WECCCOMP Numerical Model Description

A WEC-Sim model of the WaveStar device is available on the WECCCOMP GitHub repository. This model should be used to develop the WECCCOMP control algorithm.

WEC-Sim code

WEC-Sim (Wave Energy Converter SIMulator) is a publicly available, open-source wave energy converter simulation code jointly developed by the National Renewable Energy Laboratory (NREL) and Sandia National Laboratories (Sandia), funded by the U.S. Department of Energy's Water Power Technologies Office. WEC-Sim can 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 for each body and system constraints.

The WEC-Sim code is publicly available on the WEC-Sim GitHub repository and requires the following MATLAB toolboxes: Simulink, Simscape, and Simscape Multibody. For WECCCOMP, use WEC-Sim v3.0 (or newer) and MATLAB 2017a (or newer and please use Option 1 Clone with GitHub for the WEC-Sim download). For more information about WEC-Sim, refer to the WEC-Sim website.

MathWorks licenses

The WECCCOMP organizers have negotiated 25 temporary free licenses for competitors to use in developing and simulating their controllers. Licenses will be distributed upon registration to WECCCOMP, on a first- come, first-served basis. A maximum of one license will be available to each group/institutional entry. To download MATLAB and associated toolboxes, each participant will need to register and create a MathWorks account; a link to the registration page can be found here. Once participants have registered, send an email notifying the organizers who in return will provide the MATLAB download link. Please follow the download prompts and if there are any issues with the download notify the organizers.

Online forums

Two forums are available to field queries and responses. The WECCCOMP forum will deal with issues related to the competition, while the WEC-Sim forum is for all queries related to WEC-Sim. The WEC-Sim forum is in the public domain to benefit the WEC-Sim user community.

Running the WECCCOMP model

Once the WEC-Sim code and the WECCCOMP repository have been cloned locally, and WEC-Sim has been added to the MATLAB path, the WECCCOMP model is run by navigating to the WECCOMP model's local path and typing `*wecSim*` into the MATLAB Command Window. For more information on how to set up and run WEC-Sim, refer to the WEC-Sim website.

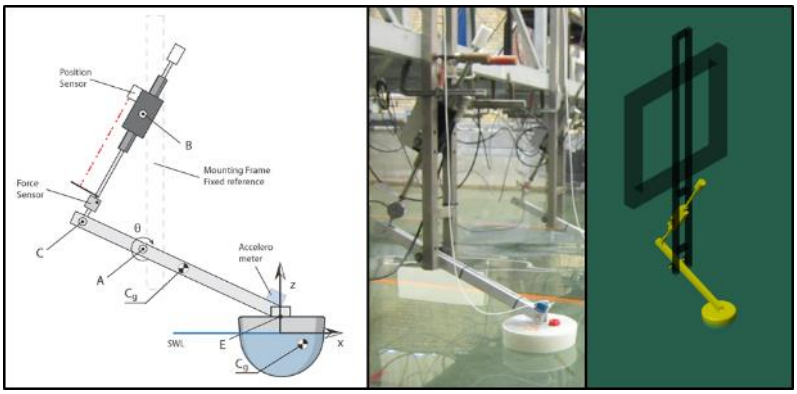

Figure 1. WECCCOMP model experimental setup and visualization in WEC-Sim Simscape Mechanics Explorer

Controller development

The WECCCOMP model consists of a WEC-Sim input file (*wecSimInputFile.m*), a Simulink model file (*WaveStar.slx*, see LHS of Figure 2), a controller initialization script (*controller_init.m*), and a post-processing script (*userDefinedFunctions.m*). The WECCCOMP controller can either be developed using inputs of the linear force and displacement of the motor, or the rotary torque and displacement of the float. The controller linear/rotary implementation may be changed by selecting the appropriate variant subsystem, as shown in Figure 2. The WECCCOMP controller should be developed in the *WaveStar->Linear/Rotary->WECCCOMP Controller* subsystem in the *WaveStar.slx* model.

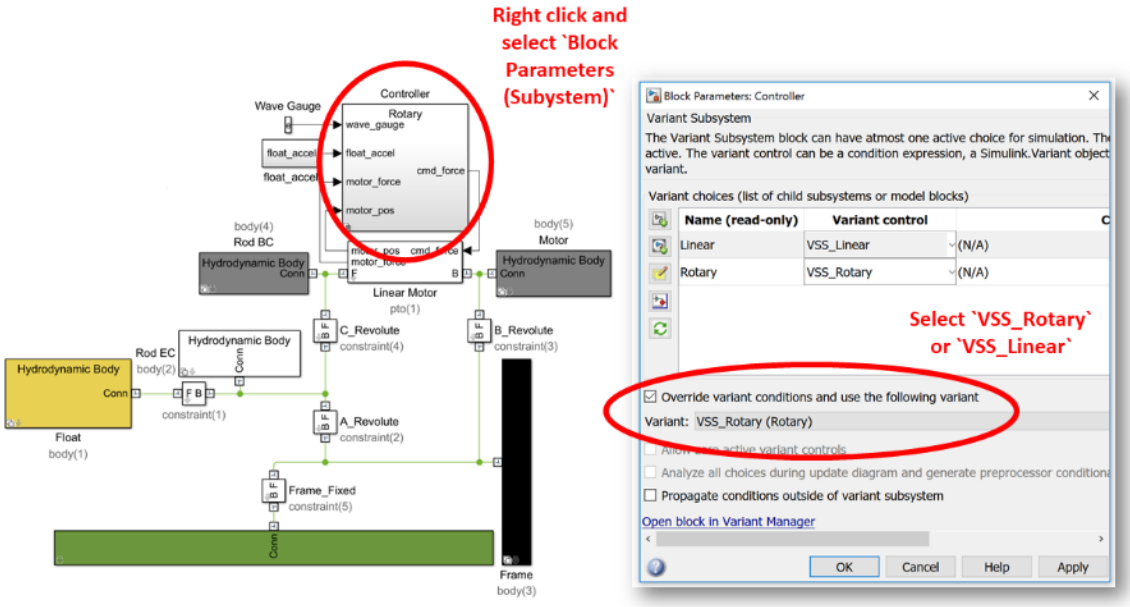

Figure 2. (LHS) WEC-Sim Simulink model (RHS) how to change the `Controller` variant subsystem from rotary to linear

Additionally, contestants will need to modify the *wecSimInputFile.m* and *controller_init.m* scripts to run different wave cases, and initialize their controller. For more information about the WEC-Sim model files and input file options, refer to the WEC-Sim website.
The example depicted in Figure 3 shows the implementation of a spring/damper/mass control system. The controller generates the control signal "cmd_torque_rot" by adding a signal proportional to the position, a signal proportional to the velocity (which is obtained as the low pass filtered derivative of the position), and another signal proportional to the acceleration.

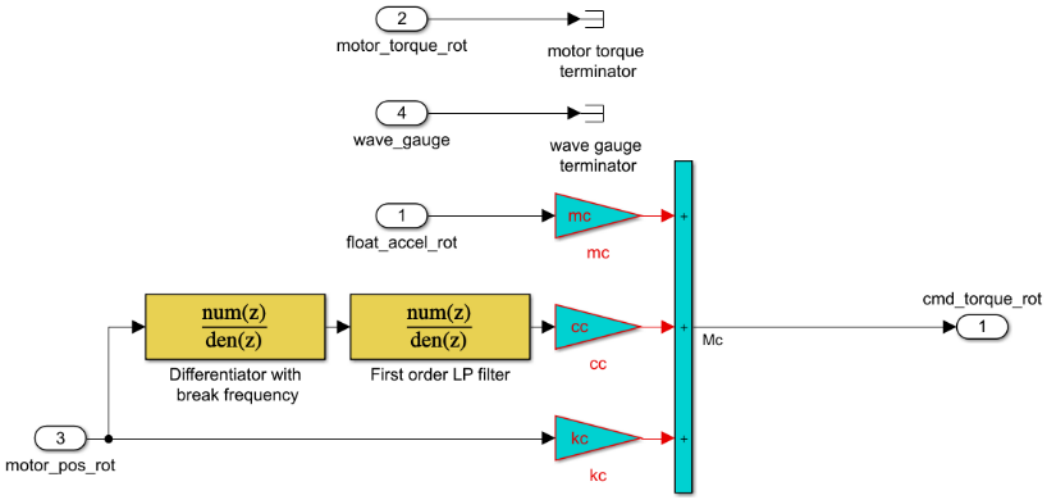

Figure 3: Implementation of a spring / damper control system

Running WECCCOMP sea states

WEC-Sim multiple condition runs were developed to run the WECCCOMP sea states, allowing contestants to perform batch runs by typing `*wecSimMCR*` into the MATLAB Command Window. The WEC-Sim multiple condition runs are defined in *WECCCOMP_ss.mat* and post-processing scripts are in *userDefinedFunctionsMCR.m*. Each of the WEC-Sim multiple condition runs has a ramp time of 5*Tp, and a simulation end of time 100*Tp. The first 25 seconds of WEC-Sim simulation will be discarded for calculation of the competition metrics (the evaluation criteria will only be calculated for time > 25s).

Calculating power and performance

For the WECCCOMP control competition, the controller's performance will be calculated according to the performance metrics defined in the **WECCCOMP info sheet**. The power performance will be calculated discarding the first 25 seconds of WEC-Sim simulation for calculation of the competition metrics. This has already been implemented in the post-processing script (*userDefinedFunctions.m*).

Upstream Wave Gauge Placement

By default, the wave surface elevation is calculated at the origin. Users are allowed to use up to 3 x-locations to calculate the wave surface elevation offset from the origin in the global x-direction by defining the waveClass variable, *waves.wavegauge<i>loc,* in the WEC-Sim input file:

waves.wavegauge1loc = -1.70 m
waves.wavegauge2loc = -1.50 m
waves.wavegauge3loc = -1.25 m

Resulting in the following wave elevation time series:

waves.waveAmpTime1 - incident wave elevation time series at *x = -1.70 m*
waves.waveAmpTime2 - incident wave elevation time series at *x = -1.50 m*
waves.waveAmpTime3 - incident wave elevation time series at *x = -1.25 m*

The placement of the wave gauges has been set at -1.25 m, -1.50 m, and -1.70 m upstream of the Wavestar float and have been written in the *wecSimInputFile.*m. The WEC-Sim numerical wave gauges output the undisturbed linear incident wave elevation at the wave gauge locations defined above. The numerical wave gauges do not handle the incident wave interaction with the radiated or diffracted waves that are generated because of the presence and motion of the Wavestar float. The location of the nearest wave gauge was chosen to reduce significant contamination of the incident wave elevation during the experimental stage. However, competitors may want to keep in mind how to handle this issue if the observed effect is greater than expected during experimental trials.