

WEC-Sim

USER MANUAL

Version Pre-Release

June 19, 2014

License

Copyright 2014 the National Renewable Laboratory and Sandia Corporation

Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

Acknowledgments

WEC-Sim is a multi-lab project sponsored by the U.S. Department of Energy's Wind and Water Power Technologies Office. WEC-Sim code development is a collaboration between the National Renewable Energy Laboratory (NREL) and Sandia National Laboratories (SNL)¹.

Principal Developers

Kelley Ruehl (SNL)
Carlos Michelen (SNL)

Michael Lawson (NREL)
Yi-Hsiang Yu (NREL)

Contributors

Nathan Tom (NREL)
Sam Kanner (University of
California at Berkeley)

Adam Nelessen (Georgia Tech)
Chris McComb (Carnegie
Mellon University)



¹Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Contents

1	Introduction	5
2	Theory	6
2.1	Introduction	6
2.2	Coordinate System in WEC-Sim	7
2.3	Boundary Element Method	7
2.4	Time-Domain Numerical Method	8
2.4.1	Sinusoidal Steady-State Response Scenario	8
2.4.2	Convolution Integral Formulation	9
2.4.3	Wave Spectrum	10
2.4.4	Ramp Function	13
2.5	Viscous Drag	14
2.6	Power Takeoff Forces	14
2.7	Mooring Forces	15
3	Getting Started	16
3.1	Downloading and Installing WEC-Sim	16
3.2	Running WEC-Sim	18
4	Library Structure	22
4.1	Library Structure Overview	22
4.2	Body Elements Sublibrary	23
4.2.1	Rigid Body Block	23
4.3	Frames Sublibrary	23
4.3.1	Global Reference Frame Block	25
4.4	Constraints Sublibrary	25
4.4.1	Floating Block	25
4.4.2	Heave Block	28

4.4.3	Surge Block	28
4.4.4	Pitch Block	28
4.4.5	Fixed Block	28
4.5	PTOs Sublibrary	29
4.5.1	Translation PTO (Local Z) Block	29
4.5.2	Translation PTO (Local X) Block	29
4.5.3	Rotational PTO (Local RY) Block	29
4.6	A Note on SimMechanics TM and Coordinate Systems	31
5	Code Structure and Input Parameters	33
5.1	Units	33
5.2	WEC-Sim Input File	33
5.3	WEC-Sim Code	36
5.3.1	simulationClass	36
5.3.2	bodyClass	37
5.3.3	waveClass	39
5.3.4	constraintClass	41
5.3.5	ptoClass	42
6	Applications	43
6.1	RM3 Two-Body Point Absorber	43
6.1.1	Geometry Definition	43
6.1.2	Modeling RM3 in WEC-Sim	45
6.2	Oscillating Surge-Pitch Device	51
6.2.1	Geometry Definition	51
6.2.2	Modeling OSWEC in WEC-Sim	52
6.3	WEC-Sim Test Cases	58
6.3.1	RM3 Test Case	58
6.3.2	OSWEC Test Case	60
7	Troubleshooting	61

Chapter 1

Introduction

WEC-Sim (Wave Energy Converter SIMulator) is an open source wave energy converter (WEC) simulation tool. The code is developed in MATLAB/Simulink using the multi-body dynamics solver SimMechanics. WEC-Sim has the ability to model devices that are comprised of rigid bodies, power-take-off systems, and mooring systems. Simulations are performed in the time-domain by solving the governing WEC equations of motion in 6 degrees-of-freedom as described in Chapter 2.

The WEC-Sim project is funded by the U.S. Department of Energy's Wind and Water Power Technologies Office and the code development effort is a collaboration between the National Renewable Energy Laboratory and Sandia National Laboratories.

Chapter 2

Theory

2.1 Introduction

Modeling a wave energy converter (WEC) involves the interaction between the incident waves, device motion, power takeoff (PTO) mechanism, and mooring (Figure 2.1). WEC-Sim uses a radiation and diffraction method [1, 2] to predict power performance and design optimization. The radiation and diffraction method generally obtains the hydrodynamic forces from a frequency-domain boundary element method solver using linear coefficients to solve the system dynamics in the time domain.

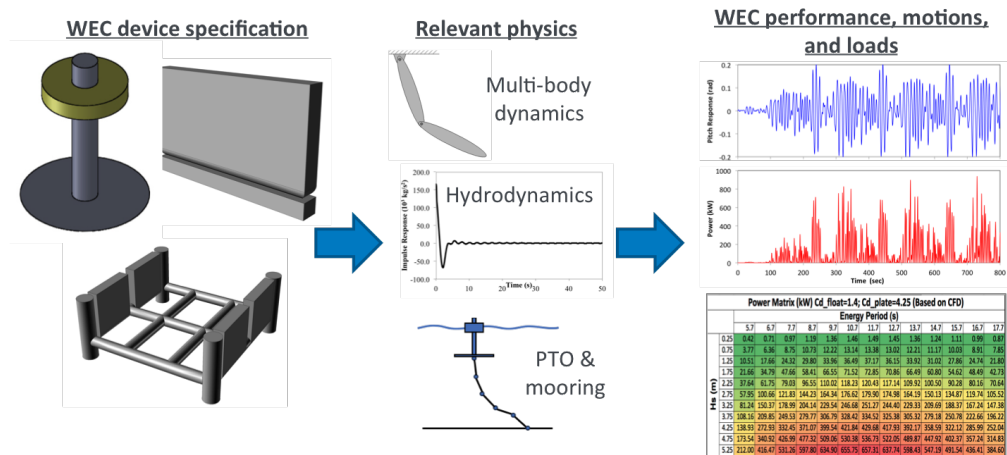


Figure 2.1: Methodology for WEC-Sim

2.2 Coordinate System in WEC-Sim

Figure 2.2 illustrates a three-dimensional floating point absorber subject to incoming waves in water. The figure also defines the coordinates and the six degrees of freedom in WEC-Sim.

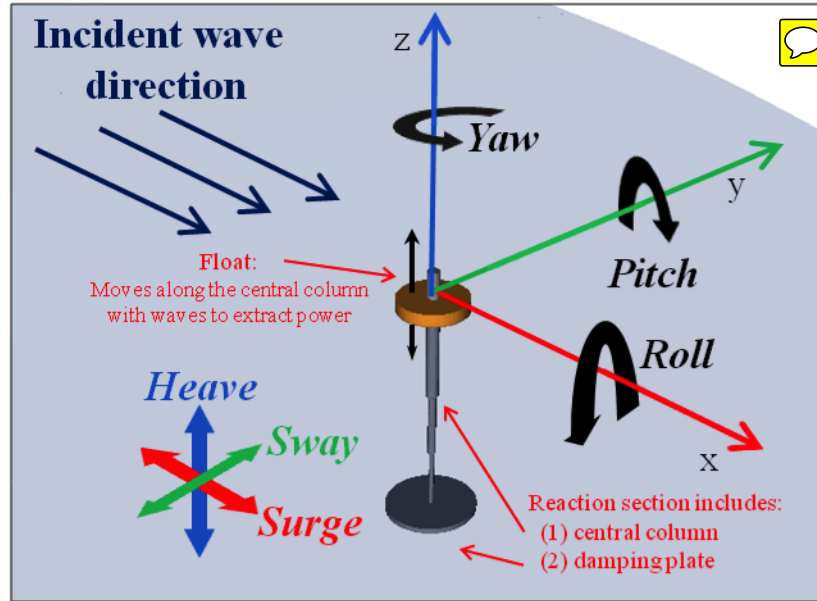


Figure 2.2: Sketch defining the coordinate system

2.3 Boundary Element Method

A common approach to determine the hydrodynamic forces is to presume that it is the sum of incident, radiated, and diffracted wave components. These forcing components are modeled using linear coefficients and can be obtained from a frequency-domain potential flow boundary element method (BEM) solver (e.g., WAMIT [3], AWQAPER, and Nemoh [4]). The BEM solutions are obtained by solving the Laplace equation for the velocity potential, which assumes the flow is inviscid, incompressible, and irrotational. More details on the theory for the frequency-domain BEM can be found in [3].

2.4 Time-Domain Numerical Method

The dynamic response of the system was calculated by solving the Cummins equation [5], often used to represent the equation of motion for marine systems. The equation of motion for a floating-body, about its center of gravity, can be given as:

$$m\ddot{X} = F_{ext} + F_{rad} + F_{PTO} + F_v + F_B + F_m, \quad (2.1)$$

where \ddot{X} is the (translational and rotational) acceleration vector of the device, m is the mass matrix, F_{ext} is the wave excitation force vector, F_{rad} is the force vector due to wave radiation, F_{PTO} is the PTO force vector, F_v is the viscous damping force vector, F_B is the net buoyancy restoring force vector, and F_m is the force vector due to the mooring connection.

Both F_{ext} and F_{rad} are calculated from the frequency-domain BEM solver. The radiation term includes an added-mass and wave damping term associated with the acceleration and velocity of the floating body respectively. The wave excitation term includes a Froude–Krylov force component generated by the undisturbed incident waves and a diffraction component due to the presence of the floating body. WEC-Sim can be used for regular and irregular wave simulations, but note that F_{ext} and F_{rad} are calculated differently for sinusoidal steady-state response scenarios and random sea simulations. The sinusoidal steady-state response is often used for simple WEC designs with regular incoming waves; however, for random sea simulations or any simulations where fluid memory effects of the system are essential the convolutional integral method is recommended to represent the fluid memory retardation force on the floating body.

2.4.1 Sinusoidal Steady-State Response Scenario

This approach assumes the system response is in sinusoidal steady-state form, and is only valid for regular wave simulations. The radiation term can be calculated using the added-mass and the wave radiation damping term for a given wave frequency, which is obtained from

$$F_{rad} = -A(\omega_r)\ddot{X} - B(\omega_r)\dot{X} \quad (2.2)$$

where $A(\omega_r)$ and $B(\omega_r)$ are the added mass and wave radiation damping matrices, ω_r is the wave frequency, and \dot{X} is the velocity vector of the floating body.

The free surface profile is based on linear Airy wave theory for a given wave height, wave frequency, and water depth. The regular wave excitation force is obtained from

$$F_{ext} = \Re \left[R_f \frac{H}{2} F_X(\omega_r) e^{i(\omega_r t)} \right], \quad (2.3)$$

where \Re denotes the real part of the formula, R_f is the ramp function, H is the wave height, and F_X is the excitation vector, including the magnitude and phase of the force.

2.4.2 Convolution Integral Formulation

To include the fluid memory effect on the system, the convolution integral calculation is used, which is based upon the Cummins equation. The radiation term can be calculated by

$$F_{rad} = -A_\infty \ddot{X} - \int_0^t K(t - \tau) \dot{X}(\tau) d\tau \quad (2.4)$$

where A_∞ is the added mass matrix at infinite frequency and K is the impulse response function.

For regular waves, Eq. 2.3 is used to calculate the wave excitation force. For irregular waves, the free surface elevation is constructed from a linear superposition of a number of regular wave components. It is often characterized using a wave spectrum (Figure 2.3) that describes the wave energy distribution over a range of wave frequencies, characterized by a significant wave height and peak wave period. The irregular excitation force can be calculated as the real part of an integral term across all wave frequencies,

$$F_{ext} = \Re \left[R_f \int_0^\infty \sqrt{2S(\omega_r)} F_X(\omega_r) e^{i(\omega_r t + \phi)} d\omega_r \right], \quad (2.5)$$

where S is the wave spectrum and ϕ is a random phase angle.

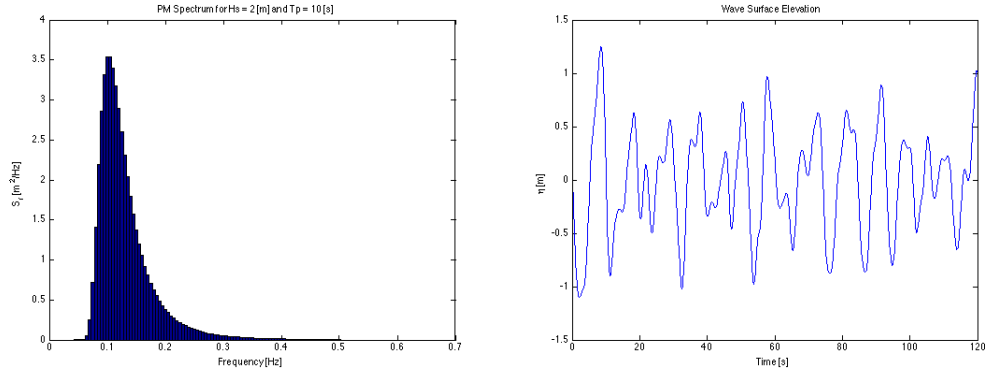


Figure 2.3: An example of wave spectrum and irregular wave elevation generated by WEC-Sim (Pierson–Moskowitz spectrum)

2.4.3 Wave Spectrum

The ability to generate regular waves provides an opportunity to observe the response of a model under specific conditions. However, sea states with constant wave heights and periods are rarely found outside testing conditions. Normal sea conditions are more accurately represented by random-wave time series that model the superposition of various wave forms with different amplitudes and periods. This superposition of waves is characterized by a sea spectrum. Through statistical analysis, spectra are characterized by specific parameters such as significant wave height, peak period, wind speed, fetch length, and others. The common types of spectra that are used by the offshore industry are discussed in the following sections. The general form of the sea spectrums available in WEC-Sim is given by:

$$S(f) = Af^{-5} \exp[-Bf^{-4}] \quad , \quad (2.6)$$

where f is the wave frequency (in Hertz) and \exp stands for the exponential function.

Pierson-Moskowitz

One of the simplest spectrums was proposed by *Pierson and Moskowitz 1964*. The assumption made was that after the wind blew steadily for a long time over a large area, the waves would come into equilibrium with the wind. This is the concept of a fully developed sea, where a "long time" is roughly ten-thousand wave periods,

and a "large area" is roughly five-thousand wave-lengths on a side. The spectrum is calculated from:

$$S(f) = \frac{\alpha_{PM} g^2}{(2\pi)^4} f^{-5} \exp \left[-\frac{5}{4} \left(\frac{f_p}{f} \right)^4 \right] , \quad (2.7)$$

$$A = \frac{\alpha_{PM} g^2}{(2\pi)^4} , \quad B = \frac{5}{4} f_p^4 , \quad (2.8)$$

where $\alpha_{PM} = 0.0081$, g is gravitational acceleration, and f_p is the peak frequency of the spectrum.

Bretschneider Spectrum

A two parameter spectrum based on significant wave height and peak wave frequency. For a given significant wave height, the peak frequency can be varied to cover a range of conditions including developing and decaying seas. In general, the parameters depend on wind speed (most important), wind direction, fetch and locations of storm fronts.

$$S(f) = \frac{\alpha_{PM} g^2}{(2\pi)^4} f^{-5} \exp \left[-\frac{5}{4} \left(\frac{f_p}{f} \right)^4 \right] , \quad (2.9)$$

$$A = \frac{H_{m0}^2}{4} B = \frac{H_{m0}^2}{4} (1.057 f_p)^4 \approx \frac{5}{16} H_{m0}^2 f_p^4 , \quad (2.10)$$

$$B = (1.057 f_p)^4 \approx \frac{5}{4} f_p^4 , \quad (2.11)$$

where H_{m0} is the significant wave height which is generally defined as the mean wave height of the one third highest waves.

JONSWAP Spectrum

At the 17th International Towing Tank Conference (ITTC) the JONSWAP (Joint North Sea Wave Project) spectrum was defined as:

$$S(f) = \frac{155}{(2\pi)^4} \frac{H_{m0}^2}{(0.834 T_p)^4} f^{-5} \exp \left[-\frac{5}{4} \left(\frac{f_p}{f} \right)^4 \right] \gamma^\Gamma$$

$$\approx \frac{310}{(2\pi)^4} H_{m0}^2 f_p^4 f^{-5} \exp \left[-\frac{5}{4} \left(\frac{f_p}{f} \right)^4 \right] \gamma^\Gamma , \quad (2.12)$$

$$\Gamma = \exp \left[-\left(\frac{\frac{f}{f_p} - 1}{\sqrt{2}\sigma} \right)^2 \right] , \quad \sigma = \begin{cases} 0.07 & f \leq f_p \\ 0.09 & f > f_p \end{cases} , \quad (2.13)$$

$$A = \frac{310}{(2\pi)^4} H_{m0}^2 f_p^4 , \quad B = \frac{5}{4} f_p^4 . \quad (2.14)$$

The above representation was decided at the 17th International Towing Tank Conference where the original spectrum definition is found in *Hasselmann et. al 1977* and given by:

$$S(f) = \frac{\alpha_j g^2}{(2\pi)^4} f^{-5} \exp \left[-\frac{5}{4} \left(\frac{f_p}{f} \right)^4 \right] \gamma^\Gamma$$

$$\Gamma = \exp \left[-\left(\frac{\frac{f}{f_p} - 1}{\sqrt{2}\sigma} \right)^2 \right] , \sigma = \begin{cases} 0.07 & f \leq f_p \\ 0.09 & f > f_p \end{cases} , \quad (2.15)$$

$$A = \frac{\alpha_j g^2}{(2\pi)^4} , \quad B = \frac{5}{4} f_p^4 , \quad (2.16)$$

where α_j is a nondimensional variable that is a function of the wind speed and fetch length. Empirical fits were applied in an attempt to find a mean value that would capture the spectral shape of most measured sea states; however, no conclusive results were made in the original work resulting in the ITTC definition. Therefore, it is reasonable to fit α_j such that the desired significant wave height is as achieved.

Spectral Moment and Significant Wave Height

Related to the spectrum is a series of characteristic numbers called the spectral moments. These numbers, denoted m_k , $k = 0, 1, 2, \dots$ are defined as:

$$m_k = \int_0^\infty f^k S(f) df . \quad (2.17)$$

The spectral moment, m_0 is ~~just~~ the variance of the free surface which allows one to define:

$$H_{m0} = 4\sqrt{m_0} . \quad (2.18)$$

In order to fit α_j to match the desired significant wave height the following calculation must be performed:

$$\alpha_j = \frac{H_{m0}^2}{16 \int_0^\infty S^*(f) df} , \quad (2.19)$$

$$S^*(f) = \frac{g^2}{(2\pi)^4} f^{-5} \exp \left[-\frac{5}{4} \left(\frac{f_p}{f} \right)^4 \right] \gamma^\Gamma . \quad (2.20)$$

The two spectral methods: 1) α_j fit and 2) the ITTC description are compared in Fig. 2.4 which have almost perfect agreement.

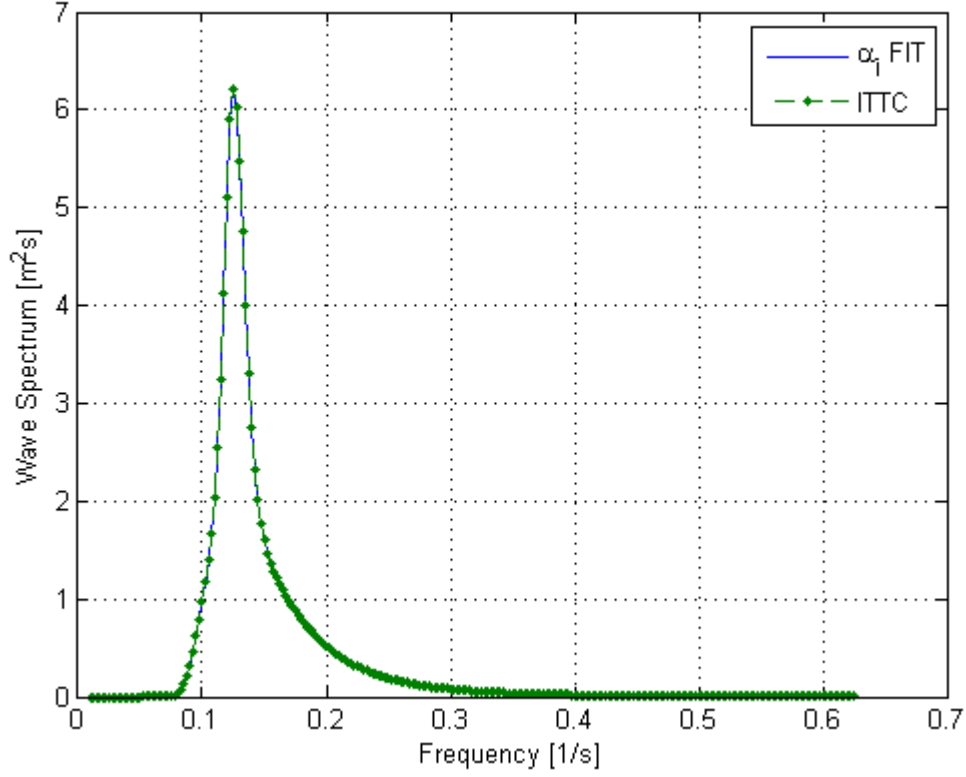


Figure 2.4: Comparison of α_j fit to the ITTC description of the JONSWAP spectrum with $H_{m0} = 2$ m and $T_p = 8$ s.

2.4.4 Ramp Function

A ramp function (R_f), necessary to avoid strong transient flows at the earlier time steps of the simulation, is used to calculate the wave excitation force. The ramp function is given by

$$R_f = \begin{cases} \frac{1}{2}(1 + \cos(\pi + \frac{\pi t}{t_r})), & \frac{t}{t_r} < 1 \\ 1, & \frac{t}{t_r} \geq 1 \end{cases} \quad (2.21)$$

where t is the simulation time and t_r is the ramp time.

2.5 Viscous Drag

Generally, the effect of viscosity on the WEC dynamics needs to be considered while neglecting this effect may lead to an overestimation of the power generation of the system, particularly when a linear model is applied. A common way of modeling the viscous damping is to add a (Morison-equation-type) quadratic damping term to the equation of motion,

$$F_V = \frac{1}{2}C_d\rho A_D\dot{X}|\dot{X}|, \quad (2.22)$$

where C_d is the viscous drag coefficient, ρ is the fluid density, A_D is the characteristic area, and $|\cdot|$ denotes absolute value. The viscous damping coefficient for the device must be carefully selected [1, 2]. The drag coefficient is dependent on device geometry, scale, and relative velocity between the body and the flow around it. The drag coefficient becomes much larger when the Reynolds and the Keulegan–Carpenter number are smaller. Note that empirical data on the viscous damping coefficient can be found in various literature and standards. However, the available data may be limited to existing simple geometries. For practical point absorber geometry, the hydrodynamic forces may have to be evaluated by conducting wave tank tests or prescribed motion computational fluid dynamic (CFD) simulations.

2.6 Power Takeoff Forces

The power takeoff (PTO) mechanism was represented as a linear spring-damper system, where the reaction force is given by:

$$F_{PTO} = -K_{PTO}X_{rel} - C_{PTO}\dot{X}_{rel}, \quad (2.23)$$

where K_{PTO} is the stiffness of the PTO, C_{PTO} is the damping of the PTO, and X_{rel} and \dot{X}_{rel} are the relative motion and velocity between two bodies. The power consumed by the PTO is given by:

$$P_{PTO} = -F_{PTO}\dot{X}_{rel} = \left(K_{PTO}X_{rel}\dot{X}_{rel} + C_{PTO}\dot{X}_{rel}^2\right); \quad (2.24)$$

however, the relative motion and velocity between two bodies will be out of phase by $\pi/2$ resulting in the time-averaged product to be 0. This allows the absorbed power to be written as:

$$P_{PTO} = C_{PTO}\dot{X}_{rel}^2. \quad (2.25)$$

2.7 Mooring Forces

The mooring load was represented using a linear quasi-static mooring stiffness, which can be calculated by

$$F_m = -K_m X, \quad (2.26)$$

where K_m is the stiffness matrix for the mooring system, and X is the response of the body.

Chapter 3

Getting Started

WEC-Sim is implemented in MATLAB [6] and running the code requires that you install MATLAB, the MATLAB toolboxes presented in Table 3.1, and the WEC-Sim source code. WEC-Sim was developed in MATLAB R2014a and we recommend using this MATLAB version. In this chapter we describe how to download and install WEC-Sim (Section 3.1) and how to run a WEC-Sim simulation (Section 3.2).

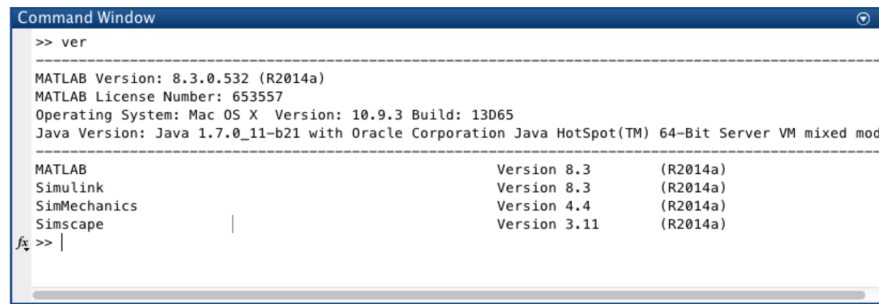
3.1 Downloading and Installing WEC-Sim

Step 1: Install MATLAB

Download and install MATLAB and the MATLAB toolboxes presented in Table 3.1. Ensure you have the required toolboxes installed by running the `ver` command from the MATLAB Command Window (Figure 3.1). Consult the MathWorks website for more information on performing this step.

Matlab Package	Required Release
MATLAB Base	R2014a Version 8.3
Simulink	R2014a Version 8.3
SimMechanics	R2014a Version 4.4
Simscape	R2014a Version 3.11

Table 3.1: Running the `ver` command from the MATLAB Command Window.



```

Command Window
>> ver

-----
MATLAB Version: 8.3.0.532 (R2014a)
MATLAB License Number: 653557
Operating System: Mac OS X Version: 10.9.3 Build: 13D65
Java Version: Java 1.7.0_11-b21 with Oracle Corporation Java HotSpot(TM) 64-Bit Server VM mixed mode
-----
MATLAB                               Version 8.3      (R2014a)
Simulink                             Version 8.3      (R2014a)
SimMechanics                         Version 4.4      (R2014a)
Simscape                             Version 3.11     (R2014a)
fx >> |

```

Figure 3.1: Required MATLAB toolboxes

Step 2: Download WEC-Sim

Download WEC-Sim from the WEC-Sim GitHub website.




Step 3: Add the WEC-Sim Source Code to the MATLAB Search Path

Open the `wecSimStartup.m` file that is located in the `functions` folder within the WEC-Sim source code directory (referred to as `wecSimSource` in this document). Copy the code in the `wecSimStartup.m` file and paste it into the `startup.m` file within the MATLAB Startup Folder. Set the `wecSimPath` variable to the location of the `wecSimSource` folder on your computer. Close MATLAB, restart the code, and then run the `path` command from the MATLAB Command Window to verify that the directories listed in `wecsim-path-setup.m` are in the MATLAB Search Path.

Step 4: Add the WEC-Sim Blocks to the Simulink Library Browser

Open the Simulink Library Browser by typing `simulink` from the MATLAB Command Window. Once the Simulink Library Browser opens, select `View>Refresh Tree View`. At this point, you should be able to expand the WEC-Sim menu in the `Libraries` pane to view the WEC-Sim `Body`, `Constraints`, `PTOs`, and `Frame` blocks. The function of these blocks will be explained in Chapter 4. From now on every time you open Simulink in the WEC-Sim Library Browser the WEC-Sim `Body Elements`, `Constraints`, `PTOs`, and `Frames` blocks will be available. For more help using and modifying library blocks refer to the Simulink Documentation.

3.2 Running WEC-Sim

This section will give an overview of the WEC-Sim work-flow (Figure ) , how to run WEC-Sim, and the file structure of the WEC-Sim case directory. The remainder of this section describes the steps in setting up and running a WEC-Sim simulations. Details descriptions and option of input parameters for the input file are described in Chapter 5. Specific examples of using WEC-Sim to simulate WEC devices are presented in Chapter 6.

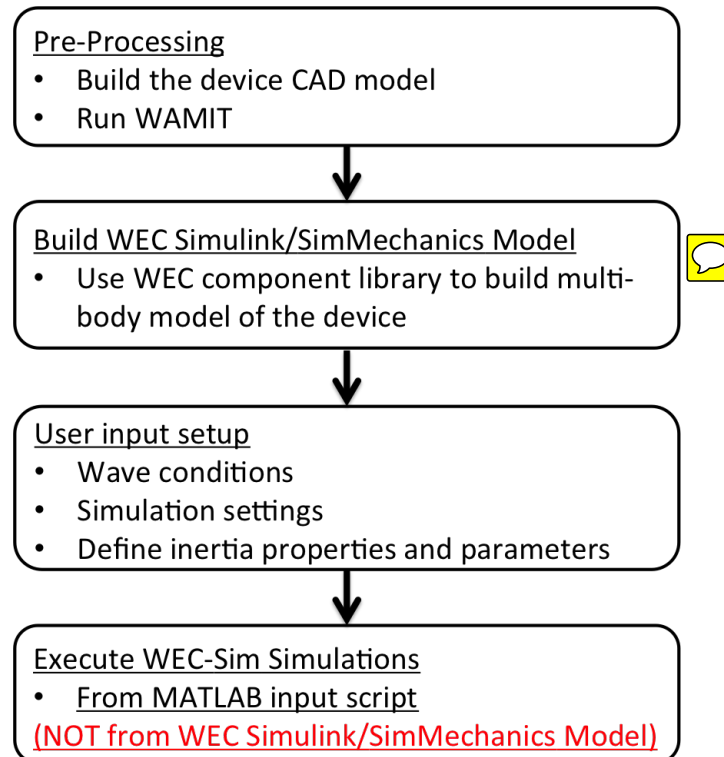


Figure 3.2: Workflow diagram for running WEC-Sim simulations

Table 3.2 shows the default file structure for WEC-Sim. All the necessary files for running WEC-Sim are located in the case directory or inside its subfolder.

	File name	Location
Input Script	wecSimInputFile.m	Case directory
WEC Model	WEC Model Name.slx	Case directory
WAMIT	WAMIT File Name.out	wamit
Geometry	STL File Name.stl	geometry

Table 3.2: Default files names and locations

Step 1: Pre-Processing

The user first needs to run WAMIT to generate the hydrodynamic coefficients for each body of the WEC device. WEC-Sim will read the WAMIT generated hydrodynamic coefficients from the WAMIT output file (`<wamit file name>.out`). To assure WEC-Sim uses the correct hydrodynamic coefficients to model the WEC system, the center of gravity for each body has to be at the origin of the body coordinate system (XBODY) in WAMIT. Note that the current version of WEC-Sim does not account for the multi-directional wave heading and WEC-Sim will use whatever wave heading was modeled in WAMIT. More details on WAMIT setup are described in the WAMIT User Manual [3].

Next the user must create representations of the WEC bodies in the STL file format. The STL files are used to visualize the WEC bodies in the WEC-Sim/MATLAB graphical user interface.

Step 2: Build Device Simulink/SimMechanics Model

The ~~user next needs~~ to build the device model using the Simulink/SimMechanics toolboxes and the WEC-Sim Library (see Chapter 4). Figure 3.3 shows an example of a two-body point absorber modeled in Simulink/SimMechanics.

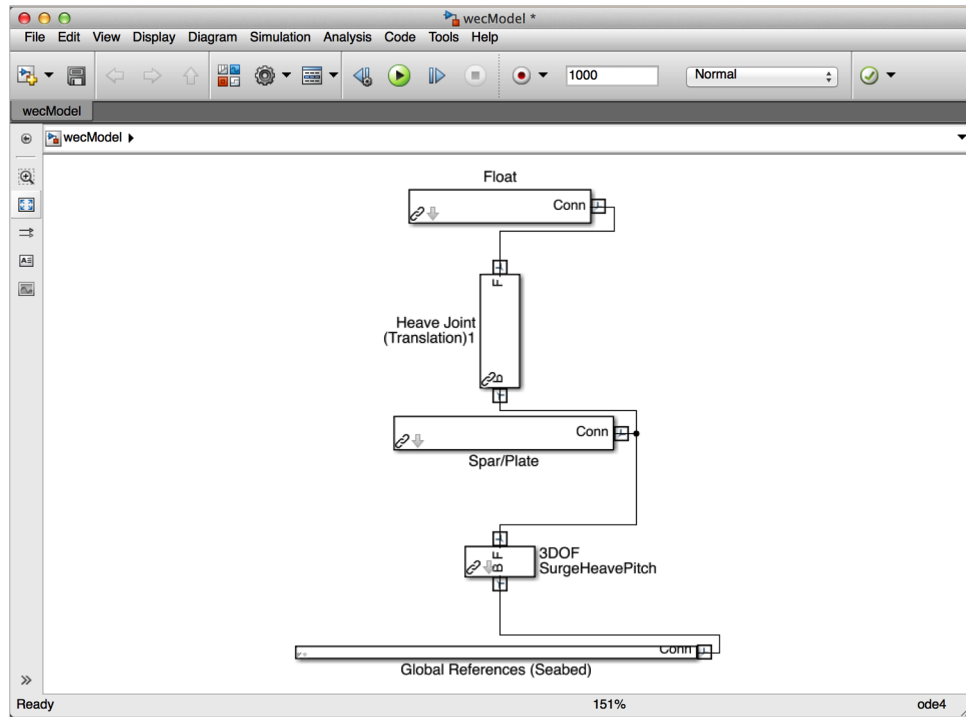


Figure 3.3: An example of device Simulink/SimMechanics model for a two-body point absorber

Step 3: Create WEC-Sim Input File

A WEC-Sim input file must be written in MATLAB and ~~needs to be created~~ in the case directory. It MUST be named `wecSimInputFile.m` and an example of the input file for a two-body point absorber is shown in Figure 3.4. In the input file, the simulation settings, sea state, body mass properties, PTO, and constraints are specified. In addition, users MUST specify the Simulink/SimMechanics model file name in the `wecSimInputFile.m`, as shown below

```
simu.simMechanicsFile='<WEC Model Name>.slx';
```

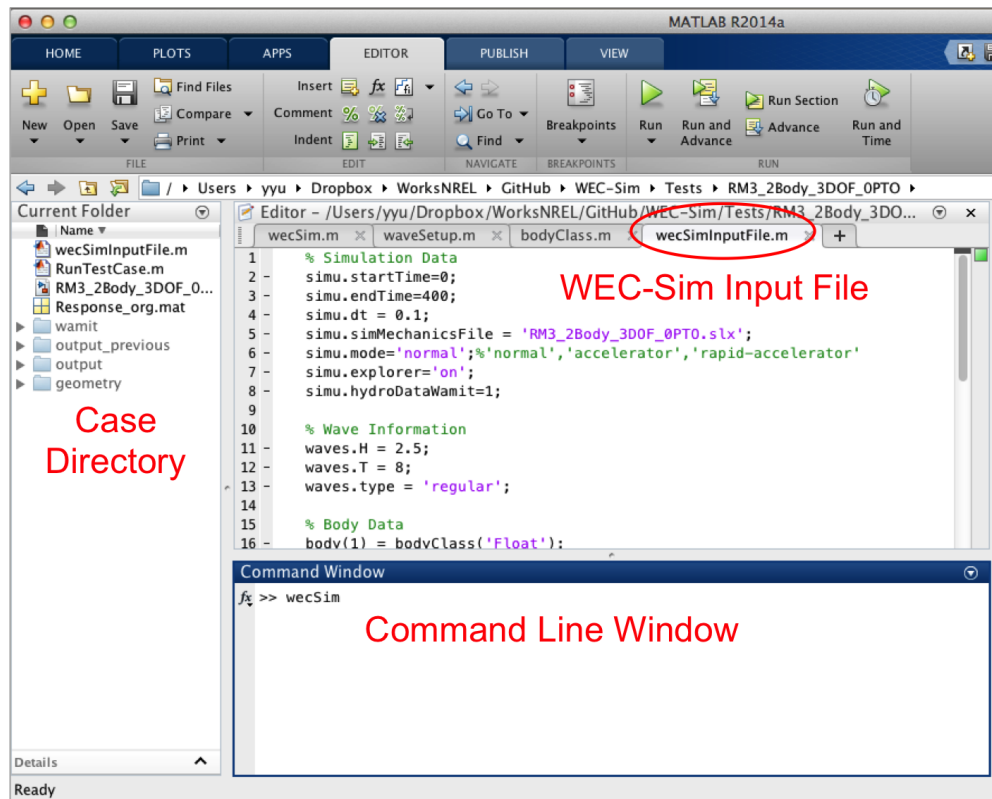


Figure 3.4: An example of running WEC-Sim

Step 4: Execute WEC-Sim

Finally, the user will need to execute the simulation by running `wecSim` [command](#) from the MATLAB Command Window (Figure 3.4). The `wecSim` command must be executed in the WEC-Sim case directory where the `wecSimInputFile.m` is located.

[Note that](#), WEC-Sim should always be executed from the MATLAB Command Window and not from the Simulink/SimMechanics model. This insures that the correct variables are in the MATLAB workspace during simulation.


Chapter 4

Library Structure

4.1 Library Structure Overview

The WEC-Sim library is divided into 4 sublibraries. The user should be able to model their WEC device using the available WEC-Sim blocks, and possibly some SimMechanicsTM blocks. Table 4.1 lists the WEC-Sim blocks and their organization into sublibraries.

Table 4.1: WEC-Sim library structure

WEC-Sim Library	
Sublibrary	Blocks
Body Elements	Rigid Body
Frames	Global Reference Frame
Constraints	Heave Pitch  Surge Fixed Floating
PTOs	Rotational PTO (Local RY) Translational PTO (Local X) Translational PTO (Local Z)

The following sections will describe the four sublibraries and their general purpose. The **Body Elements** sublibrary contains the **Rigid Body** block used to simulate the different bodies. The **Frames** sublibrary contains the **Global Reference Frame** block necessary for every simulation. The **Constraints** sublibrary contains blocks that are used to constrain the degrees of freedom of the bodies, without including any additional forcing or resistance. The **PTOs** sublibrary contains blocks used to both simulate a power take-off system and restrict the body motion. Both constraints and PTOs can be used to restrict the relative motion between multi-body systems. ~~In the rest of this chapter each sublibrary and block will be described.~~

4.2 Body Elements Sublibrary

The **Body Elements** sublibrary, shown in Figure 4.1, contains 1 block; the **Rigid Body** block, used to represent rigid bodies. At least one instance of this block is required in each model.

~~4.2.1~~ Rigid Body Block

The **Rigid Body** block is used to represent a rigid body in the simulation. The user has to name the blocks 'body(i)' where $i=1,2,\dots$. The mass properties, hydrodynamic data, geometry file, mooring, and other properties are then specified in the input file. Within the body block the following forces are calculated:

- Wave excitation
- Wave radiation
- Hydrostatic restoring force
- Viscous damping
- Mooring

4.3 Frames Sublibrary

The **Frames** sublibrary, shown in Figure 4.2, contains 1 block that is necessary in every model. The **Global Reference Frame** block defines global references and can

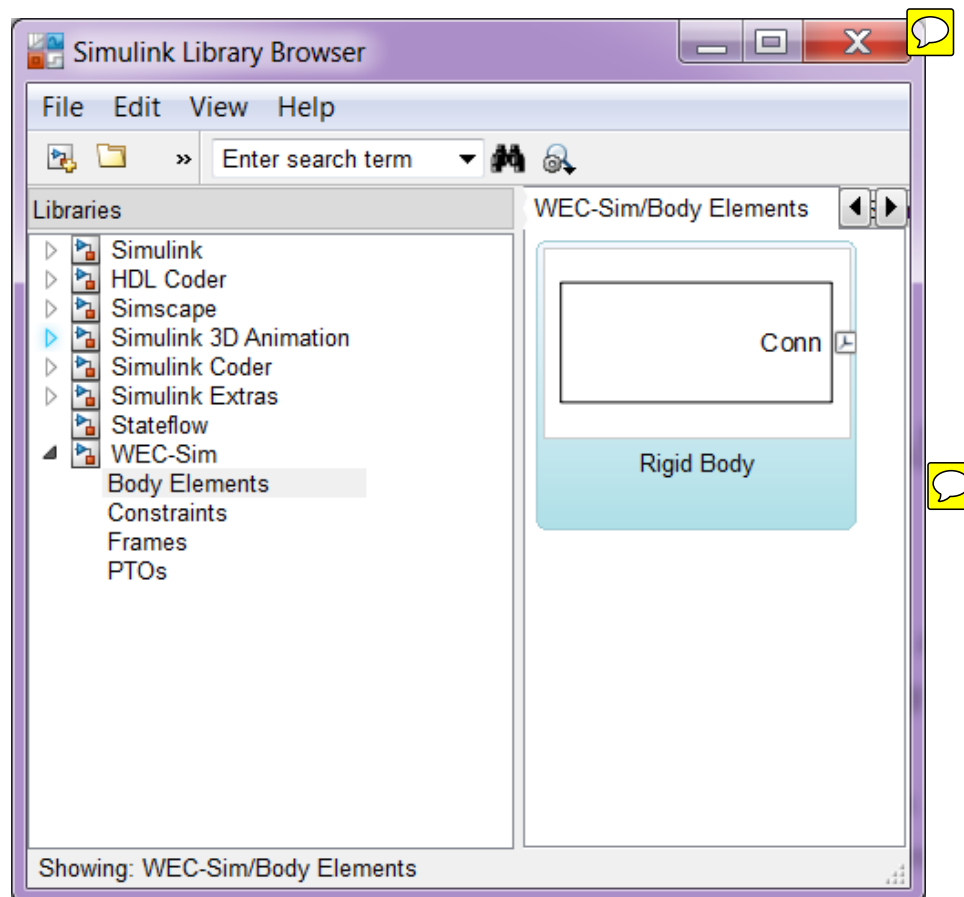


Figure 4.1: Body Elements sublibrary

be thought of as the seabed.

4.3.1 Global Reference Frame Block

The **Global Reference Frame** block defines the solver configuration, seabed and free surface description, simulation time, and other global settings. Every model requires one instance of the **Global Reference Frame** block. The simulation class variable `simu` and the wave class variable `waves` are defined in the input file.

4.4 Constraints Sublibrary

The blocks within the **Constraints** sublibrary, shown in Figure 4.3, are used to define the degrees of freedom of a specific body. **Constraints** blocks only define the degrees of freedom, but do not otherwise apply any forcing or resistance to the body motion. Each **Constraints** block has two connections; a base (B) and a follower (F). The **Constraints** blocks restrict the motion of (the block connected to) the follower relative to (the block connected to) the base. The base of these blocks is typically the **Global Reference Frame** (which can be thought of as the seabed) and the follower is a **Rigid Body**.

There are 5 **Constraints** blocks; three that restrict motion to one degree of freedom (**Heave**, **Surge**, **Pitch**), a free-floating (**Floating**) block, and a rigid connection (**Fixed**) block. The rest of this section will describe each **Constraints** block in more detail.

4.4.1 Floating Block

The **Floating** block is used to simulate a free-floating body. It constrains the motion of the follower to be along the XZ plane of the base. That is, it allows translation in the X- and Z-axis, and rotation about the Y-axis. It is usually used with the base connected to the **Global Reference Frame** (seabed) , in which case the motion of the follower is along the global XZ plane.

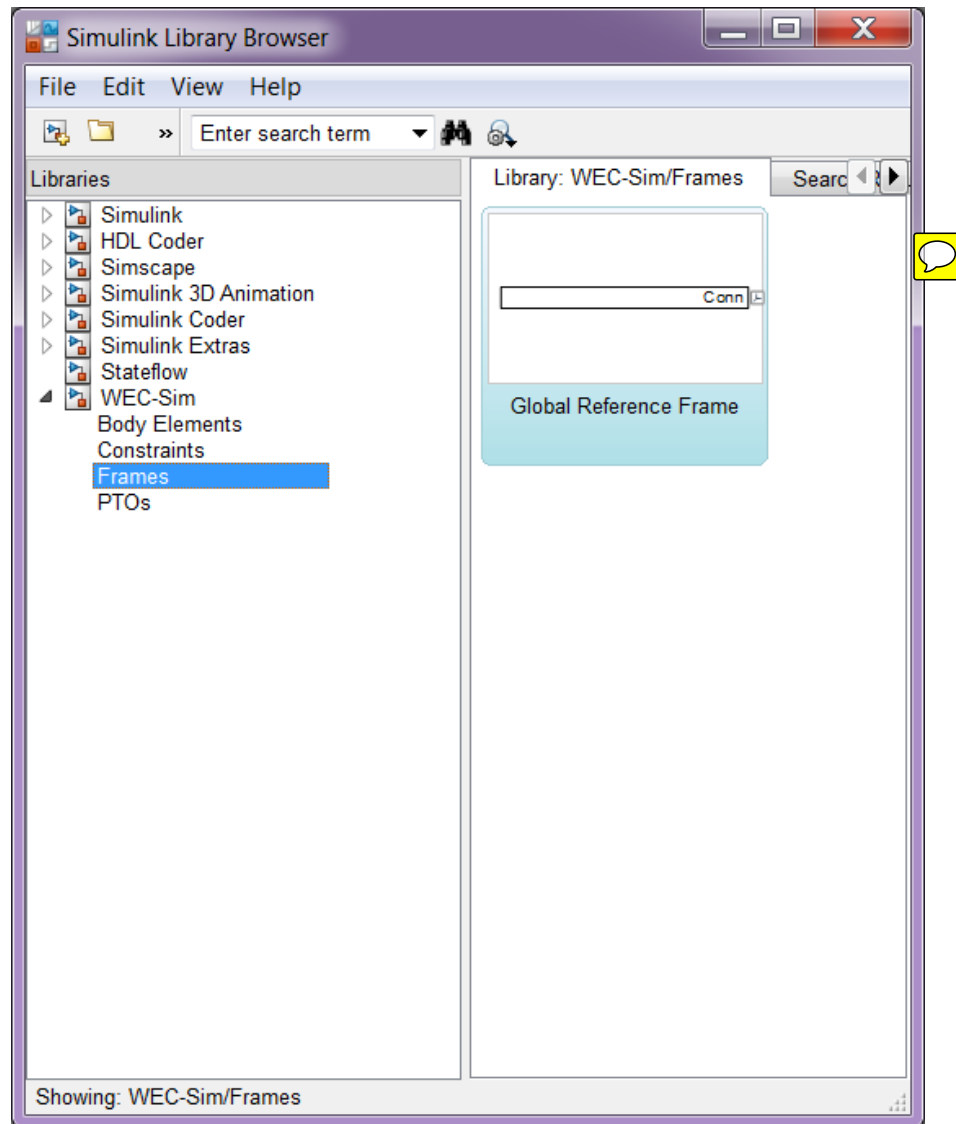


Figure 4.2: Frames sublibrary

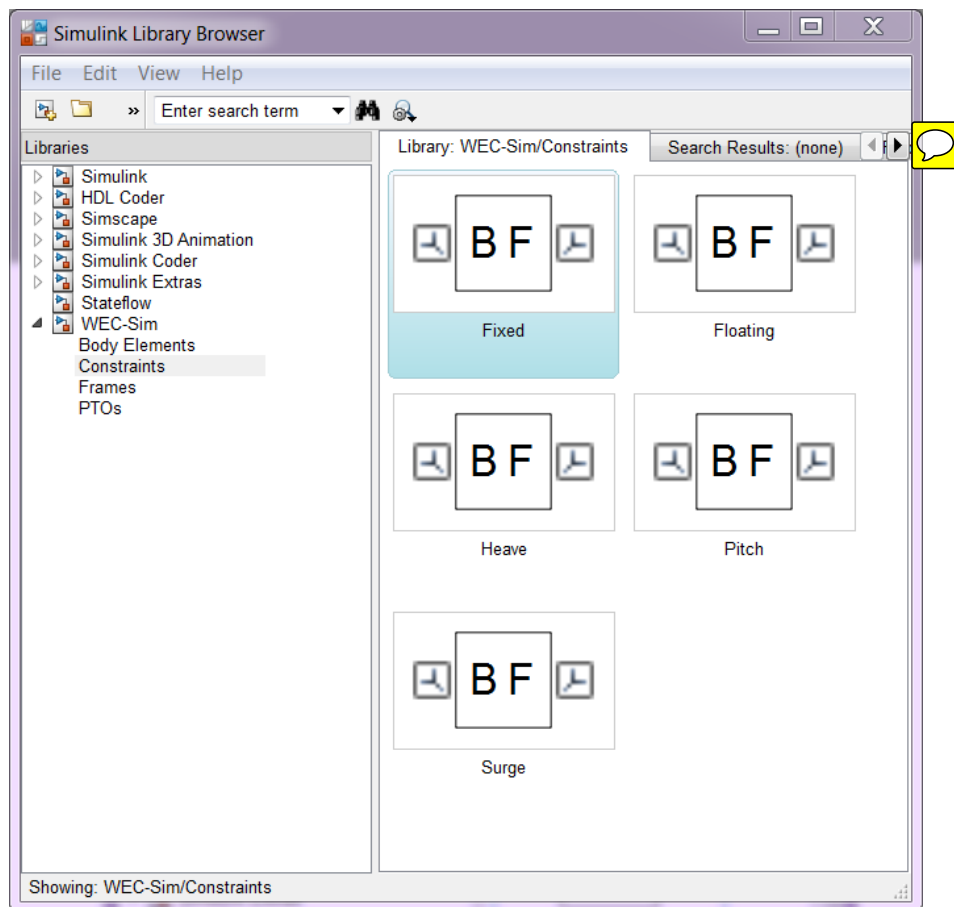


Figure 4.3: Constraints sublibrary

4.4.2 Heave Block

The **Heave** block constrains the motion of the follower relative to the base to be along the Z-axis. In the case of the base connected to the **Global Reference Frame** (seabed), the body is allowed to move only in the vertical (Z) direction. In the case of the **Heave** block connecting two bodies, the relative motion of the two bodies is constrained to be only along their Z-axes. The Z-axis of the follower and base will always be parallel and their perpendicular distance will be constant. The actual direction of movement of the follower depends on the orientation of the base.

4.4.3 Surge Block

The **Surge** block constrains the motion of the follower relative to the base to be along the X-axis. In the case of the base connected to the **Global Reference Frame** (seabed), the body is allowed to move only in the horizontal (X) direction. In the case of the **Surge** block connecting two bodies, the relative motion of the two bodies is constrained to be only along their X-axes. The X-axis of the follower and base will always be parallel and their perpendicular distance will be constant. The actual direction of movement of the follower depends on the orientation of the base.

4.4.4 Pitch Block

The **Pitch** block constrains the relative motion between the follower and the base to be pitch rotation only (about the Y-axis). The distance from both body-fixed coordinate systems to the point of rotation stays constant. The orientation of both body-fixed Y-axes also stays constant. The point about which the rotation occurs must be specified as the constraint's location in the input file.

4.4.5 Fixed Block

The **Fixed** block is a rigid connection that constrains all motion between the base and follower. It restricts translation in the X- and Z-axis, and rotation about the Y-axis. Its most common use is for a rigid body fixed to the seabed.

4.5 PTOs Sublibrary

The **PTOs** sublibrary, shown in Figure 4.4, is used to simulate simple power take-off (PTO) systems and to restrict relative motion between multiple bodies or between one body and the seabed. The **PTO** blocks can simulate simple PTO systems by applying a linear stiffness and damping to the connection. Similar to the **Constraints** blocks, the **PTO** blocks have a base (B) and a follower (F). The user has to name each PTO block 'pto(i)' where $i=1,2,\dots$, and then define their properties in the input file.

4.5.1 Translation PTO (Local Z) Block

The **Translation PTO (Local Z)** is identical to the **Heave** constraint, but applies a linear stiffness and damping coefficient to the connection. The user has to name the PTOs as described earlier. The user then specifies the stiffness coefficient, k (N/m), and damping coefficient, c (Ns/m) in the input file.

4.5.2 Translation PTO (Local X) Block

The **Translation PTO (Local X)** is identical to the **Surge** constraint, but additionally applies a linear stiffness and damping coefficient to the connection. The user has to name the PTOs as described earlier. The user then specifies the stiffness coefficient, k (N/m), and damping coefficient, c (Ns/m) in the input file.

4.5.3 Rotational PTO (Local RY) Block

The **Rotational PTO (Local RY)** is identical to the **Pitch** constraint, but additionally applies a linear rotational stiffness and damping coefficient to the connection. The user has to name the PTOs as described earlier. The user then specifies the stiffness coefficient, k (Nm/rad), and damping coefficient, c (Nms/rad) in the input file.

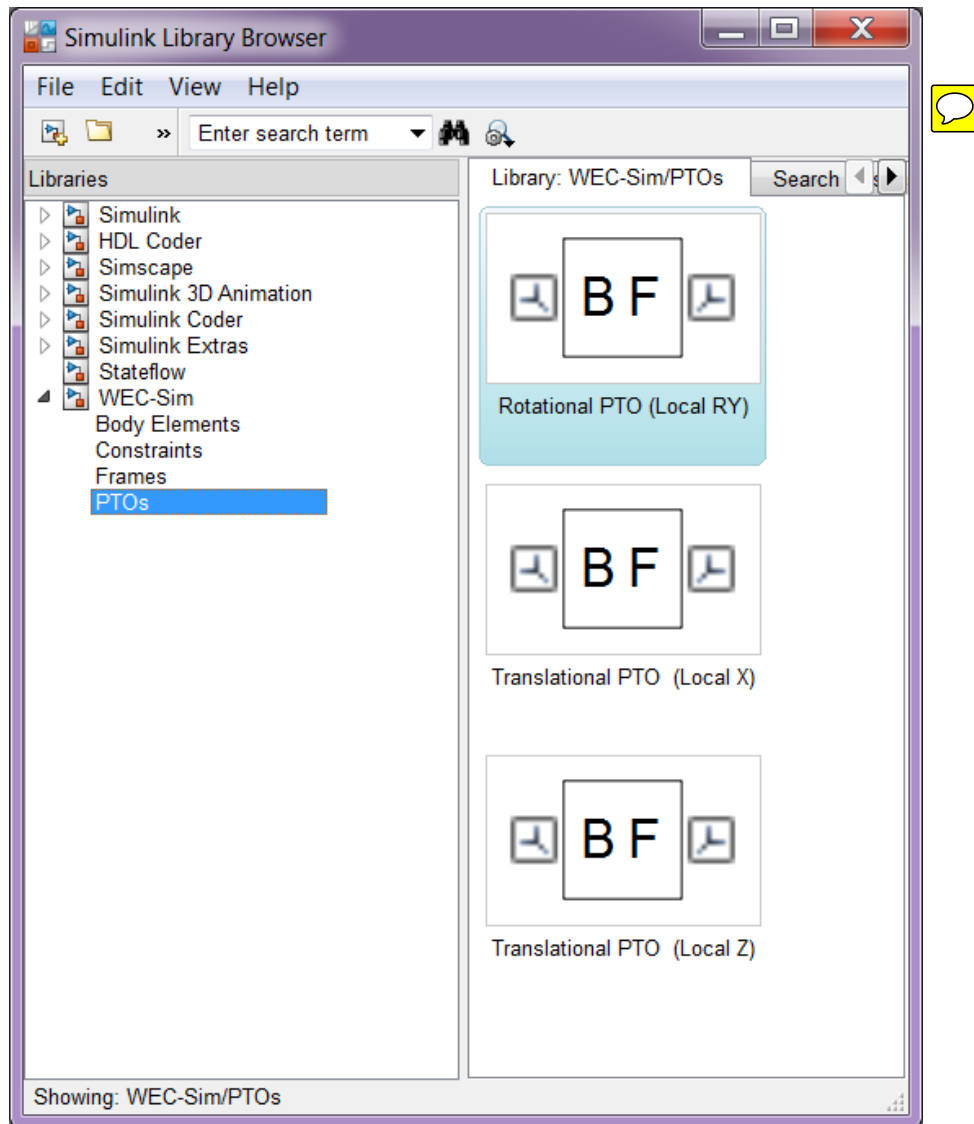


Figure 4.4: PTOs sublibrary

4.6 A Note on SimMechanicsTM and Coordinate Systems

In order to understand how the multibody solver, SimMechanicsTM, works it is important to understand how it uses frames (coordinate systems). The *SimMechanicsTM User's Guide* explains the use of frames in detail, in Chapter 1 *Spatial Relationships*.

The SimMechanicsTM Mechanics Explorers can be utilised to visualize the different frames and the constraints applied to them. Figure 4.5 shows the Mechanics Explorers with frames and center of gravity turned on. The menu on the left can also be useful in identifying specific frames.

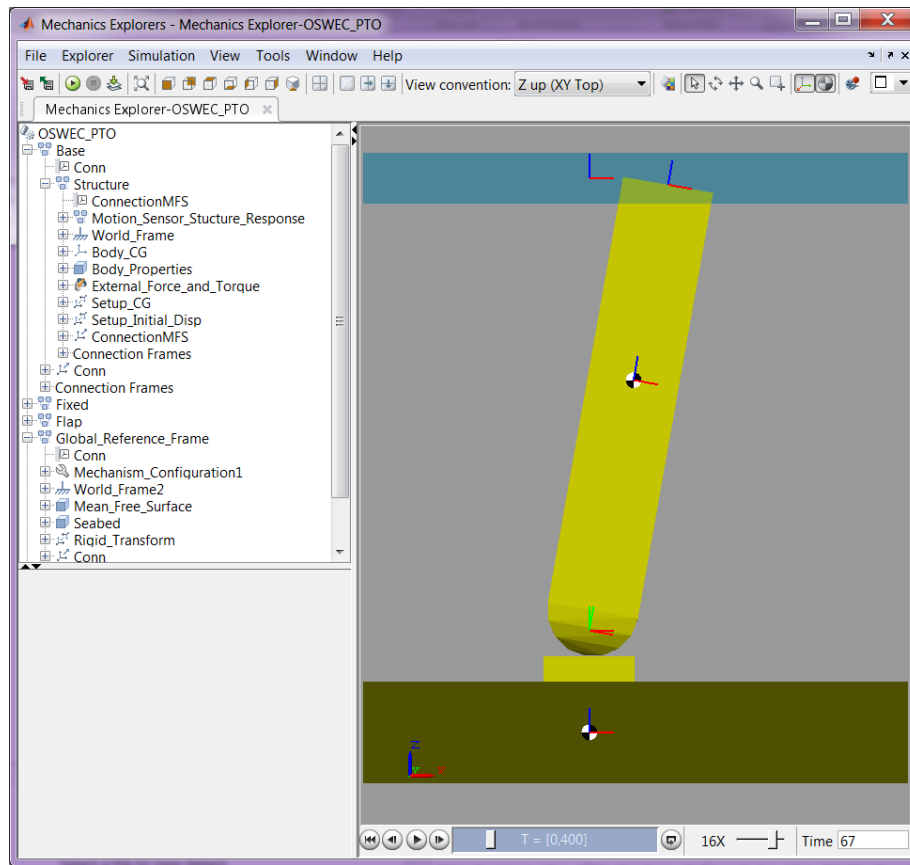


Figure 4.5: SimMechanicsTM Explorer view of an OSWEC

Figure 4.5 also shows the default WEC-Sim coordinate system. Even though the

geometry and visualization are three-dimensional, a WEC-Sim simulation is really only two-dimensional. This is because WEC-Sim currently does not allow multi-directional waves. The default WEC-Sim coordinate system is with the X axis in the direction of wave propagation, and the Z axis in the vertical upwards direction. The Y axis direction is defined by the right-hand-rule. Translational motion only occurs in the XZ plane, and rotation only occurs about the Y axis.

In some situations you might have to use SimMechanicsTM blocks not included in the WEC-Sim Library. One commonly used block is the **Rigid Transform** which can be used to rotate the frames on PTOs, constraints, and bodies. This is also explained in the *SimMechanicsTM User's Guide* Chapter 1.

Chapter 5

Code Structure and Input Parameters

This section describes the WEC-Sim source code structure. For the purposes of this document, we define the the source code as the MATLAB m-files that read the user input data, perform pre-processing calculations that take place before the Simulink/SimMechanics time-domain simulations are performed. The `wecSim` executable, which is part of the source code is also responsible for launching Simulink/SimMechanics time-domain simulations.

5.1 Units

All units within WEC-Sim are in the MKS (meters-kilograms-seconds system) and angular measurements are specified in radians unless otherwise specified.

5.2 WEC-Sim Input File

A WEC-Sim input file is required for each run. ~~It has to be placed~~ inside the case directory for the run and has to be named as `wecSimInputFile.m`. Figure 5.1 shows an example of the input file for a two-body point absorber. The input file contains information needed to run WEC-Sim simulations. ~~Specifically, it serves four primary functions;~~

Specification of Simulation Parameters

Within the input file the user specifies simulation parameters, such as simulation duration, timestep, etc.. As shown in Figure 5.1, simulation parameters are specified within the `simu` variable, which is a member of the `simulationClass`. The user also specifies the name of the Simulink/SimMechanics WEC model within the `simu` variable. The simulation parameters that are available for the user to set within the `simulationClass` are described in more detail in Section 5.3.1.

Specification of Body Parameters

WEC-Sim assumes that every WEC device is comprised of rigid bodies that are exposed to wave forces. For each body, the user must specify body properties within the `body` variable in the input file, including mass, moment of inertia, center of gravity, and the WAMIT files that describe the hydrodynamic properties (see Figure 5.1). The body parameters that are available for the user to set within the `bodyClass` are described in more detail in Section 5.3.2.

Specification of Wave Parameters

Within the input file the user must provide information on the wave condition for the simulation. The user must specify the wave condition within the `waves` variable in the input file, including wave height, wave period and wave type. The options that are available for the user to set within the `waveClass` are discussed in more detail in Section 5.3.3.

Specification of Power Take-Off and Constraint Parameters

Power take-off elements and constraint connect WEC bodies to each other (and possibly the seabed). The properties of the PTO and Constraint are defined within `pto` variable and `constraint` variable, respectively. The options that are available for the user to specify are described in more detail in Sections 5.3.4 and 5.3.5 .

```
Editor - /Users/yyu/Dropbox/WorksNREL/GitHub/WEC-Sim/Tests/RM3_2Body_3DOF_0PTO/w
wecSim.m x waveSetup.m x waveClass.m x bodyClass.m x wecSimInputFile.m* x
1 % Simulation Data
2 - simu.startTime=0;
3 - simu.endTime=400;
4 - simu.dt = 0.1;
5 - simu.simMechanicsFile = 'RM3_2Body_3DOF_0PTO.slx';
6 - simu.mode='normal';%'normal','accelerator','rapid-accelerator'
7 - simu.explorer='on';
8 - simu.hydroDataWamit=1;
9
10 % Wave Information
11 - waves.H = 2.5;
12 - waves.T = 8;
13 - waves.type = 'regular';
14
15 % Body Data
16 - body(1) = bodyClass('Float');
17 - body(1).hydroDataType = 'wamit';
18 - body(1).hydroDataLocation = './BEM/WAMIT_Run01_Float/buoywamit.out';
19 - body(1).mass = 'wamitDisplacement';
20 - body(1).cg = 'wamit';
21 - body(1).momOfInertia = [20907301 21306090.66 37085481.11];
22 - body(1).geometry = 'geometry/float.stl';
23
24 - body(2) = bodyClass('Spar_Plate');
25 - body(2).hydroDataType = 'wamit';
26 - body(2).hydroDataLocation = './BEM/WAMIT_Run01_Plate/buoywamit.out';
27 - body(2).mass = 'wamitDisplacement';
28 - body(2).cg = 'wamit';
29 - body(2).momOfInertia = [94419614.57 94407091.24 28542224.82];
30 - body(2).geometry = 'geometry/plate.stl';
31
32 % Joint and Constraint Parameters
33 - pto(1) = ptoClass('Joint1');
34 - constraint(1) = constraintClass('constraint1');
35
36
```

Figure 5.1: Example of a WEC-Sim input file for a two-body point absorber.

5.3 WEC-Sim Code

All data that is needed for a WEC-Sim simulation is contained within `simu`, `body`, `waves`, `pto` and `constraint` variables that are instances of the `simulationClass`, `bodyClass`, `wavesClass`, and `jointClass` objects, respectively. The user ~~can interact with~~ these variables within the WEC-Sim input file, `wecSimInputFile.m`, (see Figure 5.1). The remainder of this section describes the data within the WEC-Sim objects and how to interact with the objects to set relevant simulation input parameters. Example of using WEC-Sim to simulate WEC devices and their input files are **discribed** in Chapter 6.

5.3.1 `simulationClass`

The `simulationClass` contains the simulation parameters and solver settings needed to execute WEC-Sim. The user can set the relevant simulation properties in the `wecSimInputFile.m`. The user MUST specify the name of the Simulink/SimMechanics WEC model, which can be set by entering

```
simu.simMechanicsFile='<newWecModelName>.slx';
```

The user has the option of accepting the default values for all the other simulation parameters by doing nothing. Available simulation properties and the default values for each, shown in Figure 5.2, can be explored further by typing `doc simulationClass` from within the MATLAB Command Window.

The users can also specify simulation parameters and solver settings in the input file to overwrite the default values. For example, the end time of a simulation can be set by entering

```
simu.endTime = <user specified end time>
```

~~Note that~~ properties that have default values of 'NOT DEFINED' must be specified by the user within the input file. Dependent variables cannot be set by the user and are calculated internally by WEC-Sim.

Property Summary

CfTime	Convolution integral time (default = 60 s)
Clkt	Calculate the number of convolution integral timesteps (default = dependent)
CTTime	Convolution integral time series (default = dependent)
caseDir	WEC-Sim case directory (default = 'NOT DEFINED')
domainSize	Size of free surface and seabed. This variable is only used for visualization (default = 200 m)
dt	Simulation time step (default = 0.1 s)
endTime	Simulation end time (default = 500 s)
explorer	SimMechanics Explorer 'on' or 'off' (default = 'on')
g	Acceleration due to gravity (default = 9.81 m/s)
hydroDataWamit	Equal to 1 if data from 1 WAMIT file, Equal to 0 if data from more than 1 input file (default = 0)
inputFile	Name of WEC-Sim input file (default = 'wecSimInputFile')
maxIt	Total number of simulation timesteps (default = dependent)
mode	'normal', 'accelerator', 'rapid-accelerator' (default = 'normal')
numConstraints	Number of constraints in the wec model (default = 'NOT DEFINED')
numFreq	Number of wave frequencies for interpolation (default = 201)
numPts	Number of power take-off elements in the model (default = 'NOT DEFINED')
numWecBodies	Number of hydrodynamic bodies that comprise the WEC device (default = 'NOT DEFINED')
outputDir	Data output directory name
rampT	Ramp time for wave forcing (default = 100 s)
rho	Density of water (default = 1000 kg/m ³)
simMechanicsFile	Simulink/SimMechanics model file (default = 'NOT DEFINED')
solver	PDE solver used by the Simulink/SimMechanics simulation (default = 'ode4')
startTime	Simulation start time (default = 0 s)
version	WEC-Sim version
zeroVel	Matrix of zeros with a size of 6,obj.Clkt+1 (default = dependent)

Figure 5.2: Data contained within the `simulationClass`.

5.3.2 bodyClass

The `bodyClass` object contains the mass and hydrodynamic properties of each body that comprises the WEC device being simulated. Each body must have a `bodyClass` initiated in the input file. It is recommended that these body objects are named `body(<body number>)` as shown in the input file examples in this document (e.g. Figure 5.1). Each body object MUST be initiated by entering

```
body(<body number>)=bodyClass('<body name>');
```

The mass and hydrodynamic properties can be specified after the body object is initialized for each body. Note that the hydroDataType, hydroDataLocation, mass, cg, momOfInertia and geometry has to be specified for each body. The users have the option of accepting the default values for all the other body parameters by doing nothing or specify their own value. For example, the viscous drag coefficient in heave can be set by entering the user given value as a vector in the input file,

```
body(<body number>).cd= [0 0 1.3 0 0 0];
```

The options available within the bodyClass are shown in Figure 5.3.

Property Summary

cd	Drag coefficient (format [Cd_x Cd_y Cd_z Cd_rotationX Cd_rotationY Cd_rotationZ], default = [0 0 0 0 0 0])
cg	Center of gravity (format: [x y z])
cgCalcMethod	Method of setting the body cg (options: 'user' or 'wamit', default = 'wamit')
characteristicArea	Characteristic area for viscous drag calculations (format [Area Area Area Area Area Area], default = [0 0 0 0 0 0]).
fixed	Default is 0. If the value is equal to 1, it means the body is fixed to the ground and the mass, MOI and CG will equal to the default value and are meaning less in the calculation.
geom	Structure that defines the geometry for visualization and non-linear buoyancy and excitation force calculations
geometry	Location of the .stl file that defines the geometry of the body (default = 'NOT DEFINED')
hydro	Structure that contains the hydrodynamic data for the body (This structure is currently populated by reading WAMIT data)
hydroDataLocation	Location of the wamit .out file (default = 'NOT DEFINED')
hydroDataType	Code used to generate hydrodynamic coefficients (options: 'wamit', default = 'wamit')
hydroForce	Structure used to calculate hydrodynamic forces acting on the body
initAngularDispAngle	Initial displacement of cog - Angle of rotation - used for decay tests (format: [radians], default = 0)
initAngularDispAxis	Initial displacement of cog - axis of rotation - used for decay tests (format: [x y z], default = [1 0 0])
initLinDisp	Initial displacement of center of gravity - used for decay tests (format: [displacement in m], default = [0 0 0])
mass	Body mass (options: 'wamitDisplacement' or [mass], default = 'wamitDisplacement')
massCalcMethod	Method of calculating the center of gravity (default = dependent)
momOfInertia	Moment of inertia (format: [Ixx Iyy Izz], default = [999 999 999])
mooring	Data structure that contains the mooring stiffness and damping matrices
name	Name of the body used (default = 'NOT DEFINED')
storage	Structure to store simulation data for post processing

Figure 5.3: Data contained within the bodyClass.

5.3.3 waveClass

The **waveClass** contains all the information that defines the wave conditions for the WEC-Sim simulation. WEC-Sim provides the functionality to simulate WEC devices using several different wave fields. Specifically, the **waveClass** supports the different types of waves, as listed in Table 5.1

Table 5.1: Default files names and locations

Type of Wave	Required Input Parameters	Comments
noWave	waves.type; waves.noWaveHydrodynamicCoeffT	Free decay test with constant radiation added-mass and damping
noWaveCIC	waves.type;	Free decay test with convolution integral
regular	waves.H; waves.T; waves.type	Sinusoidal Steady-State Reponse Scenario
regularCIC	waves.H; waves.T; waves.type	Use convolution integral Formula
irregular	waves.H; waves.T; waves.type; waves.spectraType	Use for Irregular wave simulation
irregularPRE	waves.H; waves.T; waves.type; waves.spectraType	Use for irregular wave simulation with predefined wave phases

No waves (noWave): Wave type for running simulations without waves using constant added mass and radiation damping coefficients for the body. Accordingly, the user must still run WAMIT before executing WEC-Sim, and the period from which the hydrodynamic coefficients are selected must be specified by setting the **waves.noWaveHydrodynamicCoeffT** variable. This option is typically used for decay tests for comparison with analytical solutions that use constant radiation added-mass and damping coefficients.

No waves with convolution integral calculation (noWaveCIC): The same as **noWave**, except the radiation forces are calculated using the convolution integral and the infinite frequency added-mass .

Regular waves (`waves.type=regular`): Wave type for simulations using regular waves with constant period `wave.T` and wave height `wave.H` specified in the input file. Using this option, we assume the system dynamic response is in sinusoidal steady-state form, where constant added mass and damping coefficients are used and the convolution integral is not used to calculate wave radiation forces.

Regular waves with convolution integral calculations (`waves.type=regularCIC`): The same as `regular` except the radiation forces are calculated using the convolution integral and the infinite frequency added-mass.

Irregular waves (`waves.type=irregular`): Wave type for simulations using irregular waves with peak period `wave.T` and significant wave height `wave.H` specified in the input file.

Irregular waves with a predefined phase angle for different wave frequency components (`waves.type=irregularPRE`): Same as `irregular`, except the phase of the waves is defined by the user through the `phaseRand` variable.

Note that for irregular wave simulations, the available wave spectrum options are listed in Table 5.2. As always, typing `doc waveClass` from the MATLAB Command Window provides more information on the class functionality and the available wave properties are shown and described in Figure 5.4.

Table 5.2: Available wave spectrum options in WEC-Sim

Wave Spectrum Type	Input File Parameter
PiersonMoskowitz	<code>waves.spectraType='PM'</code>
Bretschneider	<code>waves.spectraType='BS'</code>
JONSWAP	<code>waves.spectraType='JS'</code>
User Defined	<code>waves.spectraType='Imported'</code>

Property Summary


A	Wave amplitude
H	Wave height
Hs	Significant wave height
I	Wave period
Tp	Wave peak period
bemFreq	frequencies from WAMIT
customSpectrum	Custom wave spectrum
customTimeSeries	Custom wave time series
numFreq	Number of wave frequencies
phaseRand	Wave phase random seed number
spectrumDataFile	Data file that contains the spectrum data file
timeSeriesDataFile	Data file that contains the time series information
type	Wave type
typeNum	Internal WEC-Sim variable
w	Wave frequency
waterDepth	Water depth
waveAmpTime	Internal WEC-Sim variable

Figure 5.4: Data contained within the `waveClass`.

5.3.4 `constraintClass`

The constraint object is used to connect bodies to the Global Reference Frame, which is actively used as the seabed. The constraint variable should be initiated by entering

```
constraint(<constraint number>)=constraintClass('<constraint name>');
```

For rotational constraint (i.e., pitch), the user also need to specify its location  which has a default value of `(constraint(<constraint number>).loc=[0 0 0])`.

5.3.5 ptoClass

The pto object is used the connector between bodies. The pto variable should be initiated by entering

```
pto(<pto number>)=ptoClass('<pto name>');
```

For Rotational PTO (Local RY), the user also need to set its location, which has a default value of `pto(<pto number>).loc=[0 0 0]`. The users also have the option to specify the damping (`pto(<pto number>).c`) and stiffness (`pto(<pto number>).k`) values to represent the PTO system. Both have a default value of 0. The users can specify a damping value in the input file to overwrite the default values,

```
pto(<pto number>).c=<pto damping value>;
```

.

Chapter 6

Applications

In this section of the user manual a description of how to use WEC-Sim to model two different WECs is discussed. The first application models a two-body point absorber WEC and the second application models an oscillating surge WEC.

6.1 RM3 Two-Body Point Absorber

6.1.1 Geometry Definition

As the first application of the WEC-Sim code, the Reference Model 3 (RM3) two-body point absorber design was chosen. While the WEC is free to move in all 6DOF in response to wave motion, power is only captured in the heave direction. The RM3 device was chosen because the design has already been well characterized both numerically and experimentally as a result of the DOE funded Reference Model Project. Additionally, it has relatively simple operating principles, and is representative of WECs currently pursued by the wave energy industry. RM3 is a simple two-body point absorber, consisting of a float and a reaction plate. The full-scale dimensions of the RM3 are shown in Figure 6.1, and the mass properties are defined in Table 6.1.

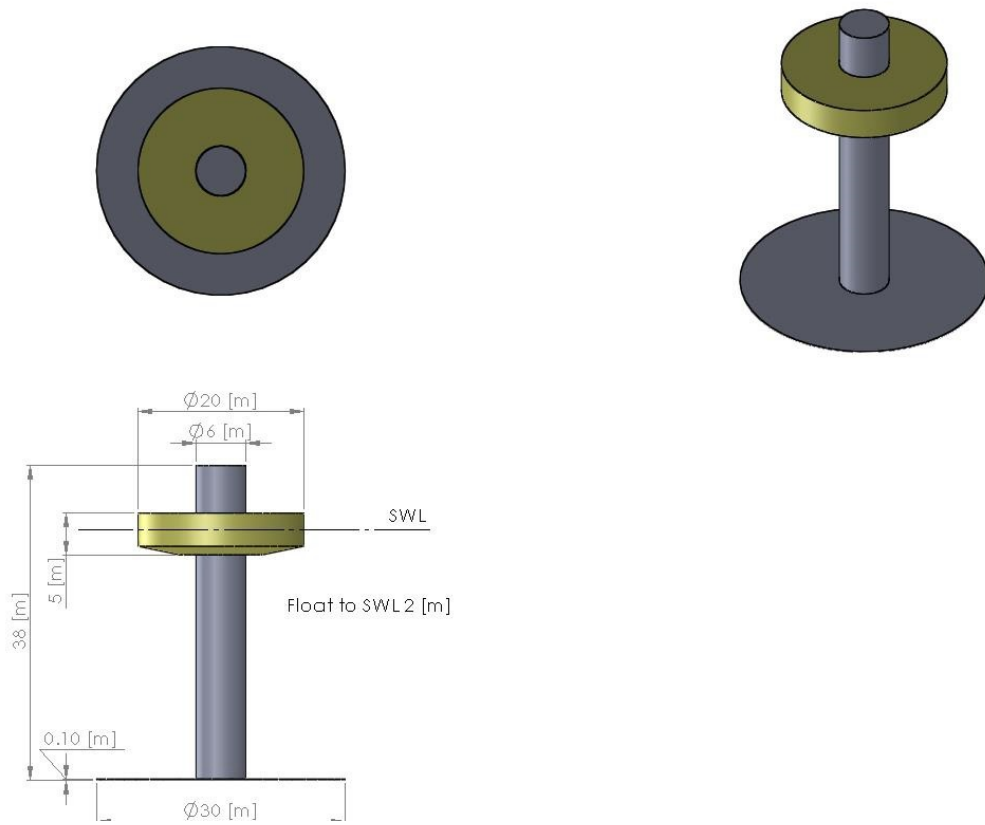



Figure 6.1: RM3 Heaving Two-Body Point Absorber Full-Scale Dimensions

Table 6.1: RM3 Heaving Two-Body Point Absorber Full-Scale Mass Properties

Float Full Scale Properties				
CG [m]	Mass [tonne]	Moment of Inertia [kg-m ²]		
0	727.01	20907301	0	0
0		0	21306090.7	4304.89323
-0.72105		0	4304.89323	37085481.1
Plate Full Scale Properties				
CG [m]	Mass [tonne]	Moment of Inertia [kg-m ²]		
0	878.30	94419614.6	0	0
0		0	94407091.2	217592.785
-21.285		0	217592.785	28542224.8

6.1.2 Modeling RM3 in WEC-Sim

In this section of the user manual, a step by step tutorial on how to set up and run the RM3 simulation in WEC-Sim is provided. **A supplemental RM3 WEC-Sim Tutorial Video has also been created to demonstrate how to setup and run the RM3 simulation in WEC-Sim.** 

As described in the Getting Started section, all WEC-Sim models consist of a MATLAB input file (titled `wecSimInputFile.m` ~~by default~~), and a Simulink model file (`*.slx`). In order to run the RM3 simulation, the user also needs to provide results from the WAMIT Boundary Element Method (BEM) solver to populate the WEC-Sim hydrodynamic coefficients. For the RM3 run, this file corresponds to the `buoywamit.out` file, contained in the `wamit` subfolder. Additionally, the user needs to specify the 3D geometry file in the form of a `*.stl` file about the center of gravity for the WEC-Sim visualizations. For the RM3 run ~~consisting of a buoy and a spar-plate~~, these files correspond to the `float.stl` and `plate.stl` files respectively, which are located in the `geometry` subfolder.


RM3 Simulink Model File

The first step to initiate a WEC-Sim simulation is to populate the Simulink model file by dragging and dropping blocks from the WEC-Sim library into the `*.slx` file, see WEC-Sim Library Structure section.

Step 1:

Two **Rigid Body** blocks from the WEC-Sim library should be placed in the Simulink model file, ~~since the RM3 WEC has two rigid bodies~~, as shown in Figure 6.2.

Step 2:

Double click on the **Rigid Body** block, and rename the instances of the body. The first body should be titled `body(1)`, and the second body should be titled `body(2)`. Additional properties of these body blocks will be defined in the following RM3 MATLAB Input File section. 

Step 3:

Once the float and the plate **Rigid Body** blocks are placed, the **Global Reference Frame** should be added to the Simulink model file, as show in Figure 6.3. The global reference frame acts as the seabed to which all other bodies are linked through joints or constraints.

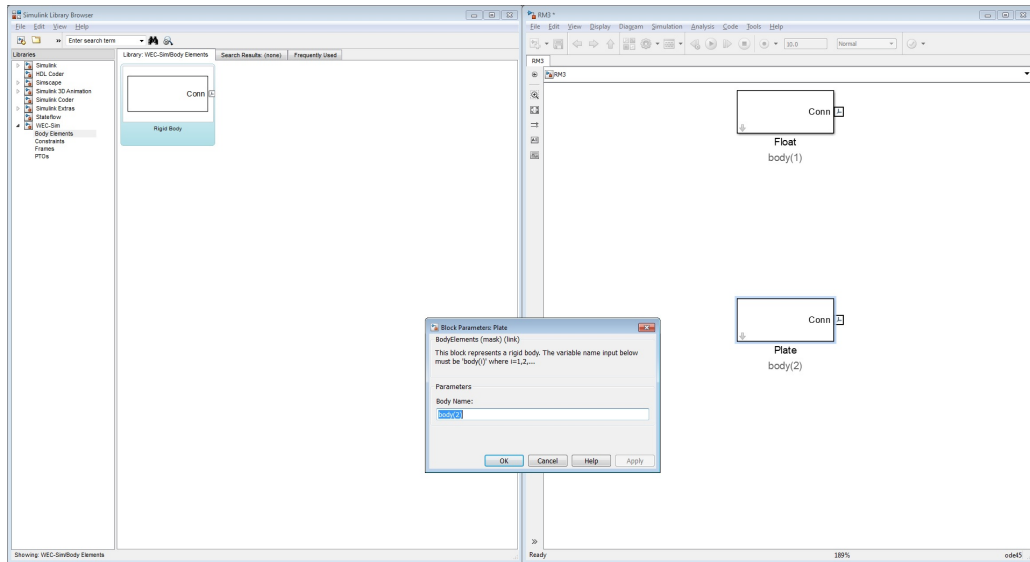


Figure 6.2: Adding 2 Rigid Body blocks to the RM3 WEC-Sim Model

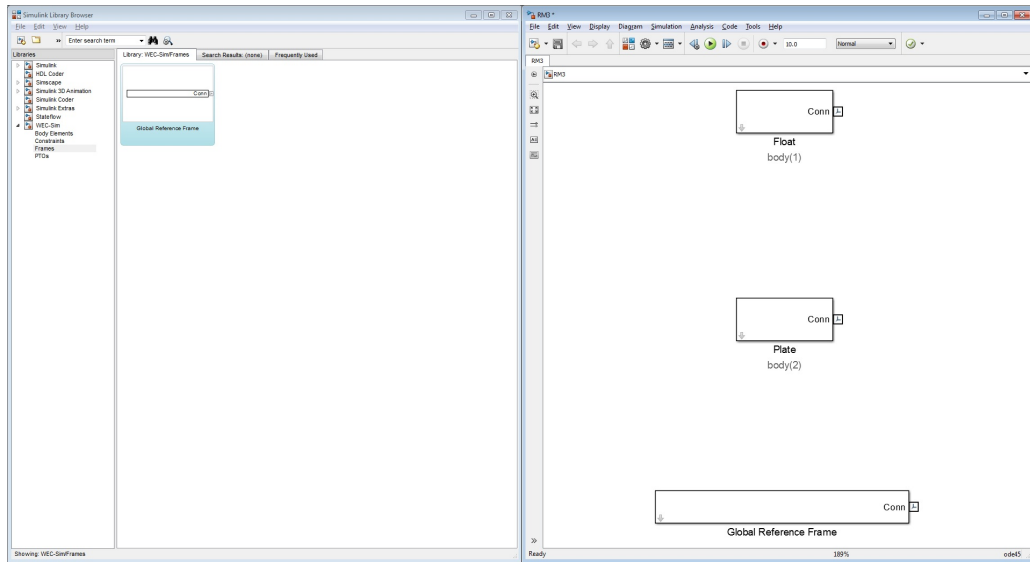


Figure 6.3: Adding Global Reference Frame block to the RM3 WEC-Sim Model

~~The next step is to define the connections between the bodies and their reference frame.~~

Step 4:

Use the **Floating** constraint block to connect the plate to the seabed. This is done because the RM3 is free to move in all 6DOF relative to the global reference frame. Step 4 and 5 connections are shown in Figure 6.4.

Step 5:

~~Then~~ place a **Translational PTO (Local Z)** PTO block to connect the float to the spar. This is done because the float is restricted to heave motion relative to the plate. For the RM3 simulation, the translational PTO is used to model the WEC's PTO as a linear damper, whose parameters are defined in the RM3 MATLAB Input File section.

When setting up a WEC-Sim model, it is important to note the base and follower frames. For example, for the constraint between the plate and the seabed, the seabed should be defined as the base because it is the Global Reference Frame. Similarly, for the PTO between the float and the plate, the plate should be defined as the base.

This completes how to set up the WEC-Sim Simulink model for the RM3 point absorber WEC. In the following section, the WEC-Sim MATLAB input file for the RM3 model will be defined.

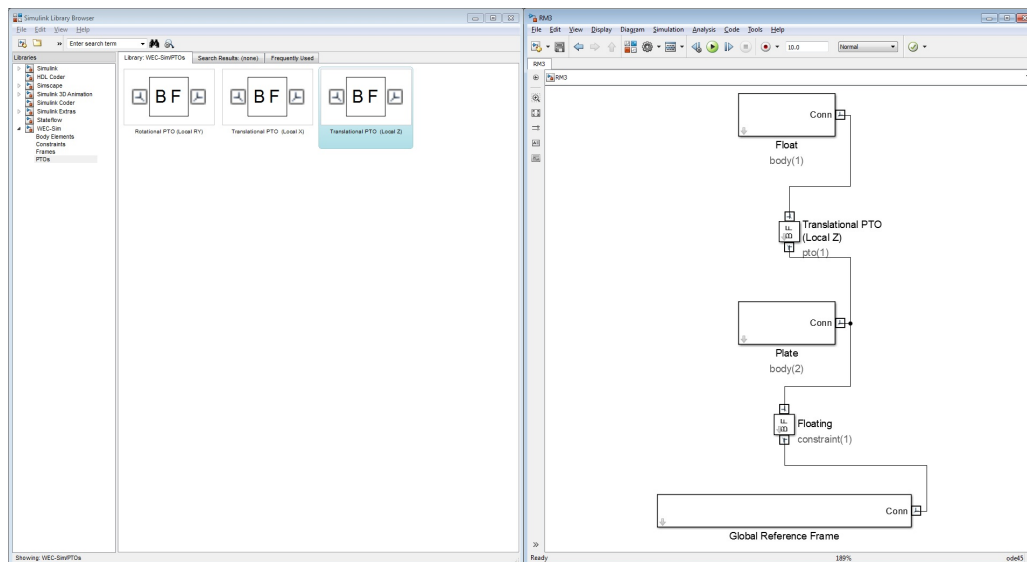


Figure 6.4: Adding PTOs and Constraints to the RM3 WEC-Sim Simulink Model

RM3 MATLAB Input File

In this section, the WEC-Sim MATLAB input file for the RM3 model is defined. Each of the lines are commented to explain the purpose of the defined parameters. For the RM3 model, the user must define the simulation parameters, body properties, PTO, and constraint definitions. Each of the specified parameters for RM3 are defined below.

```
% Simulation Data
simu.startTime=0; % Simulation Start Time [s]
simu.endTime=400; % Simulation End Time [s]
simu.dt = 0.1; % Simulation Delta Time [s]
simu.simMechanicsFile = 'RM3.slx'; % Specify Simulink Model File
simu.mode='normal'; % Specify Simulation Mode
% (normal/accelerator
% /rapid-accelerator)
simu.explorer='on'; % Turn SimMechanics Explorer
% (on/off)

% Wave Information
waves.H = 2.5; % Wave Height [m]
waves.T = 8; % Wave Period [s]
waves.type = 'regular'; % Specify Type of Waves

% Body Data
body(1) = bodyClass('Float'); % Initialize bodyClass for Float
body(1).hydroDataType = 'wamit'; % Specify BEM solver
body(1).hydroDataLocation = ... % Location of WAMIT *.out file
    './wamit/rm3.out';
body(1).mass = 'wamitDisplacement'; % Mass from WAMIT [kg]
body(1).cg = 'wamit'; % Cg from WAMIT [m]
body(1).momOfInertia = ... % Moment of Inertia [kg-m^2]
    [20907301 21306090.66 37085481.11];
body(1).geometry = 'geometry/float.stl'; % Geometry File

body(2) = bodyClass('Spar_Plate'); % Initialize bodyClass for
% Spar/Plate
body(2).hydroDataType = 'wamit'; % Specify BEM solver
body(2).hydroDataLocation = ... % Location of WAMIT *.out file
    './wamit/rm3.out';
body(2).mass = 'wamitDisplacement'; % Mass from WAMIT [kg]
body(2).cg = 'wamit'; % Cg from WAMIT [m]
body(2).momOfInertia = ... % Moment of Inertia [kg-m^2]
    [94419614.57 94407091.24 28542224.82];
```

```

body(2).geometry = 'geometry/plate.stl';    % Geometry File

% PTO and Constraint Parameters
constraint(1) = ...                        % Initialize Constraint Class
    constraintClass('Constraint1');         % for Constraint1

pto(1) = ptoClass('PTO1');                % Initialize ptoClass for PTO1
pto(1).k=0;                               % PTO Stiffness Coeff [N/m]
pto(1).c=1200000;                         % PTO Damping Coeff [Ns/m]

```

RM3 WEC-Sim Model

Once the WEC-Sim Simulink model is set up and the RM3 properties are defined in the MATLAB input file, the user can then run the RM3 model in WEC-Sim. Figure 6.5 shows the final RM3 Simulink model on the left hand side, and the WEC-Sim GUI showing the RM3 during the simulation on the right hand side. For more information on using WEC-Sim to model the RM3 device, refer to [7] and [8].

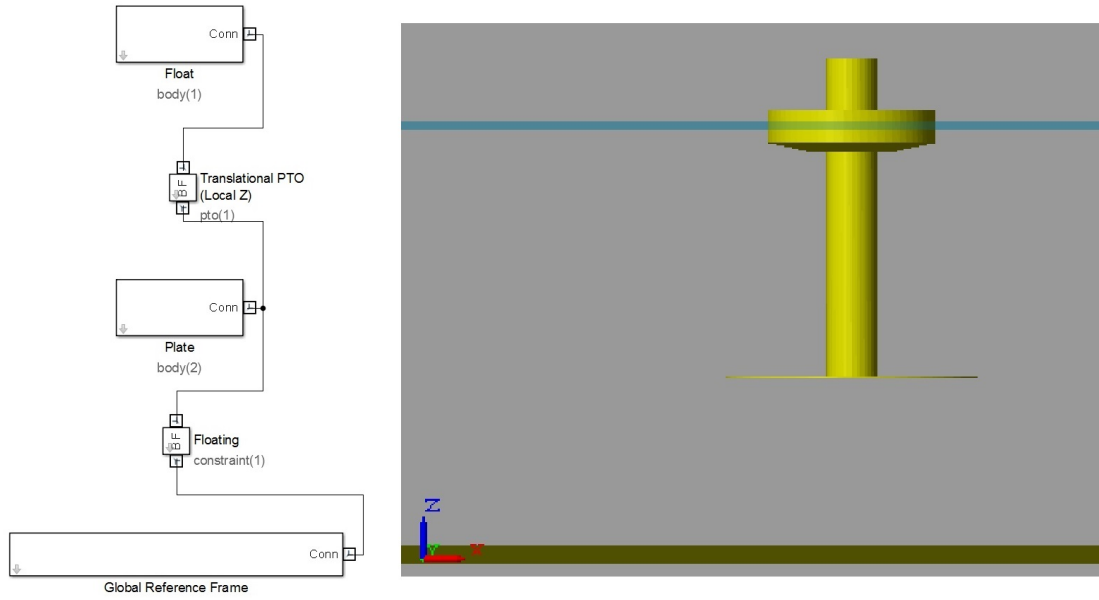


Figure 6.5: RM3 Modeled in WEC-Sim (LHS) and with the GUI (RHS)

6.2 Oscillating Surge-Pitch Device

6.2.1 Geometry Definition

As the second application of the WEC-Sim code, the oscillating surge WEC (OSWEC) device, a two-body pitching Oyster-like design was chosen. The OSWEC was chosen in part because its design is fundamentally different from the RM3. This is critical because WECs span an extensive design space, and it is important to model devices in WEC-Sim that operate under different principles. The OSWEC is fixed to the ground and has a flap that is connected through a hinge to the base that restricts the flap to pitch about the hinge. The full-scale dimensions of the OSWEC are shown in Figure 6.6, and the mass properties are defined in Table 6.2.

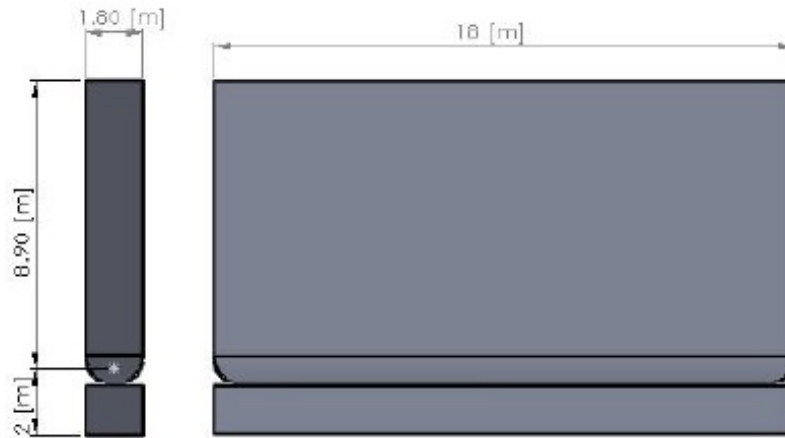


Figure 6.6: OSWEC Pitching Device Full-Scale Dimensions

Table 6.2: OSWEC Pitching Device Full-Scale Mass Properties

Flap Full Scale Properties		
CG [m]	Mass [kg]	Pitch Moment of Inertia [kg-m ²]
0	127,000	1,850,000
0		
-3.9		

6.2.2 Modeling OSWEC in WEC-Sim

In this section of the manual, a step by step tutorial on how to set up and run the OSWEC simulation in WEC-Sim is provided. **A supplemental OSWEC WEC-Sim Tutorial Video has also been created to demonstrate how to set up and run the OSWEC simulation in WEC-Sim.**

As described in the Getting Started section, all WEC-Sim models consist of a MATLAB input file (titled `wecSimInputFile.m` by default), and a Simulink model file (`*.slx`). The WEC-Sim demo folder contains templates for both the MATLAB input file, and for the Simulink model file. In order to run the OSWEC simulation, the user also needs to provide results from the WAMIT Boundary Element Method (BEM) solver to populate the WEC-Sim hydrodynamic coefficients. For the OSWEC run, this file corresponds to the `oswec.out` file, contained in the `wamit` subfolder. Additionally, the user needs to specify the 3D geometry file in the form of a `*.stl` file about the center of gravity for the WEC-Sim visualizations. For the OSWEC run consisting of a base and a flap, these files correspond to the `base.stl` and `flap.stl` files respectively, which are located in the `geometry` subfolder.

OSWEC Simulink Model File

The first step to set up a WEC-Sim simulation is to populate the Simulink model file by dragging and dropping blocks from the WEC-Sim library into the `*.slx` file, see WEC-Sim Library Structure section.

Step 1:

Two **Rigid Body** blocks from the WEC-Sim library should be placed in the Simulink model file, since the OSWEC has two rigid bodies, as shown in Figure 6.7.

Step 2:

Double click on the body block, and rename the instances of the body. The first body should be titled `body(1)`, and the second body should be titled `body(2)`. Additional properties of these body blocks will be defined in the following OSWEC MATLAB Input File section.

Step 3:

Once the base and the flap body blocks are placed, the **Global Reference** block should be added to the Simulink model file, as shown in Figure 6.8. The global reference frame acts as the base to which all other bodies are linked through joints

or constraints.

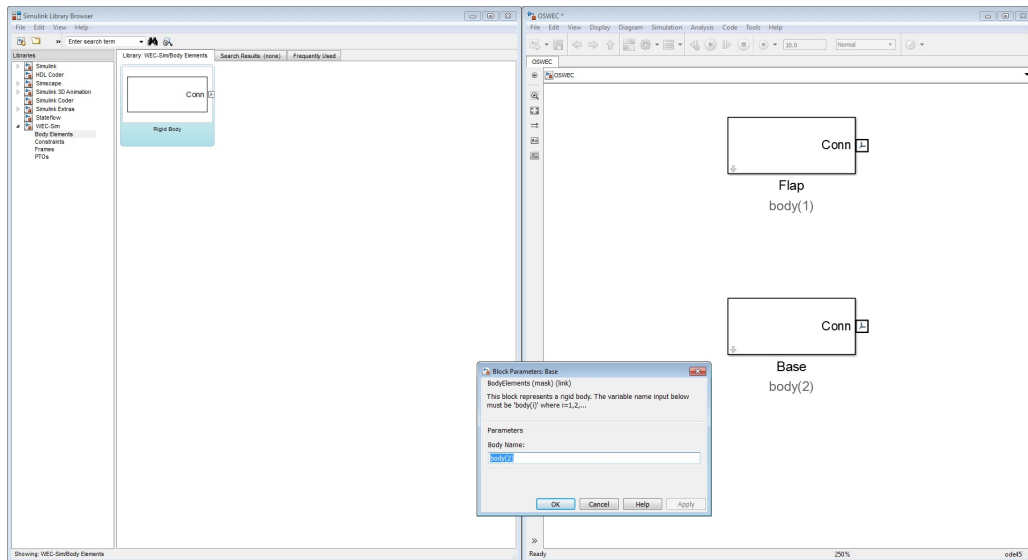


Figure 6.7: Adding 2 Rigid body blocks to the OSWEC WEC-Sim Model

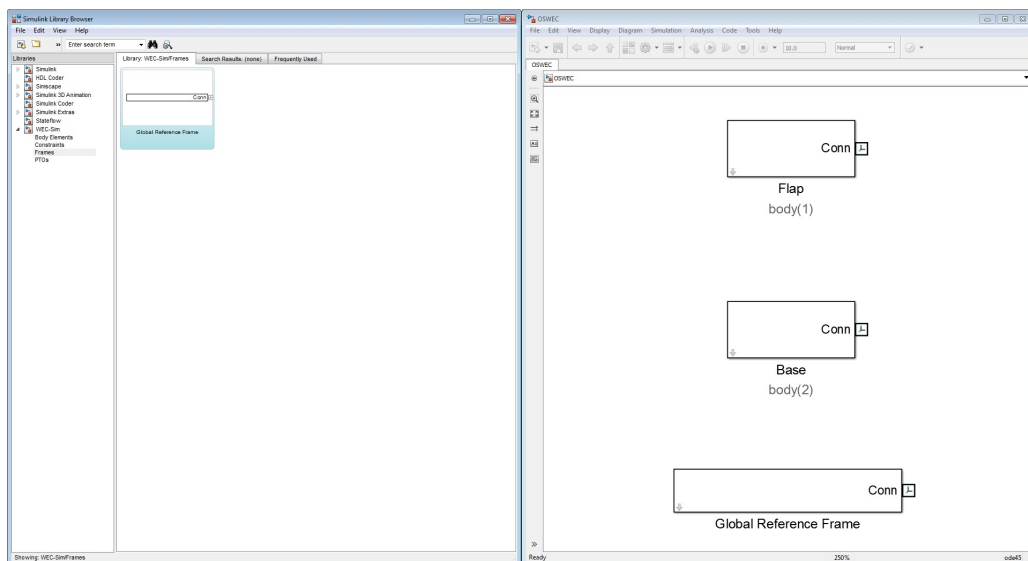


Figure 6.8: Adding a Global Reference Frame block to the OSWEC WEC-Sim Model

~~The next step is to define the connections between the bodies and their reference frame.~~

Step 4:

Place a **fixed** constraint block to connect the base to the seafloor. This is done because the OSWEC base is fixed relative to the global reference frame. Step 4 and 5 connections are shown in Figure 6.9.

Step 5:

Place a **rotational** PTO block to connect the base to the flap. This is done because the flap is restricted to pitch motion relative to the base. For the OSWEC simulation, the rotational PTO is used to model the WEC's PTO as a linear rotary damper, whose parameters are defined in the OSWEC MATLAB Input File section.

When setting up a WEC-Sim model, it is important to note the base and follower frames. For example, for the constraint between the base and the seabed, the seabed should be defined as the base because it is the Global Reference Frame.

This completes how to set up the WEC-Sim Simulink model for the OSWEC. In the following section, the WEC-Sim MATLAB input file for the OSWEC model will be defined.

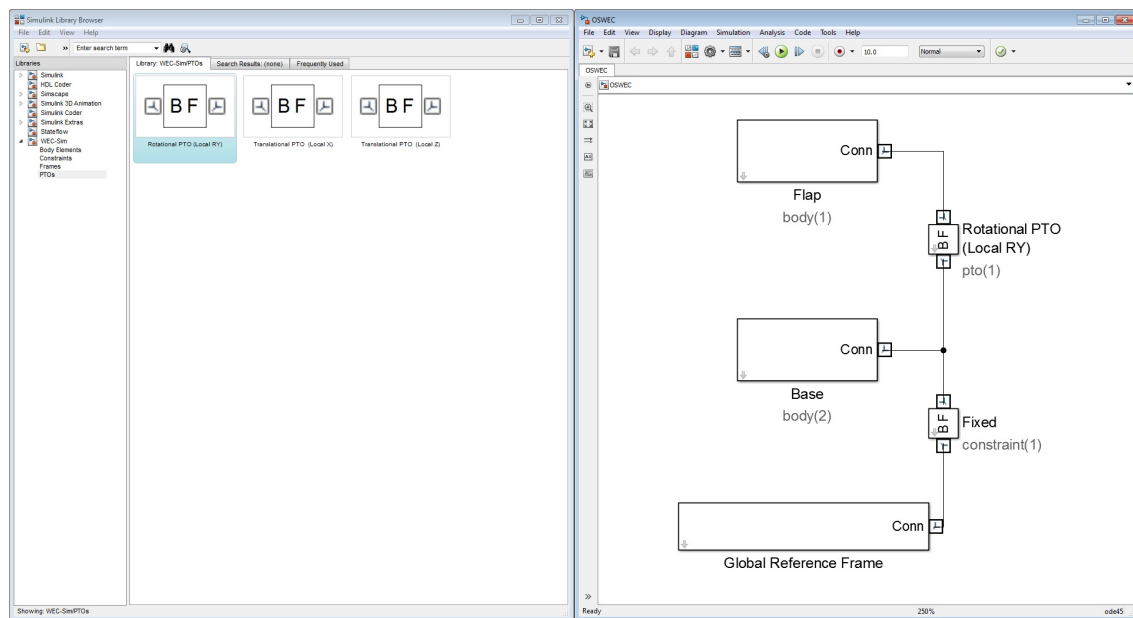


Figure 6.9: Adding PTO and Constraint to the OSWEC WEC-Sim Simulink Model

OSWEC MATLAB Input File

In this section, the WEC-Sim MATLAB input file, `wecSimInputFile.m`, for the OSWEC model is defined. Each of the lines are commented to explain the purpose of the defined parameters. For the OSWEC model, the user must define the simulation parameters, body properties, PTO, and constraint definitions. Each of the specified parameters for OSWEC are defined below.

```
% Simulation Data
simu.startTime=0;
simu.endTime=400;
simu.dt = 0.1;
simu.simMechanicsFile = 'OSWEC.slx';
simu.mode='normal';

simu.explorer='on';

% Wave Information
waves.H = 2.5;
waves.T = 8;
waves.type = 'regular';

% Body Data
body(1) = bodyClass('Flap');
body(1).hydroDataType = 'wamit';
body(1).hydroDataLocation = ...
    'wamit/oswec.out';
body(1).mass = 127000;
body(1).cg = 'wamit';
body(1).momOfInertia = ...
    [1.85e6 1.85e6 1.85e6];
body(1).geometry = 'geometry/flap.stl';

body(2) = bodyClass('Base');
body(2).hydroDataType = 'wamit';
body(2).hydroDataLocation = ...
    'wamit/oswec.out';
body(2).geometry = 'geometry/base.stl';
body(2).fixed = 1;

% PTO and Constraint Parameters
```

```
% Simulation Start Time [s]
% Simulation End Time [s]
% Simulation Delta-Time [s]
% Specify Simulink Model File
% Specify Simulation Mode
%   (normal/accelerator
%   /rapid-accelerator)
% Turn SimMechanics Explorer
%   (on/off)

% Wave Height [m]
% Wave Period [s]
% Specify Type of Waves

% Initialize bodyClass for Flap
% Specify BEM solver
% Location of WAMIT *.out file

% User-Defined mass [kg]
% Cg from WAMIT [m]

% Moment of Inertia [kg-m^2]
% Geometry File

% Initialize bodyClass for Base
% Specify BEM solver
% Location of WAMIT *.out file

% Geometry File
% Creates Fixed Body
```



```

constraint(1)=...                               % Initialize ConstraintClass
    constraintClass('Constraint1');              % for Constraint1

pto(1)=ptoClass('PTO1');                        % Initialize ptoClass for PTO1
pto(1).k=0;                                     % PTO Stiffness Coeff [Nm/rad]
pto(1).c=0;                                     % PTO Damping Coeff [Nsm/rad]
pto(1).loc=[0 0 -8.9];                          % PTO Global Location [m]

```

OSWEC WEC-Sim Model

Once the WEC-Sim Simulink model is set up and the OSWEC properties are defined in the MATLAB input file, the user can then run the OSWEC model in WEC-Sim. Figure 6.10 shows the final OSWEC Simulink model on the left hand side, and the WEC-Sim GUI showing the OSWEC during the simulation on the right hand side. For more information on using WEC-Sim to model the OSWEC device, refer to [8] and [9].

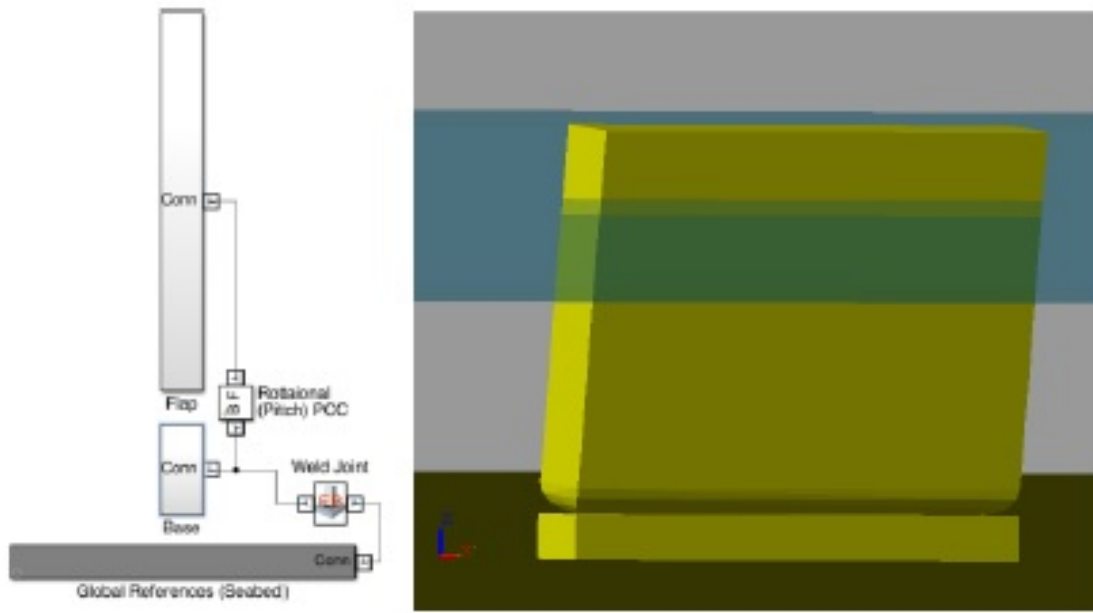


Figure 6.10: OSWEC modeled in WEC-Sim (LHS) and with the GUI (RHS)

6.3 WEC-Sim Test Cases

Provided within the applications folder in the WEC-Sim directory are the two applications of the WEC-Sim code. The first application uses the WEC-Sim code to model the the RM3 WEC, and second application uses the code to model the OS-WEC device. Should the WEC-Sim user make changes to the WEC-Sim code, scripts are provided that compare the previous version of WEC-Sim to the latest run, for debugging purposes. These test cases can be used by running the `RunTestCase.m` script. This script imports the `*.mat` file that contains results from the original WEC-Sim run, and then plots a comparison of the original run to the latest run of WEC-Sim.

6.3.1 RM3 Test Case

For the RM3, the test case is set up for direct comparison using the following model parameters:

Wave Environment

Wave Type = Regular Waves (Sinusoidal Steady-State)

Wave Height H (m) = 2.5

Wave Period T (sec) = 8

WEC-Sim Simulation Settings

Time Marching Solver = Fourth-Order Runge-Kutta Formula

Start Time (sec) = 0

End Time (sec) = 400

Time Step Size (sec) = 0.1

Ramp Function Time (sec) = 100

List of PTO(s)

Number of PTOs = 1

PTO Stiffness (N/m) = 0

PTO Damping (Ns/m) = 1.2E+06

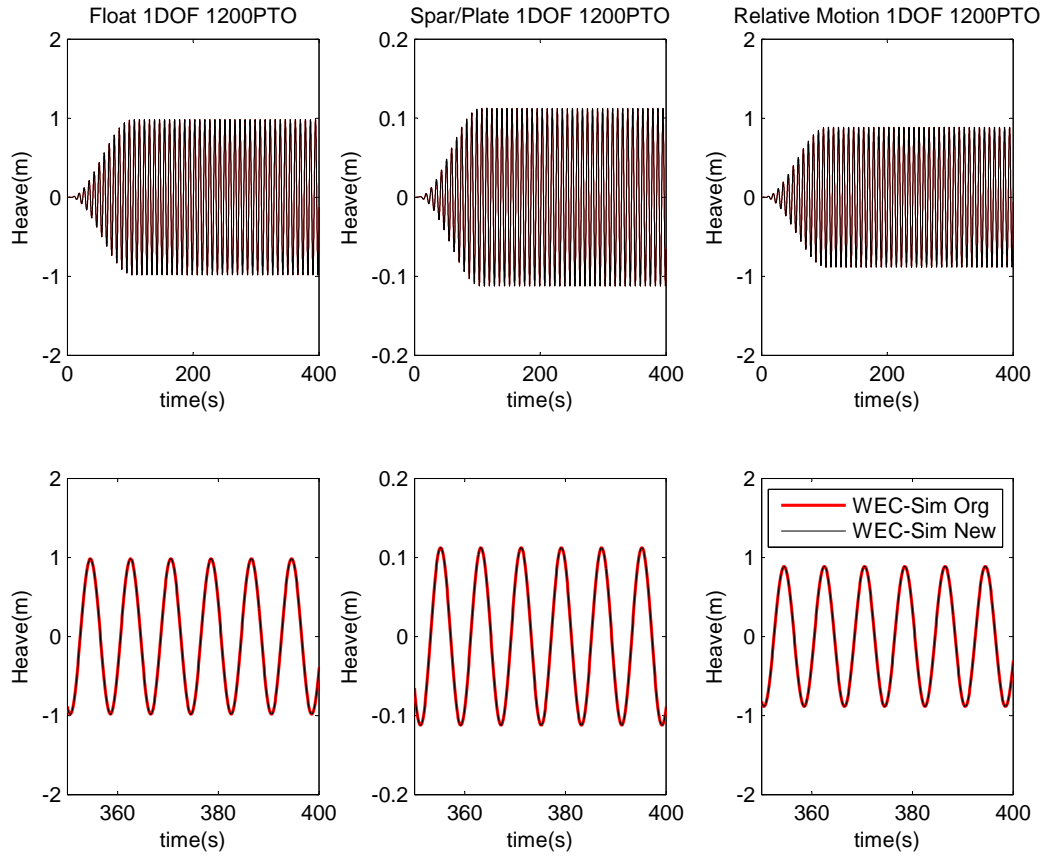


Figure 6.11: WEC-Sim Test Plot for RM3, Top: Full Simulation, Bottom: Last Few Wave Periods

6.3.2 OSWEC Test Case

For the OSWEC, the test case is set up for direct comparison using the following model parameters:

Wave Environment

Wave Type = Regular Waves (Sinusoidal Steady-State)

Wave Height H (m) = 2.5

Wave Period T (sec) = 8

WEC-Sim Simulation Settings

Time Marching Solver = Fourth-Order Runge-Kutta Formula

Start Time (sec) = 0

End Time (sec) = 400

Time Step Size (sec) = 0.1

Ramp Function Time (sec) = 100

List of PTO(s)

Number of PTOs = 1

PTO Stiffness (Nm/rad) = 0

PTO Damping (Nsm/rad) = 0

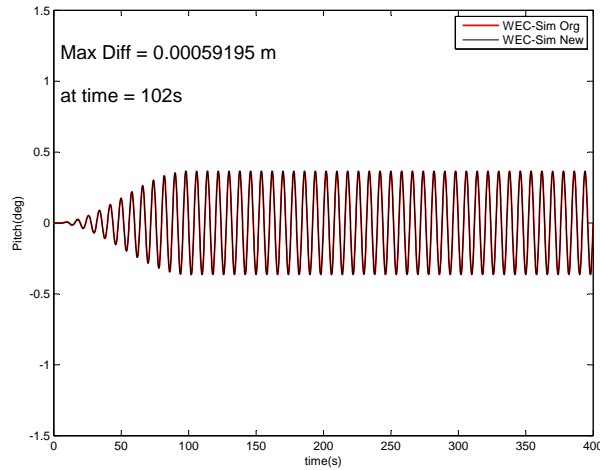


Figure 6.12: WEC-Sim Test Plot for OSWEC Full Simulation

Chapter 7

Troubleshooting

1. ex: body(1) can have multiple
2. Don't press run button in Simulink
3. Moment of inertia matrix only considers the primary diagonal values.
4. Not clear how the seabed/free surface is set-up in the code, make a note that it does not come in .
5. How to perform a coordinate system transformation.
6. How to handle wave heading directionality.
7. Insert global coordinate axis somewhere in document.
8. Default units are the MKS system of units.

Bibliography

- [1] Y. Li and Y.-H. Yu, “A Synthesis of Numerical Methods for Modeling Wave Energy Converter-Point Absorbers,” *Renewable and Sustainable Energy Reviews*, vol. 16, no. 6, pp. 4352–4364, 2012.
- [2] A. Babarit, J. Hals, M. Muliawan, A. Kurniawan, T. Moan, and J. Krokstad, “Numerical Benchmarking Study of a Selection of Wave Energy Converters,” *Renewable Energy*, vol. 41, pp. 44–63, 2012.
- [3] C. Lee and J. Newman, “WAMIT User Manual,” Chestnut Hill, MA, USA, 2006.
- [4] “Nemoh a Open source BEM.” [Online]. Available: <http://openore.org/tag/nemoh/>
- [5] W. Cummins, “The Impulse Response Function and Ship Motions,” David Taylor Model Basin-DTNSRDC, Tech. Rep., 1962.
- [6] “MATLAB.” [Online]. Available: <http://http://www.mathworks.com/>
- [7] Kelley Ruehl, Carlos Michelen, Samuel Kanner, Michael Lawson, and Y. Yu, “Preliminary verification and validation of WEC-Sim, and open-source wave energy converter design tool,” in *Proceedings of OMAE 2014*, San Francisco, CA, 2014.
- [8] Y. Yu, Michael Lawson, Kelley Ruehl, and Carlos Michelen, “Development and demonstration of the WEC-Sim wave energy converter simulation tool,” in *Proceedings of the 2nd Marine Energy Technology Symposium*, Seattle, WA, USA, 2014.
- [9] Y. Yu, Ye Li, Kathleen Hallett, and Chad Hotimsky, “Design and analysis for a floating oscillating surge wave energy converter,” in *Proceedings of OMAE 2014*, San Francisco, CA, 2014.

