsymbol{H}_B^I = (\boldsymbol{H}_I^B)^{-1} = (\boldsymbol{H}_I^B)^T$.

Stengel continues in [4] that a transformation of Euler angle rates is on the other hand not as simple as for the translational case since the Euler angles are not measured along the axes and are completely separated. If $\dot{\boldsymbol{\Theta}}$ is the Euler angle rate vector, simply the Euler angles time derivatives, and $\boldsymbol{\omega}_B$ is the inertial angular rate vector measured about the body's axes, then they are related by the non-orthogonal transformation shown in (4.3).

$$\boldsymbol{\omega}_B = \begin{bmatrix} p \\ q \\ r \end{bmatrix} = \boldsymbol{L}_E^B\dot{\boldsymbol{\Theta}} = \boldsymbol{L}_E^B \begin{bmatrix} \dot{\varphi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \quad \textbf{(4.3)}$$

$\boldsymbol{L}_E^B$ is not an orthogonal transformation matrix but a result of three separate transformations by three different orthonormal matrices. The operation leading to the $\boldsymbol{L}_E^B$ matrix in (4.5) is shown in (4.4) where $\boldsymbol{I}_n$ is an identity matrix.

$$\boldsymbol{\omega}_B = \begin{bmatrix} p \\ q \\ r \end{bmatrix} = \boldsymbol{I}_n \begin{bmatrix} \dot{\varphi} \\ 0 \\ 0 \end{bmatrix} + \boldsymbol{H}_2^B(\varphi) \begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + \boldsymbol{H}_2^B(\varphi)\boldsymbol{H}_1^2(\theta) \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} \quad \textbf{(4.4)}$$

$$\boldsymbol{L}_E^B = \begin{bmatrix} 1 & 0 & -\sin(\theta) \\ 0 & \cos(\varphi) & \sin(\varphi)\cos(\theta) \\ 0 & -\sin(\varphi) & \cos(\varphi)\cos(\theta) \end{bmatrix} \quad \textbf{(4.5)}$$

Since $\boldsymbol{L}_E^B$ is not orthonormal its transpose is not its inverse and therefore the back-transformation matrix is $\boldsymbol{L}_B^E = (\boldsymbol{L}_E^B)^{-1}$, shown expanded in (4.6).

$$\boldsymbol{L}_B^E = (\boldsymbol{L}_E^B)^{-1} = \begin{bmatrix} 1 & \sin(\varphi)\tan(\theta) & \cos(\varphi)\tan(\theta) \\ 0 & \cos(\varphi) & -\sin(\varphi) \\ 0 & \sin(\varphi)\sec(\theta) & \cos(\varphi)\sec(\theta) \end{bmatrix} \quad \textbf{(4.6)}$$

Unfortunately, the back-transformation matrix $\boldsymbol{L}_B^E$ contains singularities due to the nature of the trigonometric tangent function; the value approaches infinity when the angle goes up to ±90 degrees. In this case, the transformation crashes when the pitch angle goes up to ±90 degrees. This is unacceptable for this application where the device is supposed to be able to move freely in all kinds of directions. It is a well-known disadvantage with the Euler angle representation and the most appropriate solution according to Sir William Rowan Hamilton [5] is to implement *Quaternions*.

4.1.2 Quaternions to avoid singularities

Quaternions are not easy to grasp and this report would get too massive if they were to be thoroughly explained. Instead, the details are left in the referenced literature for the interested reader and the quaternions are briefly described what they are and how they are used in this thesis.

Jack B. Kuipers states in the article *Quaternions and Rotation Sequences* [6] that quaternions are a number system $\mathbb{H}$ that extends the complex numbers $\mathbb{C}$ and form a four-dimensional vector space; visualized in Figure 4.4. Their primary application is in rotation operations. Contrary to Euler angles the quaternions do not have problems with singularities, which is the main reason of their existence. Kupiers also states that quaternions can offer computational advantages compared to Euler angles; calculations can be made faster.

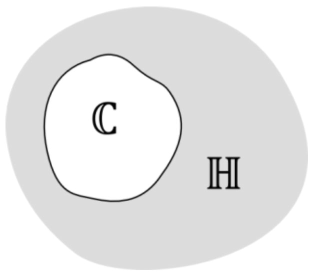

Figure 4.4 – The quaternions $\mathbb{H}$ extend the complex numbers $\mathbb{C}$.

Instead of representing the angular attitude of the device by a three-dimensional Euler angle vector it is described by a four-dimensional quaternion vector, defined by (4.7) and visualized in Figure 4.5 where its components are a scalar $\eta \in \mathbb{R}$ and an imaginary vector $\boldsymbol{\varepsilon} = [\varepsilon_1 \quad \varepsilon_2 \quad \varepsilon_3]^T \in \mathbb{R}^3$.

$$\boldsymbol{q} = \begin{bmatrix} \eta \\ \boldsymbol{\varepsilon} \end{bmatrix} \tag{4.7}$$

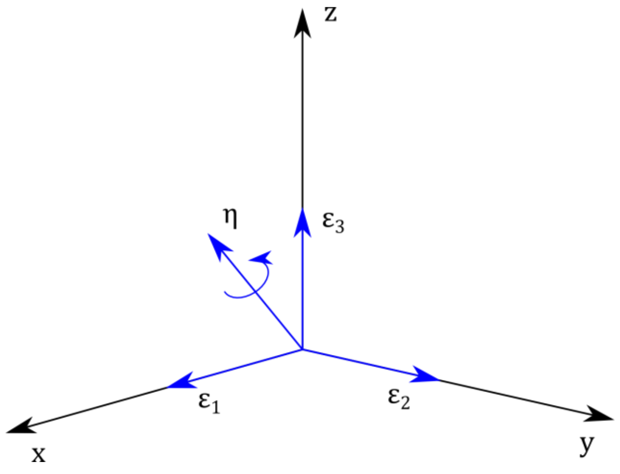

Figure 4.5 – Definition of the quaternion components.

According to Professor Thor I. Fossen at the Norwegian University of Science and Technology in Trondheim [7], the quaternions can initially be set by the Euler angles but later be independent of them. Though, the Euler angles can always be recovered from the quaternions by relating the elements in the transformation matrix $\boldsymbol{H}_I^B(\varphi,\theta,\psi)$ in (4.2) with the corresponding elements in the quaternion transformation matrix $\boldsymbol{H}_I^B(\boldsymbol{q})$ in (4.8). The translational transformation matrix $\boldsymbol{H}_I^B(\varphi,\theta,\psi)$ in (4.1) is simply replaced by $\boldsymbol{H}_I^B(\boldsymbol{q})$ when the quaternions are implemented. The characteristics of the transformation matrix are preserved, meaning the inverse is still its transpose.

$$\boldsymbol{H}_I^B(\boldsymbol{q}) = \begin{bmatrix} (\eta^2-\varepsilon_1^2-\varepsilon_2^2+\varepsilon_3^2) & 2(\eta\varepsilon_1+\varepsilon_2\varepsilon_3) & 2(\varepsilon_1\varepsilon_3-\eta\varepsilon_2) \\ 2(\varepsilon_2\varepsilon_3-\eta\varepsilon_1) & (\eta^2-\varepsilon_1^2+\varepsilon_2^2-\varepsilon_3^2) & 2(\varepsilon_1\varepsilon_2+\eta\varepsilon_3) \\ 2(\eta\varepsilon_2+\varepsilon_1\varepsilon_3) & 2(\varepsilon_1\varepsilon_2-\eta\varepsilon_3) & (\eta^2+\varepsilon_1^2-\varepsilon_2^2-\varepsilon_3^2) \end{bmatrix} \quad \textbf{(4.8)}$$

The whole point of implementing the quaternions is the absence of singularities in the transformation procedure of angular rates. The quaternion transformation matrix $\boldsymbol{L}_B^E(\boldsymbol{q})$ in (4.9) replaces the Euler transformation matrix $\boldsymbol{L}_B^E$ in (4.6), though it is used in a different way. The $\boldsymbol{L}_B^E(\boldsymbol{q})$ matrix does not only transform the angular rates to another reference frame, it transforms them to quaternion rates, which is convenient for the motion equations later derived in section 4.3. The transformation procedure using $\boldsymbol{L}_B^E(\boldsymbol{q})$ is shown in (4.10).

$$\boldsymbol{L}_B^E(\boldsymbol{q}) = \frac{1}{2}\begin{bmatrix} -\varepsilon_1 & -\varepsilon_2 & -\varepsilon_3 \\ \eta & -\varepsilon_3 & \varepsilon_2 \\ \varepsilon_3 & \eta & -\varepsilon_1 \\ -\varepsilon_2 & \varepsilon_1 & \eta \end{bmatrix} \quad \textbf{(4.9)}$$

$$\dot{\boldsymbol{q}} = \boldsymbol{L}_B^E(\boldsymbol{q})\boldsymbol{\omega}_\mathrm{B} = \boldsymbol{L}_B^E(\boldsymbol{q})\begin{bmatrix} p \\ q \\ r \end{bmatrix} \quad \textbf{(4.10)}$$

4.2 A description of the states

To be able to simulate the motion of a rigid body certain states are needed to be known and updated at all times. Stengel explains in [4] the states suitable for a state-based model for this application, briefly described in Table 4.1 and put in a state vector $\boldsymbol{x}$ in (4.11). Note the different reference frames for the states, some expressed in a body-fixed reference frame and some in the inertial earth reference frame.

Table 4.1 – The states of a rigid body.

State	Variable	Unit	Reference frame
u	Velocity of body-axis X	[m/s]	Body-axis
v	Velocity of body-axis Y	[m/s]	Body-axis
w	Velocity of body-axis Z	[m/s]	Body-axis
x	North position of body	[m]	Earth-axis
y	East position	[m]	Earth-axis
z	Negative altitude	[m]	Earth-axis
p	Roll rate	[rad/s]	Body-axis
q	Pitch rate	[rad/s]	Body-axis
r	Yaw rate	[rad/s]	Body-axis
φ	Roll angle	[rad]	Earth-axis
θ	Pitch angle	[rad]	Earth-axis
ψ	Yaw angle	[rad]	Earth-axis

$$\boldsymbol{x} = [u\ v\ w\ x\ y\ z\ p\ q\ r\ \varphi\ \theta\ \psi]^T \quad \textbf{(4.11)}$$

The states describe all possible positions and movements of a rigid body; surge, sway, heave, roll, pitch and yaw, with the directions presented in Figure 4.6.

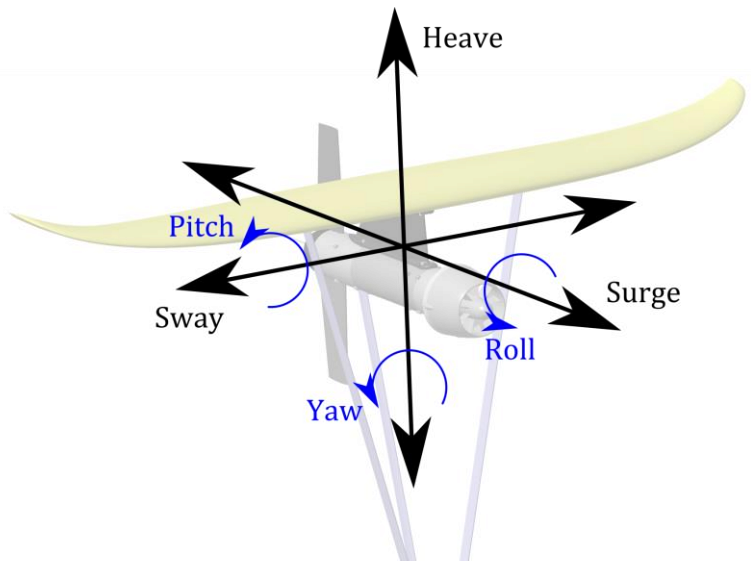

Figure 4.6 – Description of all possible movements in six dimensions.

4.2.1 The states used in the simulator

As described in section 4.1.2, quaternions are used instead of Euler angles to represent the angular attitude to avoid singularities. Thus, the states used in the simulator are as shown in Table 4.2 and the state vector is updated to (4.12). Unfortunately, this representation of the states is less intuitive since the quaternions are difficult to read. That cannot be helped since the absence of singularities is of greater importance.

Table 4.2 – The states used in the simulator.

State	Variable	Unit	Reference frame
u	Velocity of body-axis X	[m/s]	Body-axis
v	Velocity of body-axis Y	[m/s]	Body-axis
w	Velocity of body-axis Z	[m/s]	Body-axis
x	North position of body	[m]	Earth-axis
y	East position	[m]	Earth-axis
z	Negative altitude	[m]	Earth-axis
p	Roll rate	[rad/s]	Body-axis
q	Pitch rate	[rad/s]	Body-axis
r	Yaw rate	[rad/s]	Body-axis
η	Quaternion constituent	[-]	Quaternion
ε_1	Quaternion constituent	[-]	Quaternion
ε_2	Quaternion constituent	[-]	Quaternion
ε_3	Quaternion constituent	[-]	Quaternion

$$\boldsymbol{x} = [u\ v\ w\ x\ y\ z\ p\ q\ r\ \eta\ \varepsilon_1\ \varepsilon_2\ \varepsilon_3]^T \tag{4.12}$$

4.3 The equations of motion

Stengel writes in [4] that the motion of a rigid body is characterized by four sets of time dependent differential equations, describing the dynamics and kinematics for translation and rotation. In other words, the equations are describing the time derivatives of the states, presented as $\dot{\boldsymbol{x}}$ in (4.13). The states are them updated by integration in time according to (4.14).

$$\dot{\boldsymbol{x}} = [\dot{u}\ \dot{v}\ \dot{w}\ \dot{x}\ \dot{y}\ \dot{z}\ \dot{p}\ \dot{q}\ \dot{r}\ \dot{\eta}\ \dot{\varepsilon_1}\ \dot{\varepsilon_2}\ \dot{\varepsilon_3}]^T \tag{4.13}$$

$$\boldsymbol{x}(t) = \boldsymbol{x}(t_0) + \int_{t_0}^{t} \dot{\boldsymbol{x}}\,(t)dt \tag{4.14}$$

Note that the equations through the chapter are derived with the following assumptions, expressed in a body-fixed frame of reference:

- Flat earth assumption
- The device is a rigid body
- No change in mass or inertia
- The angular momentum of the turbine is neglected

4.3.1 Translational and rotational dynamics

The translational and rotational motion due to forces and moments acting on a solid body is described by *Newton's second law of motion* [4]; in this case two vector equations (4.15) and (4.16). The translational dynamics is described by (4.15) where $\boldsymbol{a}$ is the translational acceleration vector, $\boldsymbol{f}$ contains all forces acting on the body, m is the mass of the body, $\boldsymbol{\omega}$ contains the angular velocities and $\boldsymbol{v}$ the velocities of the body-axes. Regard the cross-product between the angular velocity vector and the velocity vector; this describes the inertial coupling associated with orthogonal axes due to rotating reference frames, often referred to as the *Coriolis effect.*

$$\boldsymbol{a} = \frac{d\boldsymbol{v}}{dt} = \begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} = \frac{\boldsymbol{f}}{m} - \boldsymbol{\omega} \times \boldsymbol{v} \tag{4.15}$$

The rotational dynamics are described with the vector equation (4.16) where $\dot{\boldsymbol{\omega}}$ are the angular accelerations, $\boldsymbol{I}$ is the moments of inertia matrix, $\boldsymbol{m}$ contains all moments acting on the body and $\boldsymbol{\omega}$ contains the angular velocity. Once again there is a cross product due to the *Coriolis effect.*

$$\dot{\boldsymbol{\omega}} = \frac{d\boldsymbol{\omega}}{dt} = \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \boldsymbol{I}^{-1}\big(\boldsymbol{m} - \boldsymbol{\omega} \times (\boldsymbol{I} \cdot \boldsymbol{\omega})\big) \tag{4.16}$$

4.3.2 Translational and rotational kinematics

Since the state vector $\boldsymbol{x}$ in (4.12) contains the velocities of the body, the equation describing the translational kinematics in the inertial frame of reference is simply a transformation of the velocities. Equation (4.17) shows the transformation procedure from a body-fixed reference frame to an inertial earth reference frame by using the transformation matrix $\boldsymbol{H}_B^I(\boldsymbol{q})$ derived in section 4.1.2.

$$\frac{d\boldsymbol{r}_I}{dt} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \boldsymbol{H}_B^I(\boldsymbol{q})\boldsymbol{v}_B = \boldsymbol{H}_B^I(\boldsymbol{q}) \begin{bmatrix} u \\ v \\ w \end{bmatrix} \tag{4.17}$$

The equation describing the rotational kinematics, when using quaternions, is already presented as (4.10) in section 4.1.2.

4.3.3 Added mass

Fossen explains in [7] that there is a phenomenon that needs to be added to the equations of motion due to the fact that we are working in water; often referred to as *added mass* or *virtual mass*. The total mass that is in motion is not only the mass of the body but also the mass of the water pushed away by the body. This changes the equations of motion significantly because the total mass and the moment of inertia of the system are not only changed individually but also cross coupled together in all dimensions. The mass is therefore no longer a scalar but a part of a 6-by-6 matrix together with the moments of inertia and the cross couplings between the two. The two equations (4.15) and (4.16) are no longer valid separately, they need to be put together in matrix form to include the cross couplings in the system.

4.3.4 The equations of motion in matrix form

Due to the added mass, the equations of motion need to be put together in matrix form to capture the cross couplings between the mass and the moments of inertia of the system. This was possible before the added mass was included but it is associated with far more complicated calculations compared to having them separated.

Considering the equations of motion (4.15) and (4.16), before adding the added mass, one can set up the equations in matrix form as in (4.18) where $\boldsymbol{a}$ is the system's accelerations, $\boldsymbol{M}$ is the inertia matrix visualized in (4.19) containing the mass and the moments of inertia, $\boldsymbol{F}$ contains the forces and moments acting on the body and $\boldsymbol{v}$ contains the translational and rotational velocities.

$$\boldsymbol{a} = \frac{d\boldsymbol{v}}{dt} = \boldsymbol{M}^{-1}\big(\boldsymbol{F} + \boldsymbol{v} \times (\boldsymbol{M} \cdot \boldsymbol{v})\big) \qquad \textbf{(4.18)}$$

$$\boldsymbol{M} = \begin{bmatrix} m & 0 & 0 & 0 & 0 & 0 \\ 0 & m & 0 & 0 & 0 & 0 \\ 0 & 0 & m & 0 & 0 & 0 \\ 0 & 0 & 0 & I_{xx} & -I_{xy} & -I_{xz} \\ 0 & 0 & 0 & -I_{xy} & I_{yy} & -I_{yz} \\ 0 & 0 & 0 & -I_{xz} & -I_{yz} & I_{zz} \end{bmatrix} \qquad \textbf{(4.19)}$$

The added mass matrix is a full 6-by-6 matrix $\boldsymbol{A}$ that is to be added to the inertia matrix $\boldsymbol{M}$ that becomes $\boldsymbol{M}_A$ and which changes the equations of motion from (4.18) to (4.20).

$$\boldsymbol{a} = \frac{d\boldsymbol{v}}{dt} = \boldsymbol{M}_A^{-1}\big(\boldsymbol{F} + \boldsymbol{v} \times (\boldsymbol{M}_A \cdot \boldsymbol{v})\big) \qquad \textbf{(4.20)}$$

Unfortunately, as the alert mathematician may have recognized, there is no such thing as a cross product in six dimensions. This is why the calculations become complicated, but not impossible. There is a workaround to exchange a cross product with a matrix multiplication using a *cross product equivalent matrix*[1]; in this case it is needed in several dimensions.

According to Fossen in [7], there is a matrix $\boldsymbol{C}$ that contains of two 3-by-3 cross product equivalent matrices $\boldsymbol{S}$ to handle the cross product between the vectors. The new vector equation describing the motion becomes (4.21), at its simplest form, where matrix $\boldsymbol{C}$ is shown in (4.22).

$$\boldsymbol{a} = \frac{d\boldsymbol{v}}{dt} = \boldsymbol{M}_A^{-1}(\boldsymbol{F} + \boldsymbol{C} \cdot \boldsymbol{v}) \qquad \textbf{(4.21)}$$

$$\boldsymbol{C} = \begin{bmatrix} 0 & -\boldsymbol{S}_1 \\ -\boldsymbol{S}_1 & -\boldsymbol{S}_2 \end{bmatrix} \qquad \textbf{(4.22)}$$

[1] A cross product equivalent matrix $\boldsymbol{A}$ is a skew symmetric matrix satisfying $\boldsymbol{A}^T = -\boldsymbol{A}$ and is created in such a way that the vector multiplication $\boldsymbol{c} = \boldsymbol{a} \times \boldsymbol{r}$ can be exchanged with the matrix multiplication $\boldsymbol{c} = \boldsymbol{A}(\boldsymbol{a}) \cdot \boldsymbol{r}$, [7].

The $\boldsymbol{S}$ matrices are skew symmetric and are consequently the same in their structure but depend on different vectors as parameters. Fossen explains that the parameters are derived from *Kirchhoff's Equations in Vector Form* (4.23)-(4.24) where T is the kinetic energy, $\boldsymbol{\tau}_1$ is the force vector, $\boldsymbol{\tau}_2$ is the moment vector, $\boldsymbol{v}_1$ is the translational speed vector and $\boldsymbol{v}_2$ the angular velocity vector. These equations describe the motion of a rigid body in an ideal fluid using the fluid's kinetic energy (4.25). Fossen remarks that any motion of a rigid body in an otherwise stationary fluid will induce a motion in the fluid. The fluid needs to move aside and close behind the body when the body passes through the fluid. As a result the fluid has kinetic energy it would lack otherwise.

$$\frac{d}{dt}\left(\frac{\partial T}{\partial \boldsymbol{v}_1}\right) + \boldsymbol{S}_{(\boldsymbol{v}_2)}\left(\frac{\partial T}{\partial \boldsymbol{v}_1}\right) = \boldsymbol{\tau}_1 \tag{4.23}$$

$$\frac{d}{dt}\left(\frac{\partial T}{\partial \boldsymbol{v}_2}\right) + \boldsymbol{S}_{(\boldsymbol{v}_2)}\left(\frac{\partial T}{\partial \boldsymbol{v}_2}\right) + \boldsymbol{S}_{(\boldsymbol{v}_1)}\left(\frac{\partial T}{\partial \boldsymbol{v}_1}\right) = \boldsymbol{\tau}_2 \tag{4.24}$$

$$T = \frac{1}{2}\boldsymbol{v}^T \boldsymbol{M}_A \boldsymbol{v} \tag{4.25}$$

Differentiating (4.25) with respect to $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ gives the parameters $\frac{\partial T}{\partial \boldsymbol{v}_1}$ and $\frac{\partial T}{\partial \boldsymbol{v}_2}$ used in the matrices $\boldsymbol{S}_1$ and $\boldsymbol{S}_2$, presented in (4.26)-(4.28).

$$\begin{aligned} \frac{\partial T}{\partial \boldsymbol{v}_1} &= \boldsymbol{M}_{A,11}\boldsymbol{v}_1 + \boldsymbol{M}_{A,12}\boldsymbol{v}_2 \\ \frac{\partial T}{\partial \boldsymbol{v}_2} &= \boldsymbol{M}_{A,21}\boldsymbol{v}_1 + \boldsymbol{M}_{A,22}\boldsymbol{v}_2 \end{aligned} \tag{4.26}$$

$$\begin{aligned} \boldsymbol{M}_{A,11} &= \boldsymbol{M}_{A\ (1:3,1:3)} \\ \boldsymbol{M}_{A,12} &= \boldsymbol{M}_{A\ (1:3,4:6)} \\ \boldsymbol{M}_{A,21} &= \boldsymbol{M}^T_{A\ (1:3,4:6)} \\ \boldsymbol{M}_{A,11} &= \boldsymbol{M}_{A\ (4:6,4:6)} \end{aligned} \tag{4.27}$$

$$\boldsymbol{S}_1 = \begin{bmatrix} 0 & -\frac{\partial T}{\partial \boldsymbol{v}_2}(3) & \frac{\partial T}{\partial \boldsymbol{v}_2}(2) \\ \frac{\partial T}{\partial \boldsymbol{v}_2}(3) & 0 & -\frac{\partial T}{\partial \boldsymbol{v}_2}(1) \\ -\frac{\partial T}{\partial \boldsymbol{v}_2}(2) & \frac{\partial T}{\partial \boldsymbol{v}_2}(1) & 0 \end{bmatrix}$$

$$\boldsymbol{S}_2 = \begin{bmatrix} 0 & -\frac{\partial T}{\partial \boldsymbol{v}_1}(3) & \frac{\partial T}{\partial \boldsymbol{v}_1}(2) \\ \frac{\partial T}{\partial \boldsymbol{v}_1}(3) & 0 & -\frac{\partial T}{\partial \boldsymbol{v}_1}(1) \\ -\frac{\partial T}{\partial \boldsymbol{v}_1}(2) & \frac{\partial T}{\partial \boldsymbol{v}_1}(1) & 0 \end{bmatrix} \tag{4.28}$$

4.3.5 The state-space model used in the simulator

Finally, the state-space model is thoroughly derived, including the effects of added mass. Figure 4.7 and Figure 4.8 summarizes section 4.3 and illustrates the state-space model (4.14).

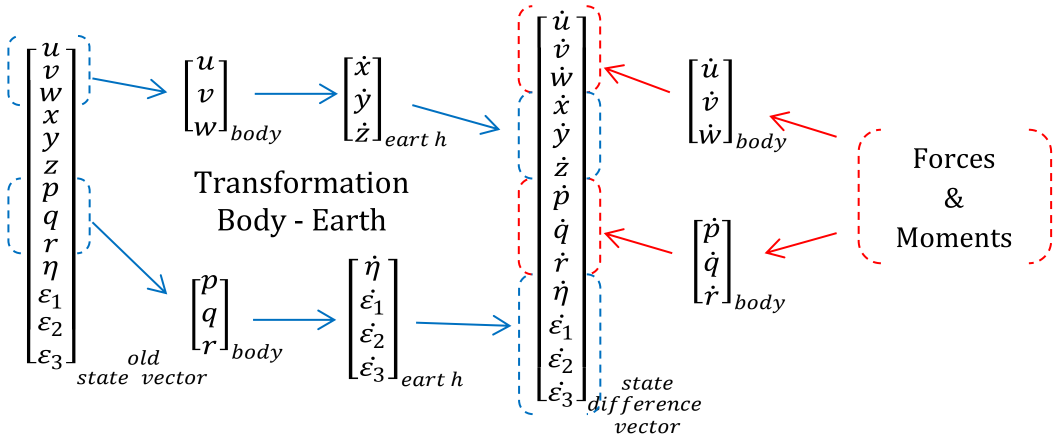

Figure 4.7 – Illustration of the state propagation; creating the state difference vector.

$$\underbrace{\begin{bmatrix} u \\ v \\ w \\ x \\ y \\ z \\ p \\ q \\ r \\ \eta \\ \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_3 \end{bmatrix}}_{\substack{new \\ state\ \ vector}} = \underbrace{\begin{bmatrix} u \\ v \\ w \\ x \\ y \\ z \\ p \\ q \\ r \\ \eta \\ \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_3 \end{bmatrix}}_{\substack{old \\ state\ \ vector}} + \int_{t_{old}}^{t_{new}} \underbrace{\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \\ \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{p} \\ \dot{q} \\ \dot{r} \\ \dot{\eta} \\ \dot{\varepsilon}_1 \\ \dot{\varepsilon}_2 \\ \dot{\varepsilon}_3 \end{bmatrix}}_{\substack{state \\ difference \\ vector}} dt$$

Figure 4.8 – Illustration of the state propagation; state integration.

5 Existing Motion Simulators in MATLAB

There are several motion simulators already made in MATLAB for simulating flight, though often with limited accessibility and documentation. Not surprisingly, the ones that are accessible are not perfectly designed for this particular application. As a consequence, a new simulator is needed, custom-made for the company's Deep Green device. However, existing simulators can be very helpful to compare with when designing a new simulator.

Since the Deep Green device is a flying rigid body in water it would be pleasant to study simulators for e.g. airplanes and underwater vehicles. The most efficient way to go is then to extract the best out of the two different worlds; air flight and under water maneuvering.

5.1 Stengel's flight simulator

At the very beginning of the project, one of the supervisors at Minesto found out about Professor Robert F. Stengel and his book Flight Dynamics, together with his MATLAB flight simulator to simulate a business jet aircraft flying in the atmosphere. The book was purchased for this project and the simulator was downloaded free to use from Stengel's website [8].

Stengel's six-degrees-of-freedom flight simulator is in some senses very complex, containing many difficult features, especially regarding maneuvering. Both the initiation and the control system are very thoroughly designed, maybe even too complicated considering readability for the user. Unfortunately, the Deep Green device only has a rudder for maneuvering, making Stengel's control system overkill for the application.

In other senses the simulator is very simplified; especially regarding the flight envelope since the aircraft is limited not to fly straight up or straight down due to singularities in the angle representation. This does not concern the simulation of a business jet since it is not supposed to fly this way, though it is a great concern for the Deep Green device that should be able to move freely in all directions.

The documentation of Stengel's simulator is excellent, not only as comments in the code and on the website, there is also a lot explained in the book Flight Dynamics. The book is well written and contains almost everything there is to know about flying.

5.2 Fossen's marine system simulator

The flight simulator made by Stengel lacks two major features essential for the new simulator; effects due to added mass since we are working in water, and quaternions to avoid singularities in the angle representation. Professor Thor I. Fossen was found to have excessive knowledge about both features needed. When contacted he e-mailed plenty of valuable literature from his own publications, some also accessible at his website [9]. Additional to the literature Fossen also linked to a marine system simulator he developed with his colleagues, accessible and free to use at [10].

Like Stengel’s simulator, Fossen’s marine system simulator is a six-degrees-of-freedom simulator, but designed for marine applications instead of air flight. It is truly advanced, handling seakeeping theory, fluid memory effects and nonlinear maneuvering. It is made for simulating different kinds of vessels, e.g. container ships, tankers and underwater vehicles. There are a handful of vessel models available, together with the ability to simulate different wave climates. Most of the features are abundant for the new simulator, though methods for handling both added mass and quaternions are implemented and nice to review.

It may seem unnecessary to design a new simulator when Fossen already made one for underwater vehicles. However, the code structure is not favorable for the HAMS project regarding the interaction with both the hydrodynamic analysis program and the control system. Instead the best features are studied, influencing the design of the new simulator.

6 Design & Implementation in MATLAB & Simulink

Engineering is not about inventing the wheel again but to use the knowledge from earlier experiences in a new perspective for other applications. This is a typical case where the work done by Stengel and Fossen is revised and partly reused for a completely different application, i.e. power generation simulation. Despite the existing material made by the two, the majority of the implementation was made from scratch on a blank paper since they lack the most of the features needed to simulate the Deep Green device.

As stated in section 5.1, the flight simulator made by Stengel was obtained in the very beginning of the project. Consequently, this was used as a starting platform for the simulation development of the new simulator, from now on referred to as *SimFlight*. The code structure in Stengel's simulator is simple and fairly effective which influenced the early design of the code structure in SimFlight. However, the two simulators are no longer alike. SimFlight is faster, has no restrictions in the flight envelope, initiates relative to the flow and has ha greater usability for product development purposes since it is built at a higher abstract level. As Figure 6.1 shows, it is built with separated modules for flow models, control systems and the actual motion equations. This enables possibilities to develop the modules separately and to easily switch between different models within the modules.

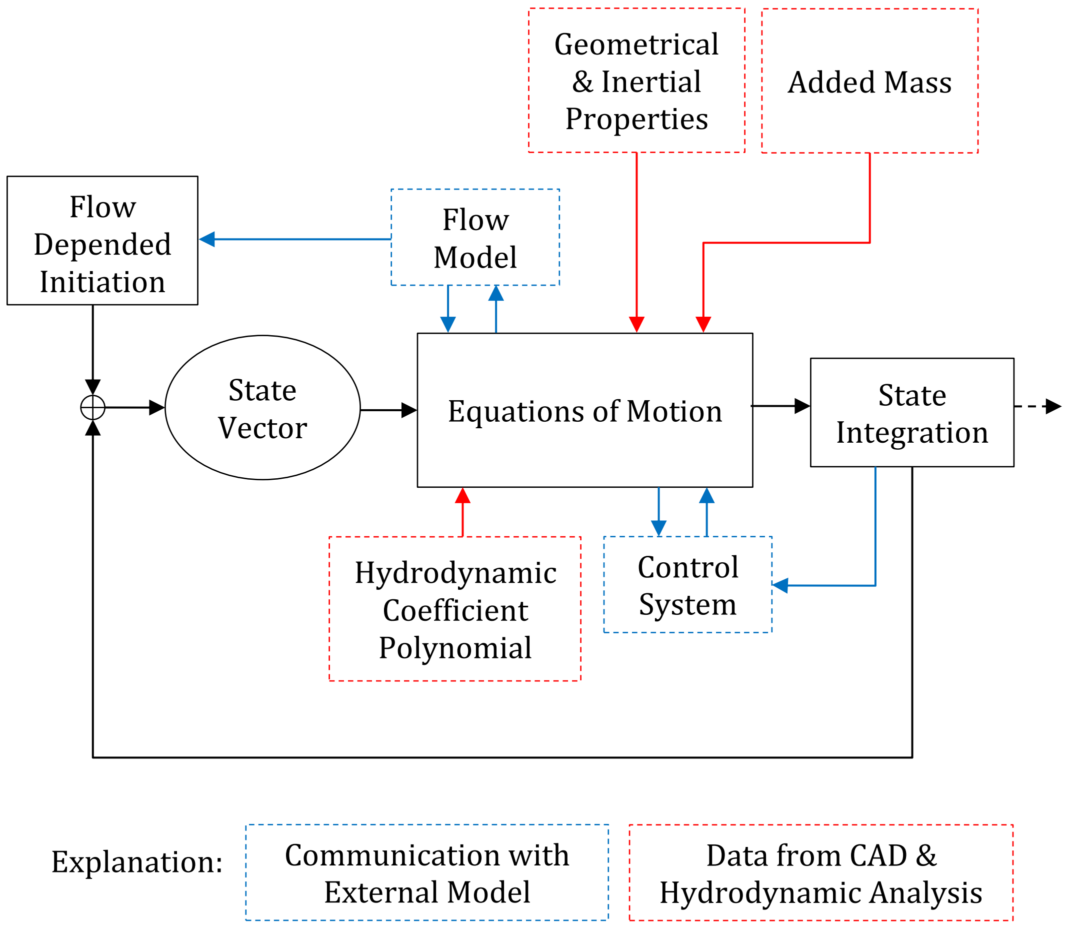
Geometrical & Inertial Properties
Added Mass
Flow Depended Initiation
Flow Model
State Vector
Equations of Motion
State Integration
Hydrodynamic Coefficient Polynomial
Control System
Explanation:
Communication with External Model
Data from CAD & Hydrodynamic Analysis

Figure 6.1 – Flowchart showing the structure of SimFlight.

The simulation initiation is a sensitive procedure and is automatically set by the current flow. The main function of the SimFlight simulator is the Equations of Motion module, from here all the different models are called and the state accelerations are calculated. The state is then updated in the State Integration stage and rerouted, creating a loop. The control system steers the device by controlling the rudder of the device. The details about the implementation are described in the subsections below.

6.1 Approximations

There are several, more or less accurate, approximations made in the implementation. As mentioned in section 4.3, the following approximations are made when deriving the governing equations:

- Flat earth assumption
- The device is a rigid body
- No change in mass or inertia
- The angular momentum of the turbine is neglected

The following assumptions and simplifications are made for making the implementation easier:

- The body is assumed to rotate around its center of mass
- The pitch and roll damping moment are artificial
- The tether is modeled as a damped spring; rigid and without drag

The approximations made are discussed in section 9.1.

6.2 Communications and data transfer

As already explained in the introduction, the SimFlight simulator is a part of the HAMS project that communicates with both a hydrodynamic analysis program and a CAD program. In the future these communications has possibilities to be automatic but are to begin with manually controlled.

All geometrical and inertial properties of a design are manually imported from the CAD program and stored in a function file only containing design specific data that does not change with time, allowing swapping between designs by easily swapping function files. The added mass matrix is calculated once in the hydrodynamic analysis program and also manually added to the function file. Whenever the properties are needed the function is called.

The hydrodynamic analysis program calculates the force coefficients corresponding to all flight conditions possible within the flight envelope specified. The communication procedure with the simulator is explained in section 6.4.2.

6.3 Flow model module

In Stengel’s simulator it is the thrust from the aircraft’s engines that is the driving factor, in the SimFlight simulator it is only the flow that makes the device move. The flow model is able to handle a variable 2D flow, defined by the user. The variability is important since a tidal flow is site specific and varies both in speed and direction over time.

The user sets two parameters to define the flow; flow speed and flow direction. The flow speed can typically be set as a sinusoidal with some amplitude, switching between for example +/- one meter per second to characterize a tidal flow. The flow direction is set by an always positive angle between 0-360 degrees in the inertial frame of reference, positioned at the swivel; 0 degrees corresponding to north and 270 degrees to west. Figure 6.2 exemplifies the flow direction angle of 210 degrees on a sea chart. It is not rare that the direction of a tidal flow is not just its inverse when the tide has switched, therefore the flow direction is set by two angles, depending on the sign of the flow speed.

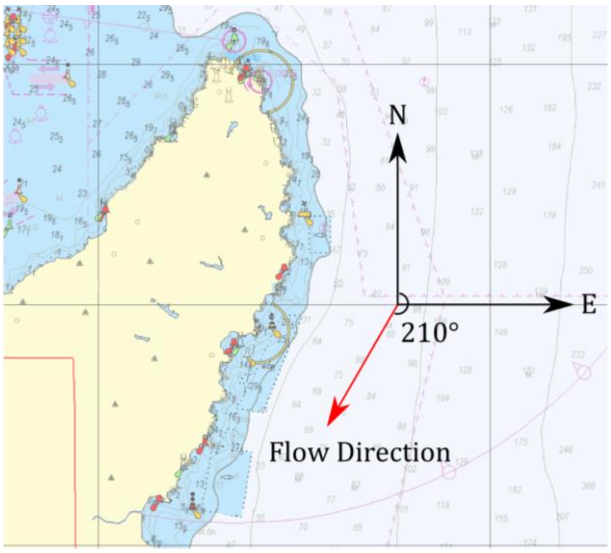

Figure 6.2 – Exemplification of the flow direction angle.

The flow model parameters are transformed to a flow vector in the motion simulator, expressed in both the inertial and the body-fixed frame of reference. This enables the flow model to be developed separately, as long as a flow vector can be created in the motion simulator.

The Oceanographer Martin Johnsson at Minesto explains that the water density can vary at a site due to the difference in salt concentrations at different depths, which can affect the characteristics of the flow. This is not considered in the thesis.

6.3.1 Flow dependent initiation procedure

The initiation is the most critical part of the simulation, it is extremely sensitive and therefore thoroughly worked through to be automatically set depending on the flow, with as little user interaction as possible. The only thing the user sets is at what depth the device is to be initiated at, the rest is automatic. The device is initiated downstream the flow at the desired depth, heading to the right when looking at it from the swivel, visualized in Figure 6.3.

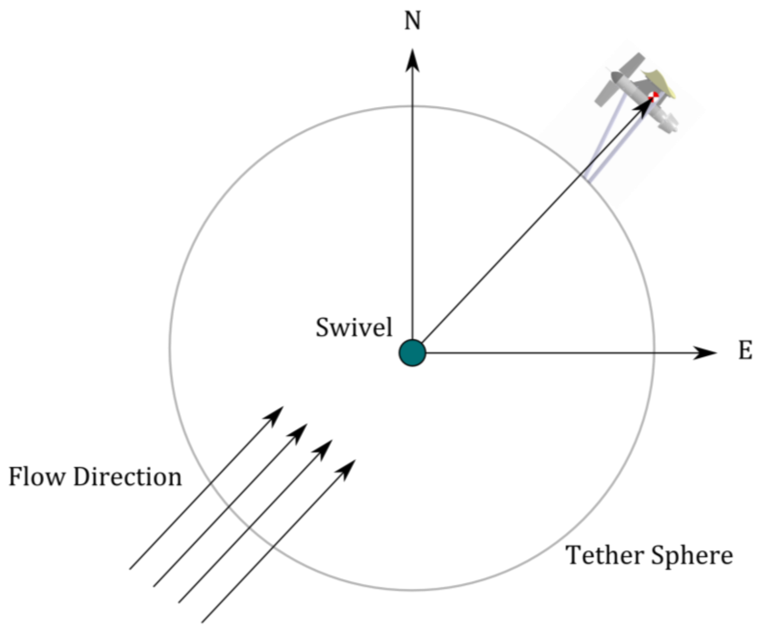

Figure 6.3 – Initiation downstream, heading to the right.

In order to position the device depending on the flow, a standard spherical coordinate system is defined with its origo correlating with the inertial frame of reference', but with its z-axis pointing upwards and the y-axis consequently pointing in the opposite direction . As shown in Figure 6.4, a position on the surface of a sphere is then defined by the radius of the sphere r_{sphere} together with two angles; polar angle θ_{sphere} and azimuth angle φ_{sphere}.

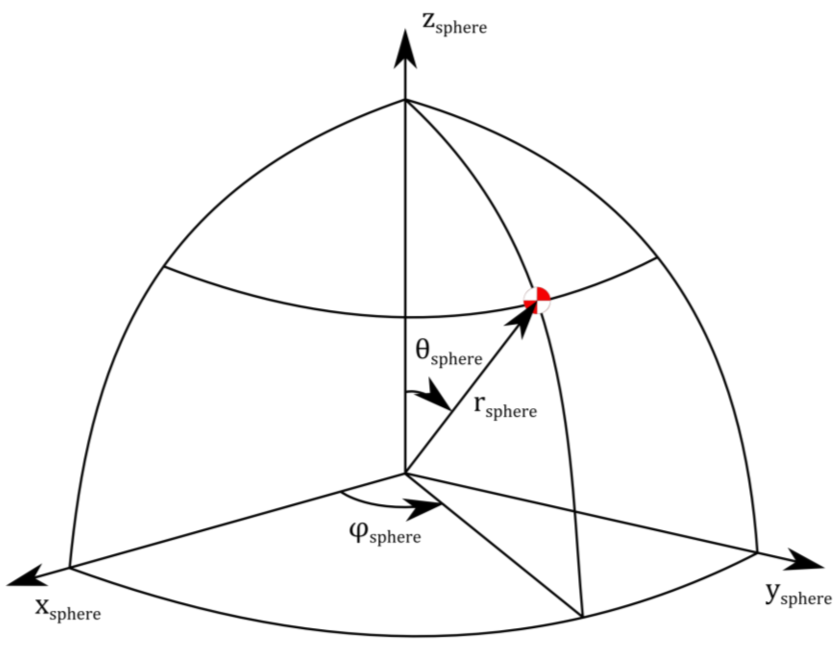

Figure 6.4 – Position of the device in a spherical coordinate system.

The radius is in this case the distance to the mass center of the device when the tether is stretched and the body z-axis is pointing in the direction of the tether. The polar angle is in this case defined by the arccosine of the altitude above sea bottom and the radius, the azimuth angle is the inverse of the flow direction angle since it is expressed in an inverted coordinate system.

The Cartesian representation of the position vector in the spherical coordinate system, $\boldsymbol{r}_{sphere}$, can now be retrieved from the spherical components as simple projections shown in (6.1).

$$\boldsymbol{r}_{sphere} = \begin{bmatrix} x_{sphere} \\ y_{sphere} \\ z_{sphere} \end{bmatrix} = \begin{bmatrix} r_{sphere}\cos(\theta_{sphere})\sin(\varphi_{sphere}) \\ r_{sphere}\sin(\theta_{sphere})\sin(\varphi_{sphere}) \\ r_{sphere}\cos(\varphi_{sphere}) \end{bmatrix} \quad \textbf{(6.1)}$$

To get the position vector represented in the inertial earth frame of reference it is transformed with an orthonormal transformation matrix that describes the difference between the two coordinate systems; both the y- and the z-axis are inverted. The transformation of the position vector therefore becomes as shown in (6.2).

$$\boldsymbol{r}_I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix} \boldsymbol{r}_{sphere} \quad \textbf{(6.2)}$$

Additional to the starting position the device needs to be correctly orientated. To be able to define the correct angular attitude one needs to know in what sequence the angles are to be executed, thoroughly explained in section 4.1.1. In SimFlight the sequence is chosen to be yaw, pitch, roll and the angles are set accordingly. The yaw angle depends on the flow direction angle and the fact that it is supposed to start heading to the right relative to origo. The pitch angle is not required to differ from zero regarding the heading, but it is set to three degrees to get an angle of attack that causes the device to accelerate. This is by experience a favorable approach, making the initiation more stable. The roll angle depends on the starting altitude above the sea bottom and in what direction it is supposed to start. It is the same as the polar angle but with a negative sign due to the fact that it is supposed to head to the right, regulated by the sequence chosen. Lastly, the Euler angles are transformed in to quaternions using a transformation function copied from Fossen’s marine system simulator that simply uses the theory described in section 4.1.2.

6.4 Equations of motion module

Figure 6.5 is a further description of the Equations of Motion module in Figure 6.1; it is here where the magic happens. Additional to the equations of motion, the module contains several independent models as subsystems for handling different phenomena, more or less fundamental. The implementation of all of them is explained in the subsections below.

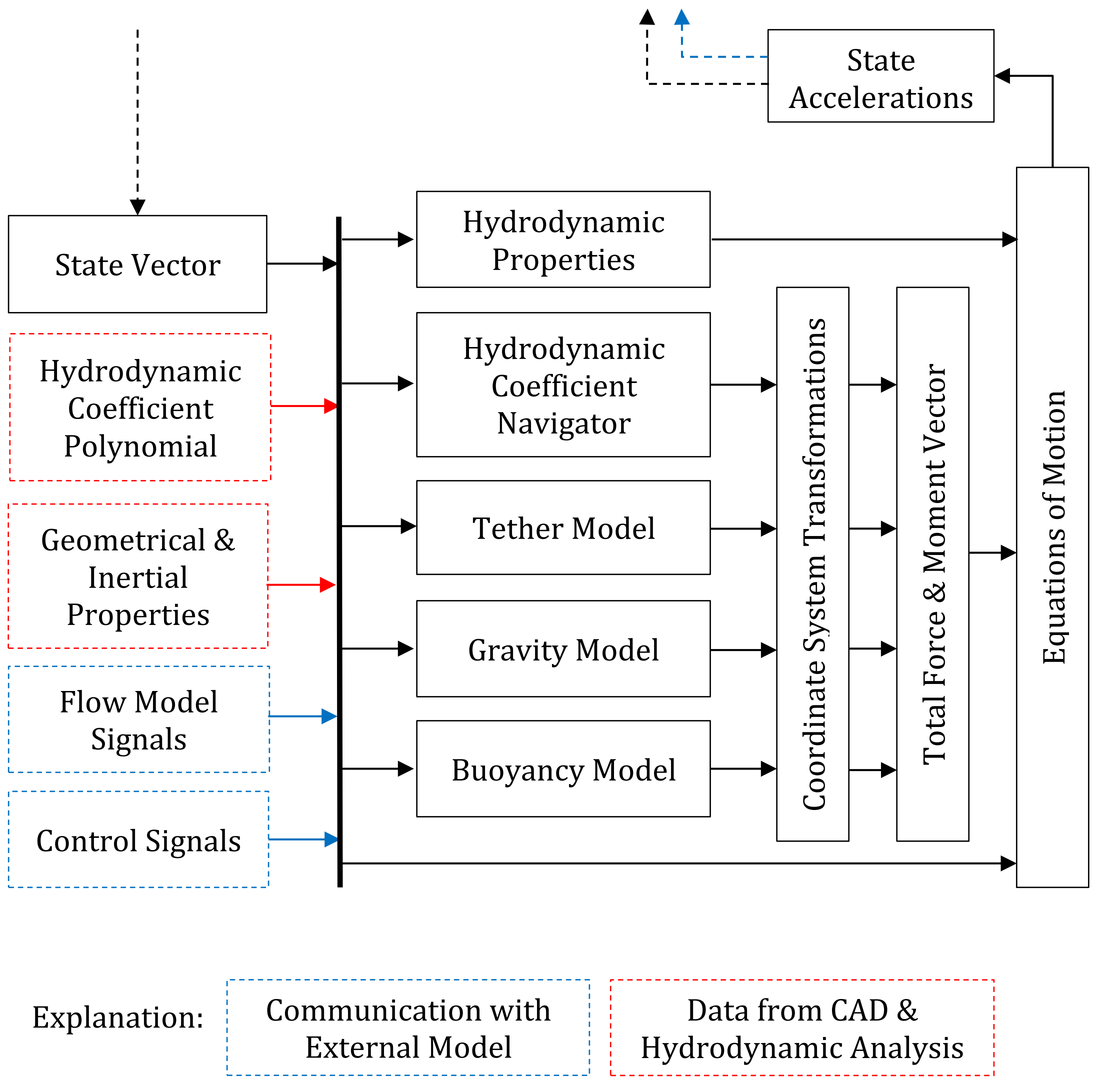
State Accelerations
State Vector
Hydrodynamic Properties
Hydrodynamic Coefficient Polynomial
Hydrodynamic Coefficient Navigator
Geometrical & Inertial Properties
Tether Model
Gravity Model
Flow Model Signals
Buoyancy Model
Control Signals
Coordinate System Transformations
Total Force & Moment Vector
Equations of Motion
Explanation:
Communication with External Model
Data from CAD & Hydrodynamic Analysis

Figure 6.5 – Equations of Motion module.

6.4.1 Hydrodynamic properties

Additional to the thirteen states in the state vector $\boldsymbol{x}$ in (4.12) there are some other parameters of interest for the simulation that are calculated in this module; i.e. angle of attack, side slip angle, flow relative velocity and dynamic pressure. All of the parameters are defined in section 3.2, but where the free stream velocity in the dynamic pressure relation is replaced by the flow relative velocity.

All of these parameters are in some way used for the calculation of the hydrodynamic forces, but the angle of attack and the side slip angle are also great markers to display the current flight conditions. These are therefore plotted in the simulation together with the speed and the position history to visualize the result of the simulation.

When calculating for instance the side slip angle one needs to be alert to avoid numerical singularities since it is possible to accidently divide by zero when the side slip angle goes to 90 degrees.

6.4.2 Hydrodynamic coefficient navigator

As earlier stated the hydrodynamic analysis calculates the force coefficients for a specified flight envelope and saves the results in a huge matrix. To communicate with the simulator a *Response Surface Methodology* is chosen due to the nature of the data and the simplicity of the calculation procedure.

Six separate multidimensional polynomials are created to fit the results from the analysis for all forces and moments with the five condition parameters used in the analysis program to specify the hydrodynamics; angle of attack, side slip angle, flow relative velocity, yaw rate and rudder angle of attack. The order of every condition parameters is specified when making the polynomial, all of which they differ between first, second and third order. The order depends on how well the data fits the polynomial; if a parameter of first order does not fit enough the order is increased to second etc. In the end the polynomial is well defined, within the range of the data.

With a well-defined polynomial the force coefficients can be explicitly calculated by simply input the current condition parameters. A schematic example of a polynomial is presented in (6.3) where P_i represents a force coefficient with index i as a function of the condition parameters α and β together with the corresponding constants a and b, all with different orders n and m.

$$P_i = a_i^{n_{i,\alpha}} \alpha^{m_{i,\alpha}} + b_i^{n_{i,\beta}} \beta^{m_{i,\beta}} \qquad \textbf{(6.3)}$$

Since the hydrodynamic analysis program is using a different body-fixed coordinate system the condition parameters from the simulation needs to be transformed before they are used in the polynomial. The coordinate system is rotated 180 degrees around the y-axis compared to the coordinate system used in the simulator, e.g. the yaw rate in the polynomial is defined in the opposite direction.

The control system controls the rudder angle but it is in fact the rudder angle of attack that is of interest. This is what gives a lift on the rudder and the ability to steer the device. The rudder angle of attack has to be calculated before it is put in the polynomial and it is a function of five parameters; rudder angle, distance between the rudder axis and the mass center, yaw rate, flow relative speed and side slip angle. Caution must be taken since the function can cause numerical issues due to singularities when dividing with parameters going to zero, therefore the calculation procedure is divided by a couple of if-statements to prevent this from happening. Once again, the parameters used in this calculation needs to be transformed to the coordinate system that corresponds to the polynomial.

The ideal would be to include every parameter in the hydrodynamic analysis that effects the simulation, but as always, everything comes with a price. Hence, both the damping effect from pitch and roll rate are excluded from the polynomial to save computation time and are instead handled with an artificial damping term. It is proven to work fairly well, but they are to be added in the analysis later on to decrease the number of artificial features.

When the hydrodynamic force coefficients are calculated from the polynomial they are represented in the coordinate system corresponding to the hydrodynamic analysis program and therefore have to be transformed to the coordinate system used in the simulator. This is a simple orthonormal transformation since the coordinate systems only are rotated 180 degrees around the y-axis relative to each other, although the whole transformation procedure in this model was difficult to generate.

Lastly, the hydrodynamic forces and moments acting on the device are calculated as explained in section 3.2.

6.4.3 Tether model

The Deep Green device is attached to the ocean floor with a rather long tether that is not completely stiff, therefore it is modelled as a spring. The characteristics of the tether are comparable with a string; it stretches when pulled and collapses when compressed. Due to classical mechanics in [11], the force vector $\boldsymbol{T}$ of the tether is described in (6.4) where k is the spring constant, $|\boldsymbol{R}|$ is the momentary tether vector length, R_0 is the nominal tether length and $\boldsymbol{e}_R$ is the unit vector of the tether.

$$\boldsymbol{T} = \begin{cases} -k(|\boldsymbol{R}| - R_0)\boldsymbol{e}_R & , |\boldsymbol{R}| \geq R_0 \\ 0 & , |\boldsymbol{R}| < R_0 \end{cases} \tag{6.4}$$

When simulating a system with different dynamics like this, one needs to think about the timescales of the dynamics. The dynamics of the Deep Green device is fairly slow and the dynamics of the tether is very fast, this makes the system unstable and implies very small time steps to be solved properly. To make the system more stable one can modify the expression for the tether force by adding a damping term; (6.4) is then modified to (6.5) where c is the damping constant and $|\dot{\boldsymbol{R}}|$ is the change of the tether length from the last time step. This is more or less true compared to the real system, and it makes the simulation possible without the infinitely small time steps used in reality. The spring constant is set due to test data for the tether and the damping constant is for simplicity set to the same value.

$$\boldsymbol{T} = \begin{cases} -\left(k(|\boldsymbol{R}| - R_0) + c|\dot{\boldsymbol{R}}|\right)\boldsymbol{e}_R & , |\boldsymbol{R}| \geq R_0 \\ 0 & , |\boldsymbol{R}| < R_0 \end{cases} \tag{6.5}$$

The tether is anchored to the device in a point where supporting struts are intersecting and the reaction force from the tether is consequently distributed to the struts. The device is designed to always strive to stay straight, with its body z-axis in the tether direction. Since the anchor point of the tether is located with a distance to the center of mass, the tether always has an active moment arm when the device is tilted. If the device is experiencing a rotating moment due to the flow, depending on the angle of the device the tether is acting in the contrary direction to stabilize it. In classical mechanics [11] the moment vector due to an acting force is calculated with a cross product between the moment arm vector and the force vector. The tether moment vector $\boldsymbol{M}_T$ is described in (6.6) where $\boldsymbol{a}$ is the moment arm vector and $\boldsymbol{T}$ is the force vector.

$$\boldsymbol{M}_T = \boldsymbol{a} \times \boldsymbol{T} \tag{6.6}$$

The built in cross product function in MATLAB is inefficient, taking too much computational capacity, and is therefore replaced by a cross product equivalent matrix, described in the footnote on page 19. This changes the cross product in (6.6) to a much more efficient matrix multiplication in (6.7). All cross products in the simulator is replaced by a cross product equivalent matrix multiplication.

$$\boldsymbol{M}_T = \boldsymbol{S}(\boldsymbol{a})\boldsymbol{T} = \begin{bmatrix} 0 & -\boldsymbol{a}(3) & \boldsymbol{a}(2) \\ \boldsymbol{a}(3) & 0 & -\boldsymbol{a}(1) \\ -\boldsymbol{a}(2) & \boldsymbol{a}(1) & 0 \end{bmatrix} \boldsymbol{T} \tag{6.7}$$

The relation may look simple but it is not rare that the calculation of the moment arm needs some linear algebra work to be found. Figure 6.6 describes the vector calculation of the moment arm vector $\boldsymbol{a}$, the vector perpendicular to the force, starting in the mass center and ending at the extended line of the force. The vector $\boldsymbol{R}$ is the tether vector and $\boldsymbol{e}_r$ is its unit vector, $\boldsymbol{r}$ is the mass center vector, $\boldsymbol{d}$ is the distance vector between the mass center and the anchor point of the tether and δ is the angle between the two vectors $\boldsymbol{r}$ and $\boldsymbol{R}$.

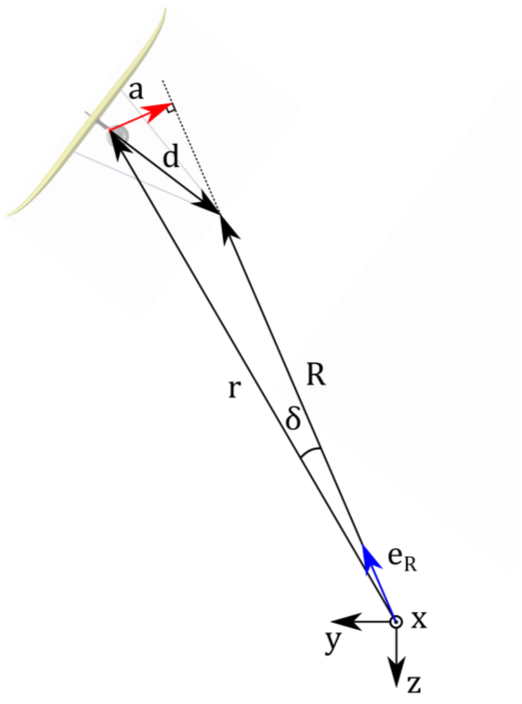

Figure 6.6 – Deriving the moment arm vector $\boldsymbol{a}$.

The only entities explicitly known are the mass center vector $\boldsymbol{r}$ and the distance vector $\boldsymbol{d}$, hence the other ones needs to be calculated. The tether vector $\boldsymbol{R}$ is simply an addition of the mass center vector $\boldsymbol{r}$ and the distance vector $\boldsymbol{d}$ in (6.8).

$$\boldsymbol{R} = \boldsymbol{r} + \boldsymbol{d} \tag{6.8}$$

The angle δ is formulated in (6.9) by the definition of the dot product [12] and is needed to be able to orthogonally project the mass center vector on the tether vector.

$$\delta = \arccos\left(\frac{\boldsymbol{r} \cdot \boldsymbol{R}}{|\boldsymbol{r}||\boldsymbol{R}|}\right) \tag{6.9}$$

Imagine the vector $\boldsymbol{r}_R$ to be the orthogonal scalar projection of the mass center vector $\boldsymbol{r}$ on the tether vector $\boldsymbol{R}$, expressed in (6.10).

$$\boldsymbol{r}_R = |\boldsymbol{r}| \cos(\delta) \boldsymbol{e}_R \tag{6.10}$$

Then the moment arm vector $\boldsymbol{a}$ is simply a vector subtraction between the vector $\boldsymbol{r}_R$ and the mass center vector $\boldsymbol{r}$ in (6.11).

$$\boldsymbol{a} = \boldsymbol{r}_R - \boldsymbol{r} \tag{6.11}$$

6.4.4 Gravity model

As explained in section 4.1, the simulated device is represented with the position of its mass center in an inertial frame of reference together with a body-fixed reference frame in its mass center to describe its angular state. Therefore it is easy to implement the effect of gravity since it only is a mass depending force pointing downwards in the inertial frame of reference, acting in the center of mass. Hence, there is no moment in the center of mass due to gravity.

6.4.5 Buoyancy model

In contrary to gravity, the buoyancy does not depend on the mass but on the volume of the device. The buoyancy force is, due to *Archimedes' principle* [13], always pointing in the opposite direction to gravity, but with a magnitude depending on the amount of water the device displaces and acting in the buoyancy point instead of in the center of mass. This makes the buoyancy force to give rise to a moment around the mass center due to the difference between the buoyancy force point and the center of mass. Similar to the tether model described in section 6.4.3, the moment is a simple cross product between the moment arm vector and the buoyancy force vector; implemented with a cross product equivalent matrix multiplication instead of a cross product.

The buoyancy force magnitude and the buoyancy point where it acts are calculated using data from a volume model of the device in the CAD program, giving the center of volume and the total volume of the device. The center of volume is where the buoyancy force will act and the magnitude of the buoyancy force is the device' volume times the density of water. Initially the buoyancy magnitude is set to gravity to facilitate the simulation, making the device weightless. Since the real system has a changeable buoyancy control system, this is not too far from reality. The buoyancy control system is excluded in the HAMS project.

Once again, the tough nut to crack is not the mathematical relation for the moment vector, but to find the moment arm vector $\boldsymbol{n}$ in Figure 6.7. The buoyancy force vector $\boldsymbol{F}_{buoy}$ and the gravity force vector $\boldsymbol{mg}$ will always be parallel and along the inertial z-axis due to the nature of the forces. This means in turn that the moment arm vector $\boldsymbol{n}$ perpendicular to the forces will always be a vector in the x/y-plane. Consequently, the moment arm vector $\boldsymbol{n}$ is the orthogonal projection of the distance vector $\boldsymbol{s}$ on the x/y-plane as in (6.12); the orthogonal projection matrix $\boldsymbol{P}$ is shown in (6.13) and the distance vector $\boldsymbol{s}$ characterizes the distance between the mass center and center of volume.

$$\boldsymbol{n} = \boldsymbol{P} \cdot \boldsymbol{s} \tag{6.12}$$

$$\boldsymbol{P} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{6.13}$$

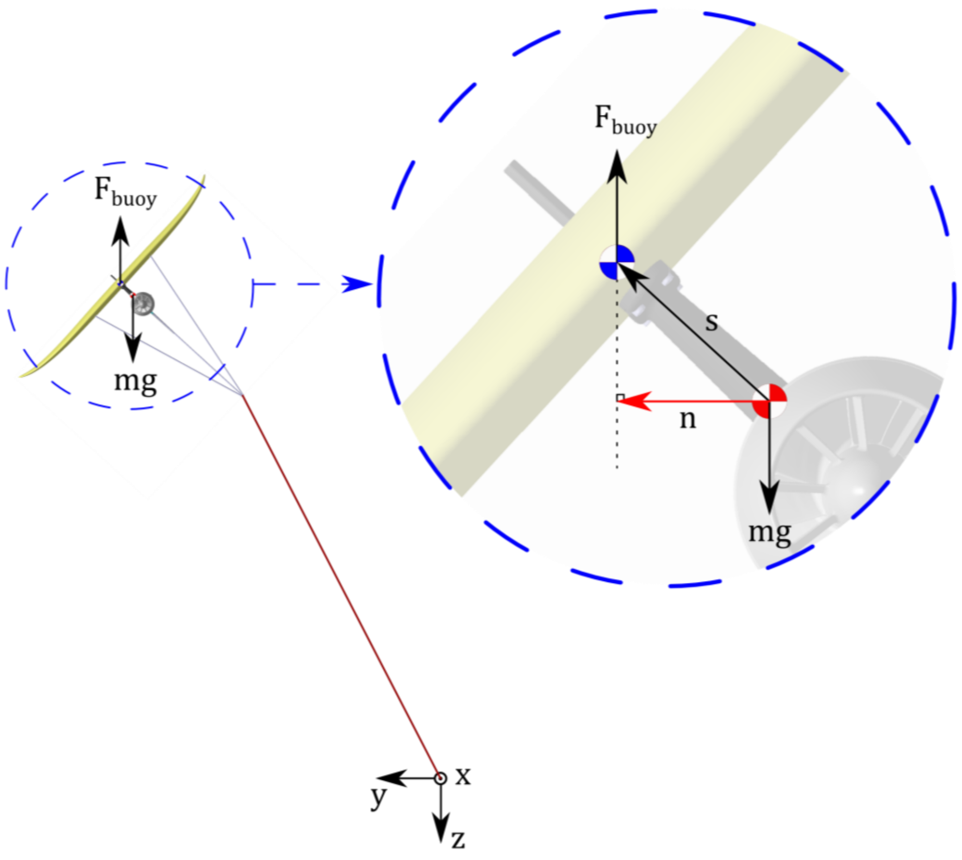

Figure 6.7 – Deriving the moment arm vector $\boldsymbol{n}$.

6.4.6 Coordinate system transformations

All forces and moments used in the equations of motion are expressed in a body-fixed frame of reference, though often calculated in the inertial earth frame of reference and later transformed. The transformation procedure is thoroughly described in section 4.1, using the different transformation matrices defined by the current quaternions.

6.4.7 Equations of motion and state accelerations

Chapter 4 thoroughly describes the equations of motion used in the simulator and the implementation does not come with any surprises. The total force and moment vector in Figure 6.9 is the sum of all forces in Figure 6.8, calculated in respective models, and used to calculate the state accelerations.

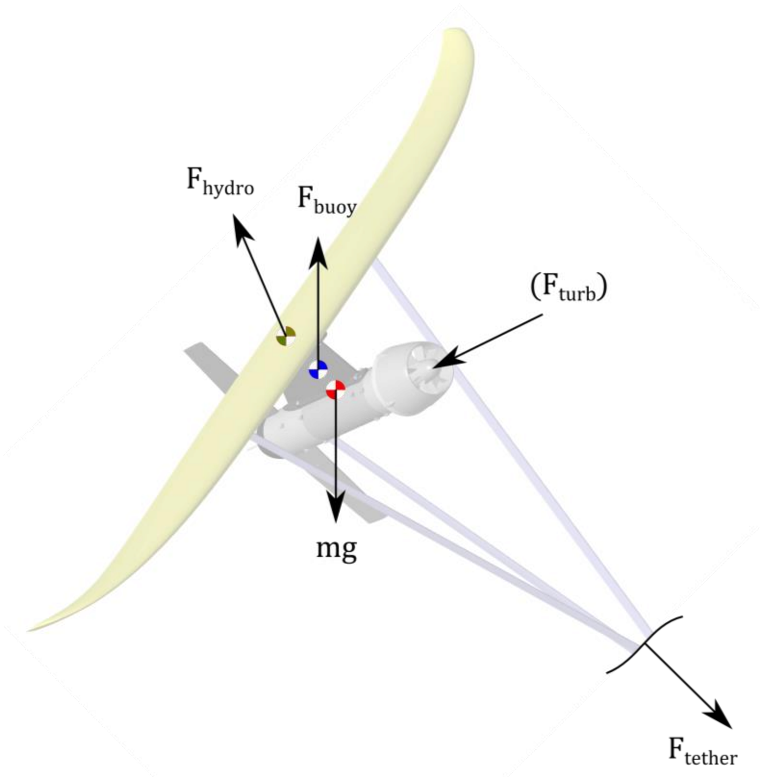

Figure 6.8 – Visualization of all forces.

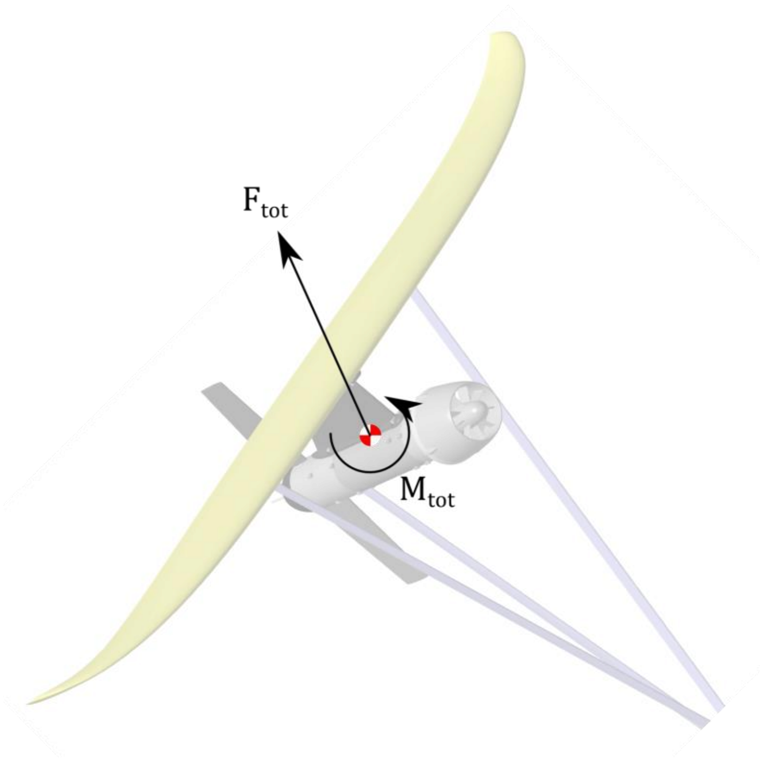

Figure 6.9 – The resultant force and moment.

As stated in section 4.1.2, the quaternions are implemented to open up the entire flight envelope without the risk of singularities in the angular representation. Not only the theory behind the quaternions is gathered from Fossen, but also the implementation of them in MATLAB is partly copied from his marine system simulator described in section 5.2.

6.5 State integration

The theory behind the state propagation is described in section 4.3, but the state integration in the simulator is made discrete instead of continuous. The state is updated for every time step dt with the simple relation in (6.14), where $\boldsymbol{x}_{new}$ is the new state, $\boldsymbol{x}_{old}$ is the old state and $\dot{\boldsymbol{x}}$ is the state difference from the previous time step.

$$\boldsymbol{x}_{new} = \boldsymbol{x}_{old} + \dot{\boldsymbol{x}}dt \tag{6.14}$$

6.6 Control system integration

The aim of the control system integration within the HAMS project is to get the simulation steered in an infinity loop, a horizontal eight. The control system to steer the Deep Green device is developed by Minesto and implemented in Simulink. In order to communicate with the control system, SimFlight is implemented in Simulink and connectivity to the control system is established. Only a small part of the control system is implemented since the simulation is a perfect world with faultless sensor signals and therefore only a fraction of the sensor signals is needed for steering. This also minimizes the sources of errors in the integration. In the future the SimFlight simulator is supposed to be used for control system development and the total control system will be integrated.

The control system is confidential and the integration implementation is therefore excluded from the public report.

7 Verification

The development of a new motion simulator is an iterative process where the performance of every change is carefully studied. The SimFlight simulator is built around three different geometrical designs and with several stages of verification during the development process, even though the finalized simulator design has not yet been completely verified. Some of the milestones for the verification are presented below.

7.1 Proof of concept

The concept of simulating a flying rigid body was proofed early in the process when simply flying using known theoretical relations for lift and drag as functions of angle of attack, together with artificial hydrodynamic moments.

7.2 Reliability of the hydrodynamic analysis

The first geometrical design used in the HAMS project is an early prototype that is tested a lot in test facilities. The test results were used to verify the hydrodynamic analysis and the results were qualitatively satisfactory. This verified the calculations and enabled the HAMS project to continue to the next step. An exported picture from the hydrodynamic analysis program of the prototype is shown in Figure 7.1.

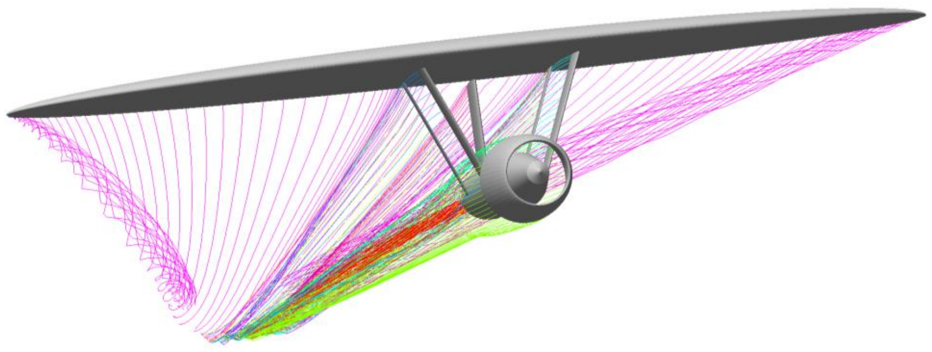

Figure 7.1 – Exported picture from the hydrodynamic analysis program of an early prototype.

7.3 Correct communication with the hydrodynamic analysis

To be sure that the results from hydrodynamic analysis were properly communicated to the simulator, with correct polynomial and coordinate system transformations, a test case with explicitly known behavior was set up. The “flying square” in Figure 7.2, with four assembled symmetrical NACA 0012 wing profiles, was chosen to be the best candidate after a proper evaluation process with several criteria to be fulfilled. Additional to that the behavior of it is explicitly known, it was easy to design in the CAD program and it was easy to get good results from the hydrodynamic analysis, without too much work.

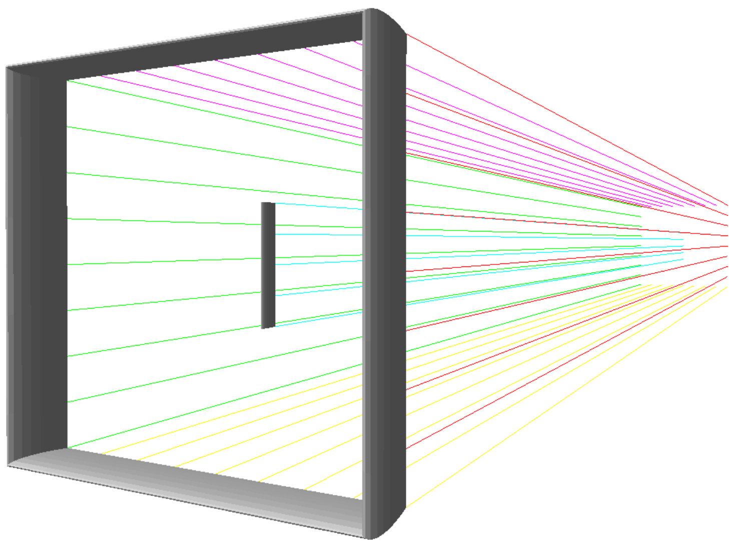

Figure 7.2 – The ”flying square” consisting of four NACA 0012 wing profiles, together with a smaller rudder of the same wing profile geometry.

The forces and moments in all directions were plotted and studied for every condition parameter in the polynomial; exemplified for the driving force in the body x-direction due to the angle of attack and the side slip angle in Figure 7.3. All that differed from the expected behavior was thoroughly investigated, e.g. inverted forces, moments acting in the wrong direction etc. In the end a truly logical transformation procedure was set up; its simplicity verifying its perfection. The “flying square” was easily simulated, but a bit difficult to steer since the rudder was undersized.

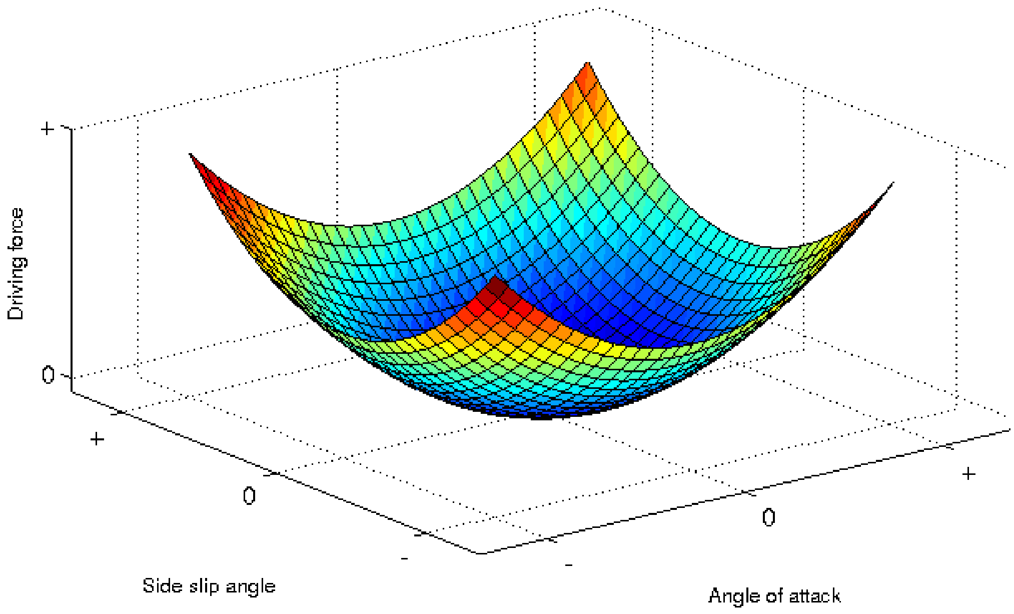

Figure 7.3 – The driving force in the body x-direction of the “flying square” as a function of the angle of attack and the side slip angle.

Figure 7.4 shows a history plot of the position that proofs that the information transfer is successful. The device is initiated downstream a sinusoidal flow, follows the flow during slack water and restarts itself when the flow gets strong enough in the other direction. The rudder is set to a constant angle of 20 degrees.

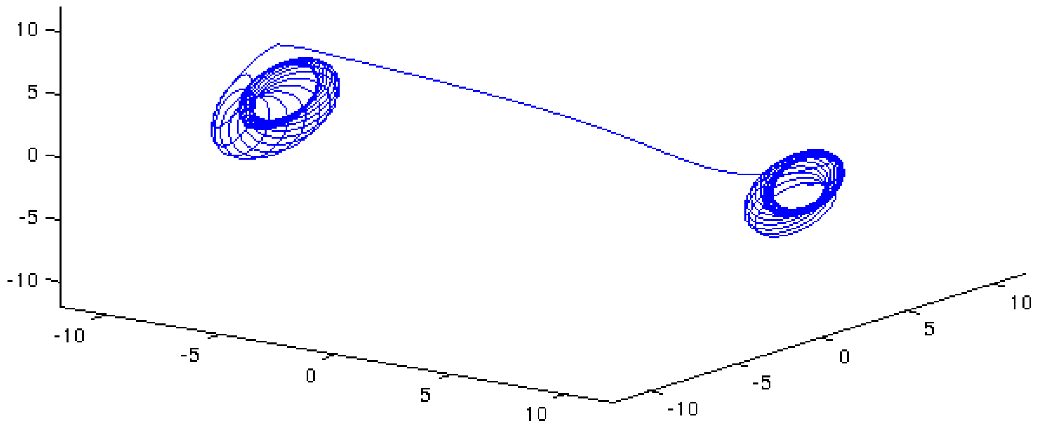

Figure 7.4 – History plot of the position for the "flying square". The flow is sinusoidal; it starts at maximum amplitude, goes to zero and later to maximum amplitude in the opposite direction. Constant rudder angle at 20 degrees.

7.4 Successful implementation of quaternions

Initially the simulation was built around Euler angles and worked satisfactory. Due to the risk of singularities in the Euler representation, quaternions were implemented instead. The simulation results before and after the implementation showed no difference.

7.5 Finalized simulator design

As mentioned, the finalized simulator design has not yet been completely verified. This has not been possible within the frame of the thesis. Physical tests are made with the latest prototype outside Northern Ireland at the time of writing and data is continuously collected suitable for simulation verification purposes. Even though it has not been completely verified, the simulation results look promising. For instance, the simulations of the "flying square" in section 7.3 do not only verify the communication, they show an expected behavior.

8 Simulations

The objectives of the master's thesis project are to develop a motion simulator for the Deep Green device that initiates downstream a variable flow and is controlled by the Minesto Control System to follow the trajectory of an infinity loop. Figure 8.1 – Figure 8.8 reveals that the objectives are reached.

Figure 8.1 and Figure 8.2 shows a history plot of the position of a simulation, steered in an infinity loop by the control system. A reference sphere is added to make the result more visible, together with two reference planes representing the seafloor and the surface of the sea. The reference sphere is the sphere spanned by the tether.

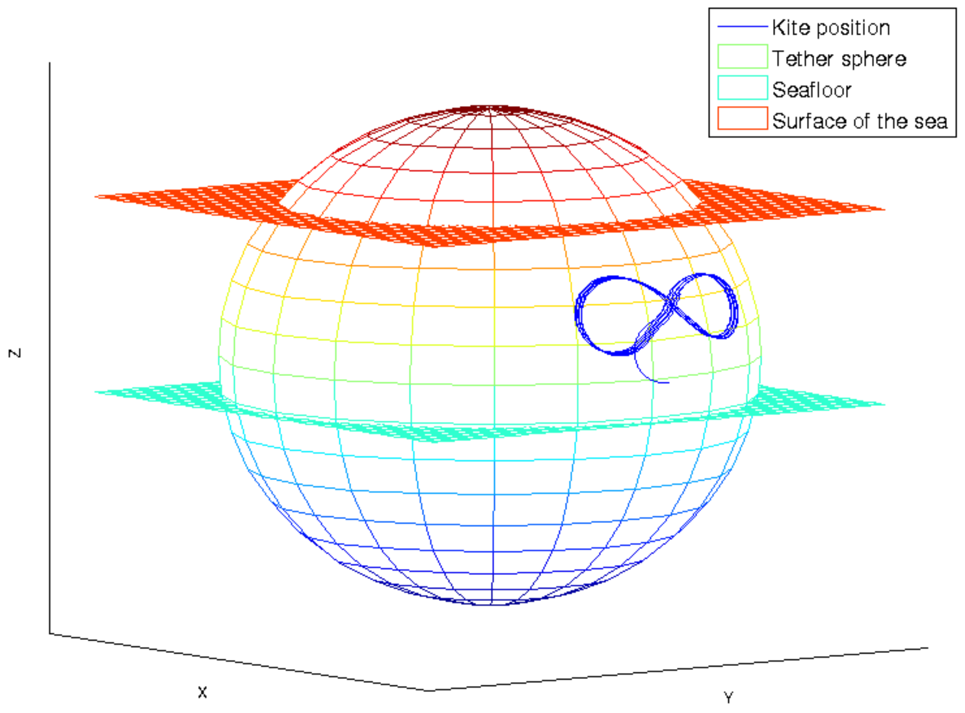

Figure 8.1 – History plot of the position, viewed in 3D. Device steered in an infinity loop by the control system.

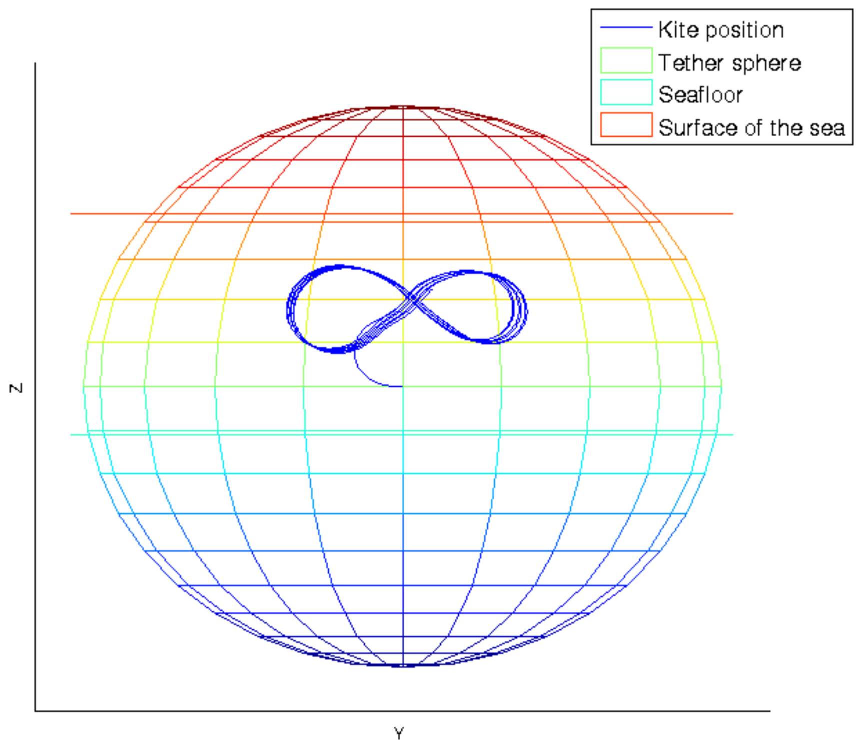

Figure 8.2 - History plot of the position, viewed in 2D. Device steered in an infinity loop by the control system.

As shown in detail in Figure 8.3, the control system manages to steer the device as desired, following the trajectory of an infinity loop or a horizontal eight. The device is initiated a distance above the seabed and the control system takes it up higher to the desired depth and starts to follow the specified trajectory.

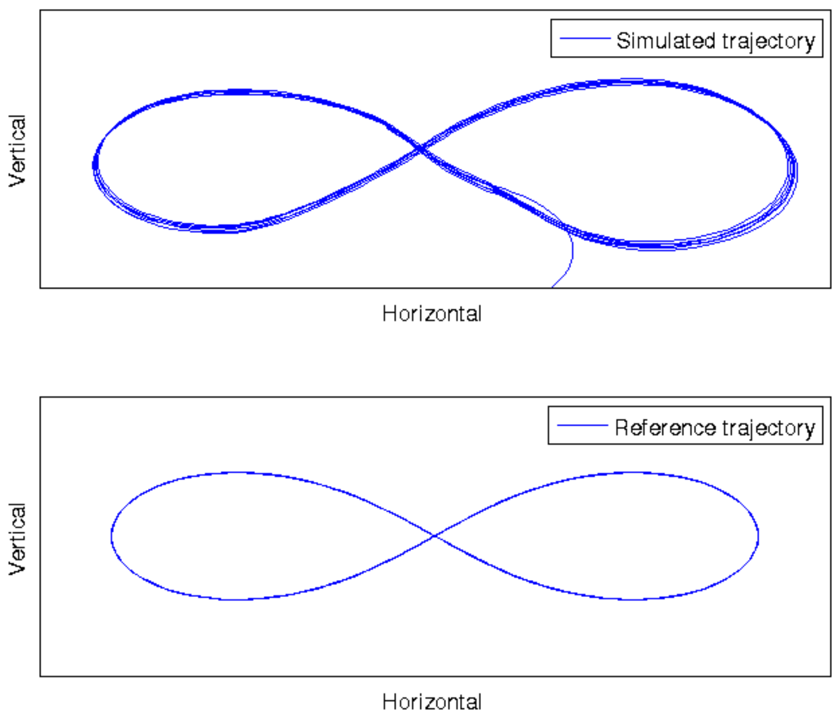

Figure 8.3 – Comparison of the simulated trajectory with the desired reference trajectory from the control system.

The speed and direction of the flow is arbitrary set and the device is initiated automatically depending on the flow and the user-defined starting depth. In Figure 8.4 the flow direction is set to 30 degrees, meaning that the flow is directed between north-north-east and north-east. The device starts directed to the right when viewing the device from the swivel, i.e. the origo of the coordinate system.

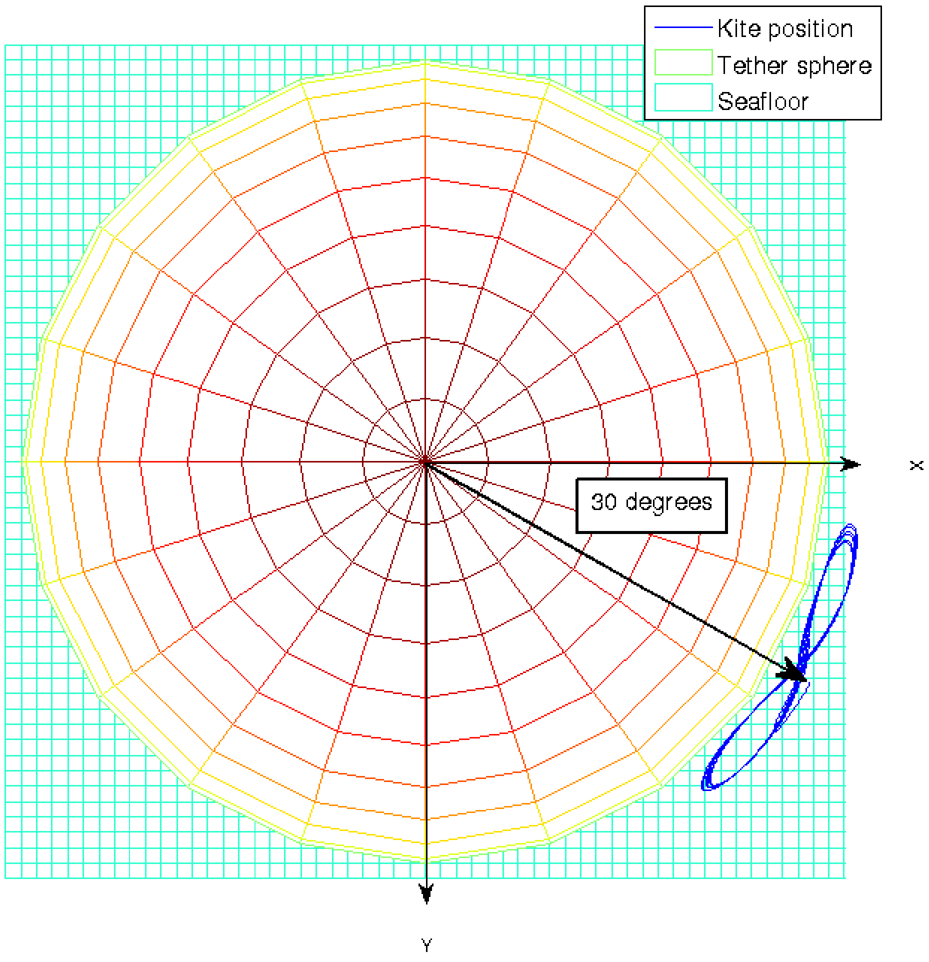

Figure 8.4 – History plot of the position, shown from above with 30 degrees flow direction angle, i.e. flow directed between NNE and NE.

When simulating the Deep Green device with a constant rudder angle it moves, as in Figure 8.5, in a smooth circle. This is an easier case to evaluate the hydrodynamic state compared to when it is steered in an infinity loop since it stabilizes after a while.

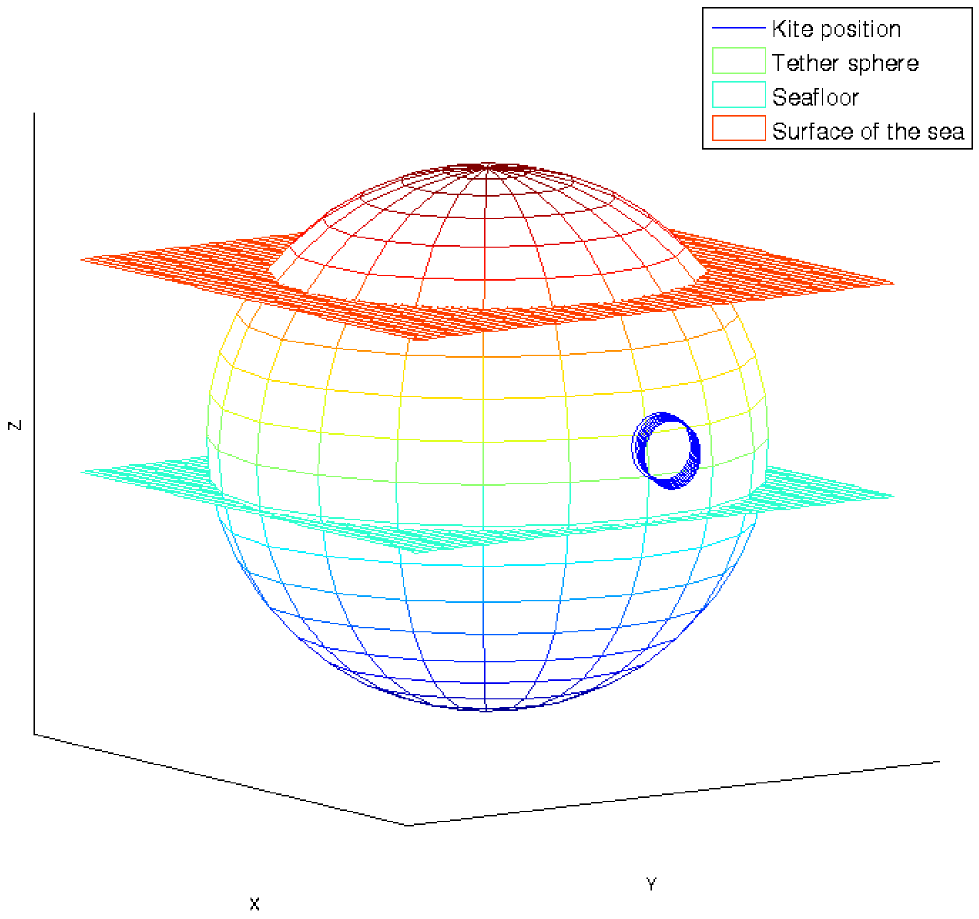

Figure 8.5 – History plot of the position with constant rudder angle.

To evaluate the hydrodynamic state of the device the hydrodynamic properties described in section 6.4.1 are plotted, i.e. the angle of attack α and the side slip angle β, together with the speed of the device. As seen in Figure 8.6, the angle of attack α stabilizes awhile after the initiation and is slightly varying with the circling. The chosen angle of attack at the initiation is clearly off the stabilized mean angle for this flow, but the simulation handles the difference fast without numerical issues.

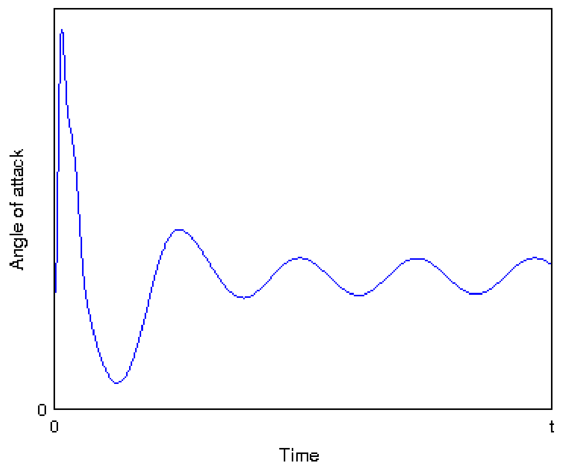

Figure 8.6 – The angle of attack as a function of time, constant rudder angle.

The initial speed of the device in Figure 8.7 is set depending on the flow speed, accelerates due to the initially high angle of attack and is later stabilized at a mean value, slightly varying with the turning.

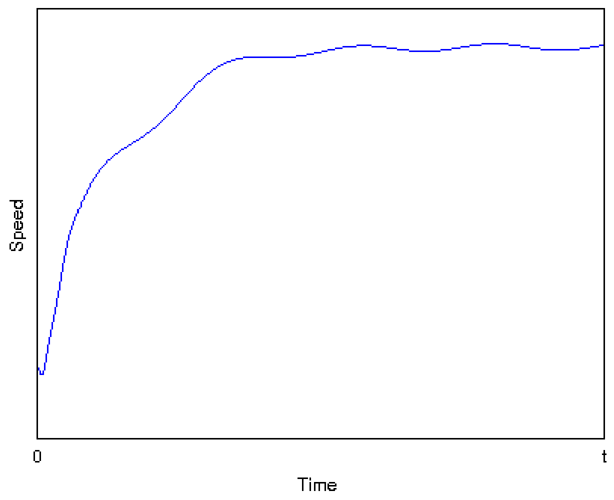

Figure 8.7 – Speed as a function of time, constant rudder angle.

The side slip angle β is also expected to stabilize at a mean value as in Figure 8.8 due to the constant rudder angle making the device to constantly turn; the flow always hitting the device a bit from the side.

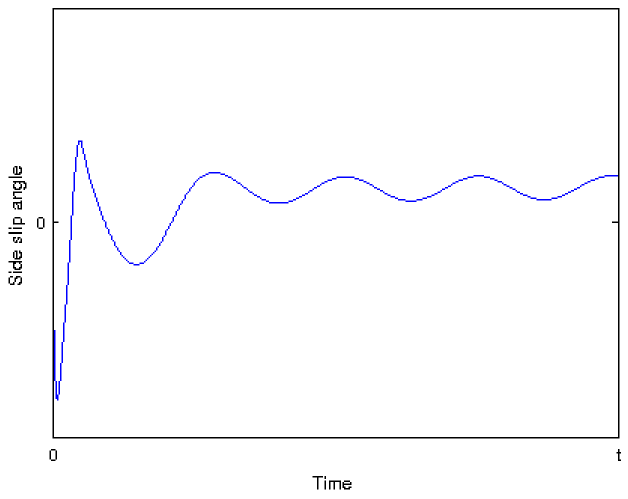

Figure 8.8 – Side slip angle as a function of time, constant rudder angle.

9 Discussion

The objectives for the master's thesis are successfully fulfilled; a working motion simulator for simulating the Deep Green device flying in water is produced within the timeframe of the thesis. The simulation initiates automatically with respect to the prevailing flow conditions and the starting depth specified by the user. A control system is integrated that steers the movement of the device as desired, following the trajectory of an infinity loop or a horizontal eight.

The application area for the simulator is to evaluate the performance of a device during normal flight conditions. Slack water, the intermediate state between tides, is not thoroughly investigated in this thesis and considered not to be a normal flight condition. The device is set to have a starting speed proportional to the flow speed at the initiation; hence the initiation procedure is not for evaluation purposes. Additional to evaluating different device designs the simulator is suitable for control system development and educational purposes, especially if it is integrated with a visualization program. Since the simulator is implemented in Simulink it is easily integrated with other applications compatible with MATLAB and Simulink.

9.1 Approximations

There are several approximations made during the simulator development to be explained. When deriving the governing motion equations the following approximations were made, all very trivial and more of academic interest:

- Flat earth assumption
- The device is a rigid body
- No change in mass or inertia
- The angular momentum of the turbine is neglected

The simulator is designed to simulate devices in shallow waters, spanning maybe hundreds of meters, and is therefore not affected by the curvature of the earth. The Deep Green device developed by Minesto is a rigid body that does not change in mass or inertia significantly. The turbine is to begin with not implemented in the simulator. When implemented, the angular momentum of the turbine is measured not to affect the simulation and can therefore be neglected.

The following assumptions and simplifications are made for making the implementation easier:

- The body is assumed to rotate around its center of mass
- The pitch and roll damping moments are artificial
- The tether is modelled as a damped spring; rigid and without drag

It is most likely that the device does not rotate around its center of mass in reality, but somewhere between the hydrodynamic center of effort and the center of mass, due to the fact that it is moving in water. But, the point of rotation will not differ a lot from the center of mass.

The damping pitch and roll moments are chosen to be artificial to decrease the computational time in the hydrodynamic analysis and the complexity of the polynomial. They are not as great in magnitude as the damping yaw moment and do not affect the simulation significantly when changed.

The implemented behaviour of the tether is the greatest approximation of them all, made with the ambition to get the simulator working and to later be developed further by the company. The main approximation is the exclusion of the retarding drag by the tether.

9.2 Future work

There are a few things to do before the simulation results are truthful enough for performance evaluation purposes. First of all, it needs to be thoroughly verified, preferably with measured data from physical tests. Before this is possible, a model needs to be added to catch the drag effects from the turbine and the generator. This also enables the ability to calculate the simulated power production. Probably the retarding drag from the tether also needs to be added before it is possible to compare the results with physical measurements in a satisfactory manner.

The flow model is fairly simple to develop further, for example by expanding the flow to 3D with site specific measurements. Then the flow can differ in all three dimensions, making the model closer to reality.

The simulator is a great tool for educational purposes, especially if it is integrated with a visualization program. The results can then be pedagogically visualized, directly during the simulation, suitable for directors, stakeholders and developers.

10 Bibliography

[1] Minesto AB, “Minesto AB,” 2013. [Online]. Available: http://www.minesto.com/deepgreentechnology/index.html. [Accessed 3 June 2013].

[2] J. D. Anderson, Fundamentals of Aerodynamics, New York: McGraw Hill Higher Education, 2011.

[3] S. L. Dixon and C. A. Hall, Fluid Mechanics and Thermodynamics of Turbomachinery, Oxford: Elsevier Inc., 2010.

[4] R. F. Stengel, Flight Dynamics, Princeton, New Jersey: Princeton University Press, 2004.

[5] W. R. Hamilton, “On Quaternions; or on a new System of Imaginaries in Algebra,” *The London, Edinburgh and Dublin Philosophial Magazince and Journal of Science,* Vols. XXV-XXXVI, 1844-1850.

[6] J. B. Kuipers, “Quaternions and Rotation Sequences,” *Geometry, Integrability and Quantization,* pp. 127-143, 1-10 September 1999.

[7] T. I. Fossen, *Lecture notes TTK4190 Guidance and control of Vehicles,* Trondheim: NTNU, 2013.

[8] R. F. Stengel, “Princeton University,” Princeton University, 1 September 2006. [Online]. Available: http://www.princeton.edu/~stengel/FDcodeB.html. [Accessed 1 February 2013].

[9] T. I. Fossen, “Norwegian University of Science and Technology,” 26 April 2013. [Online]. Available: http://www.itk.ntnu.no/ansatte/Fossen_Thor/. [Accessed 12 February 2013].

[10] T. I. Fossen and T. Perez, “Marine System Simulator,” Norwegian University of Science and Technology, 2010. [Online]. Available: www.marinecontrol.org/. [Accessed 12 February 2013].

[11] C. Nyberg, Mekanik: Grundkurs, Stockholm: Liber, 2003.

[12] D. C. Lay, Linear Algebra and its Applications, Boston: Pearson Education Inc., 2006.

[13] T. L. Heath, The Works of Archimedes, Cambridge: Cambridge University Press, 1897.

[14] E. Kreyszig, H. Kreyszig and E. J. Norminton, Advanced Engineering Mathematics, Jefferson City: John Wiley & Sons, 2010.