Title: A portable wave tank and wave energy converter for dissemination and outreach

Authors: *Nicholas Ross, Delaney Heileman, A. Gerrit Motes, Anwi Fomukong, Sean Pluemer, Francisco Colorbio, John Quinlan, Giorgio Bacelli, Steven J. Spencer, Dominic D. Forbush, Kevin Dullea, Ryan G. Coe*

Affiliations: Sandia National Laboratories, University of New Mexico

Contact email: rcoe@sandia.gov

Abstract: Wave energy converters are a nascent energy generation technology that harness the power in ocean waves. To assist in communicating both fundamental and complex concepts of wave energy, a small scale portable wave tank and wave energy converter have been developed. The system has been designed using commercial off-the-shelf components and the all design hardware and software are openly available for replication. Educational curriculum has also been developed to assist in using this hardware for classroom education.

Keywords: wave energy, educational, outreach

Specifications table:

Hardware name	Sandia Interactive Wave Energy Education Display (SIWEED)
Subject area	• *Educational Tools and Open Source Alternatives to Existing Infrastructure*
Hardware type	• *Electrical engineering and computer science* • *Mechanical engineering and materials science* • *Ocean engineering*
Open source license	GNU GENERAL PUBLIC LICENSE
Cost of hardware	*$7736 - Approximate cost of hardware (complete breakdown included in the Bill of Materials).*
Source file repository	https://github.com/SNL-WaterPower/siweed

1. Hardware in context

The Sandia Interactive Wave Energy Education Display(SIWEED) is a small scale wave tank that is designed to be portable and serve in outreach and dissemination of wave energy research. The development of this system was inspired by previous research at Sandia National Laboratories in the areas of wave energy converter (WEC) device and control design and testing. Specifically, the SIWEED demonstrates the causal feedback control and device design principles described in Bacelli and Coe [1] and Coe et al. [3].

A wide variety of similar educational wave tanks and tow tanks have been built, but few have been documented. Unger [8] reworked a tow tank at MIT to enable remote operation via the internet. Trust [7] created a tank with a similar scale($\sim$2 M) and objective, but oriented at demonstrating the effectiveness of coastal flood controls. There seem to be two major iterations, one electrically driven by a paddle like wave maker, and one plunger type driven mechanically by hand. That team has also made multiple similar hydraulic flume tanks for demonstration purposes. Ivan [5] has an educational wave tank on display in the National Museum of Scotland, but the waves do not explicitly interact with any bodies on the surface.

SIWEED is a novel expansion on similar small-scale educational wave tanks, as the wave maker is driven by a ball screw, and the user is able to control both the waves and the WEC device. The parameters of the wave are also much more controllable in this project than in similar small wave tanks, as users are able to select from various control modes, each with their own set of inputs:

Control mode	**Input parameters**
Jog	• Position (mm)
Function Mode	• Amplitude (mm) • Frequency (Hz)
Sea State	• Significant amplitude (mm) • Peak frequency (Hz) • Peakedness

The Sea State mode outputs a JONSWAP spectrum, a combination of many sine waves, meant to imitate a natural wave environment with random waves[1]. Initially, the primary intended audience for SIWEED was college students and above, but there now exists curriculum in the design files for lower age groups, and the GUI can be toggled to run a simplified version.

2. Hardware description

The SIWEED is composed of a 1.5 m × 0.3 m × 0.5 m acrylic tank, filled to roughly 0.3 m deep, housing a vertical plunger style wave maker (see, e.g., [4]), and a single body WEC modeled on the WaveBot [2]. Attached to the tank is a scale model of an ocean side town, meant to represent a potential power load. The town is equipped with four lighting zones, which allow the simulated power output to be physically represented. A photograph of the system is shown in Figure 3, while a CAD rendering is shown in Figure 1. The system is centrally controlled by a Windows PC running a graphical user interface (GUI) developed using Processing[2] and the controlP5 library[3]. Seperate hardware nodes are then manages by two Arduinos.

[1] Note that pseudo-random phasing is used.

[2] `https://processing.org`

[3] `http://www.sojamo.de/libraries/controlP5/`

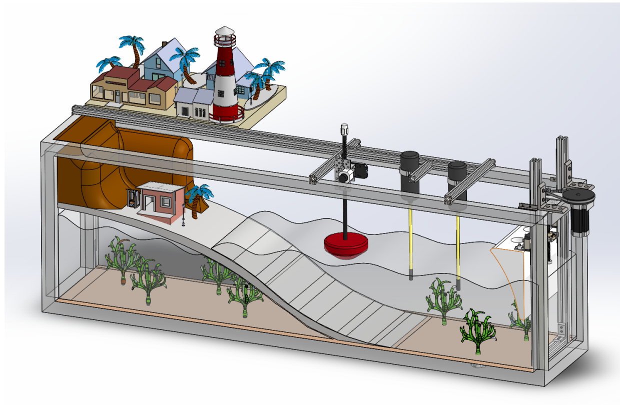

Figure 1: SIWEED CAD assembly.

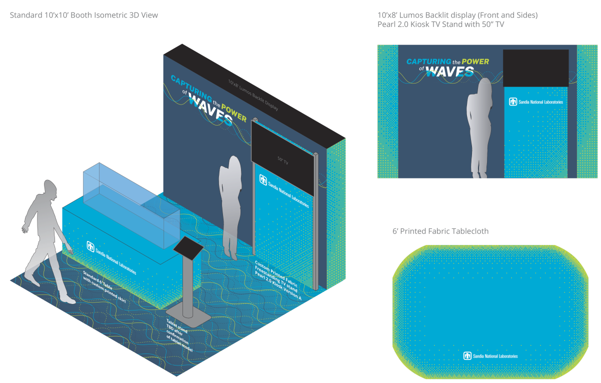

Figure 2: SIWEED presentation layout.

2.1 Hardware

A system diagram for the SIWEED is shown in Figure 4. Two Arduino Due micro controllers are used to control and acquire signals from the WEC and wave maker, communicating with the GUI through USB Serial. The Windows laptop acts as the core of the system, with the two Arduino Dues attached through USB. Each Arduino is then attached to the various devices it communicates with. It is worth noting that Dues are used for their higher clock speeds, but since they operate at 3.3 V, multiple level shifters are used to enable communication between the Arduino Dues and the components that use 5v logic(seeFigure 4).

The Windows laptop runs a Processing application that acts as the GUI, data logger, and serial communication manager. The Arduinos receive their commands from Processing over usb serial, and perform individual control loops based on control states dictated by the GUI. One Due controls the movement of the wave maker plunger, while the other controls the torque feedback of the WEC and lights in the model town. Each send data back to the Processing GUI for data logging and plotting purposes.

The Arduino Dues each have a number of components they communicate with through various protocols (PWM, SPI, Analog). These perform tasks like encoder buffering, motor control, and signal generation. The connections can be seen in Figure 4.

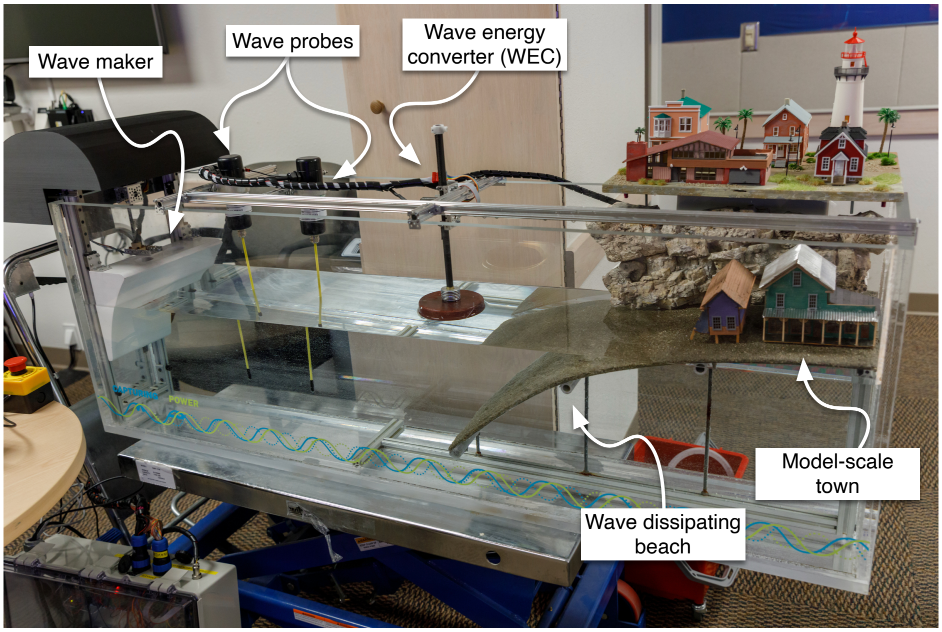

Figure 3: Photograph of SIWEED system showing key system elements.

2.2 Graphical User Interface

This GUI is intended to be used with a touchscreen interface, but it is also fully functional with a mouse. A screen shot of the GUI is shown in Figure 5. It is divided into three sections: The left instructional side, "Mission Control", and "System Status".

The left side of the "Mission Control" section of the GUI pertains to control of the wave maker, which allows the user to switch between operational modes ("jog," "function," "sea state," and "off") and set the relevant parameters. If, for example, the system is in "sea state" mode, the user can set the significant wave height (H_s), peak frequency (f_p), and peakedness factor (γ) for a JONSWAP wave spectrum. The "function" mode commands a simple sine wave, and so the parameters are amplitude and frequency. The "jog" mode simply commands a static position for the plunger to go to, so the only input parameter is that command position. There are only as many input sliders displayed as there are inputs to the active control mode. For example, if the selected mode was "jog", only one slider would be present in that section of the GUI for the user to interact with.

The right side of the "Mission Control" section of the GUI contains controls for the WEC, which can be set into "torque" mode; in which the user directly sets a commanded torque, "feedback" mode; where proportional and integral feedback factors can be set by the user; "SID" mode, which represents system identification and allows for open-loop multi-sine signals to be set by the user; and "off." The "SID" works with the same inputs and JONSWAP control method as the "Sea State" mode for the wave maker, but commanding torque instead of position. Where the purpose of the JONSWAP in the wave maker controls is to replicate a natural wave state, as a torque controller for the WEC its purpose is only to provide a wide variety of torque inputs. This is useful when characterizing the hardware using recorded data in post-analysis.

The "System Status" section shows two plots, one for the WEC and one for the wave maker, with each allowing the user to choose what data is plotted. There are multiple options for the data included in each of these plots, with the ability to plot any combination of variables simul-

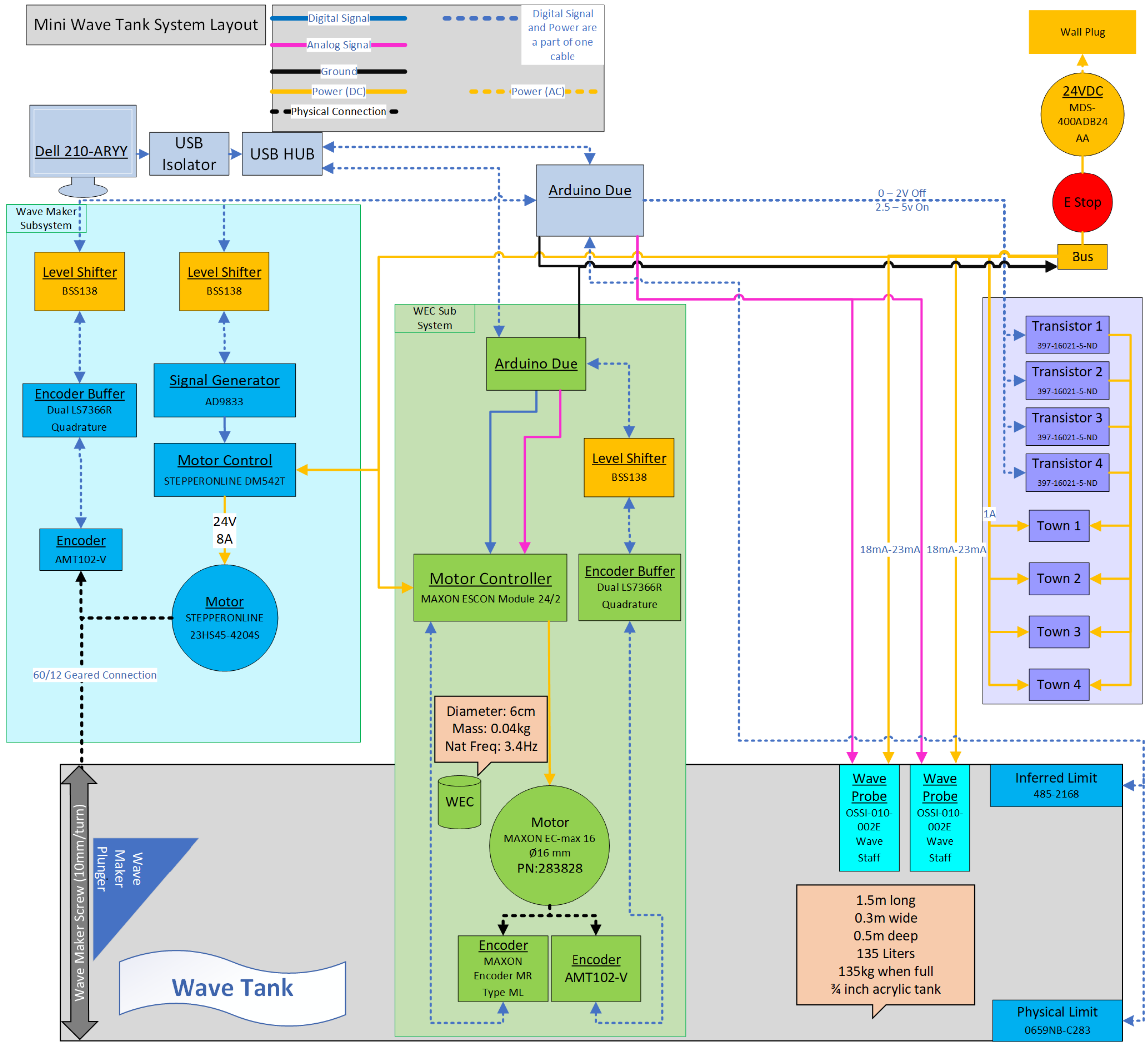

Figure 4: System layout diagram.

taneously. Each data stream is plotted in a different color, allowing the user to differentiate the various variables on each plot. The available variables in the wave maker plot (left side) are "Wave Maker Position"(as measured by the encoder) and "Wave Elevation"(As measured from one of the submerged probes). The WEC plot variables are "Position"(measured from an encoder), "Velocity"(differentiated from position), "Torque"(derived from a torque constant and motor current), and "Power"(as derived below). Below the plots are a discrete Fourier transform of the generated waves (measured from the encoder), and a radial meter that displays the hypothetical generated power of the WEC. That hypothetically generated power meter is driven by the same variable as the "Power" variable in the WEC information plot above the meter, given by the following formula:

$$P = -\tau \cdot v \tag{1}$$

Where P is the calculated output power, τ is the torque commanded by the WEC motor controller, and v is the velocity.

A number of key factors of SIWEED are:

- **Open-source and documented design:** To our knowledge, the SIWEED is the first edu-

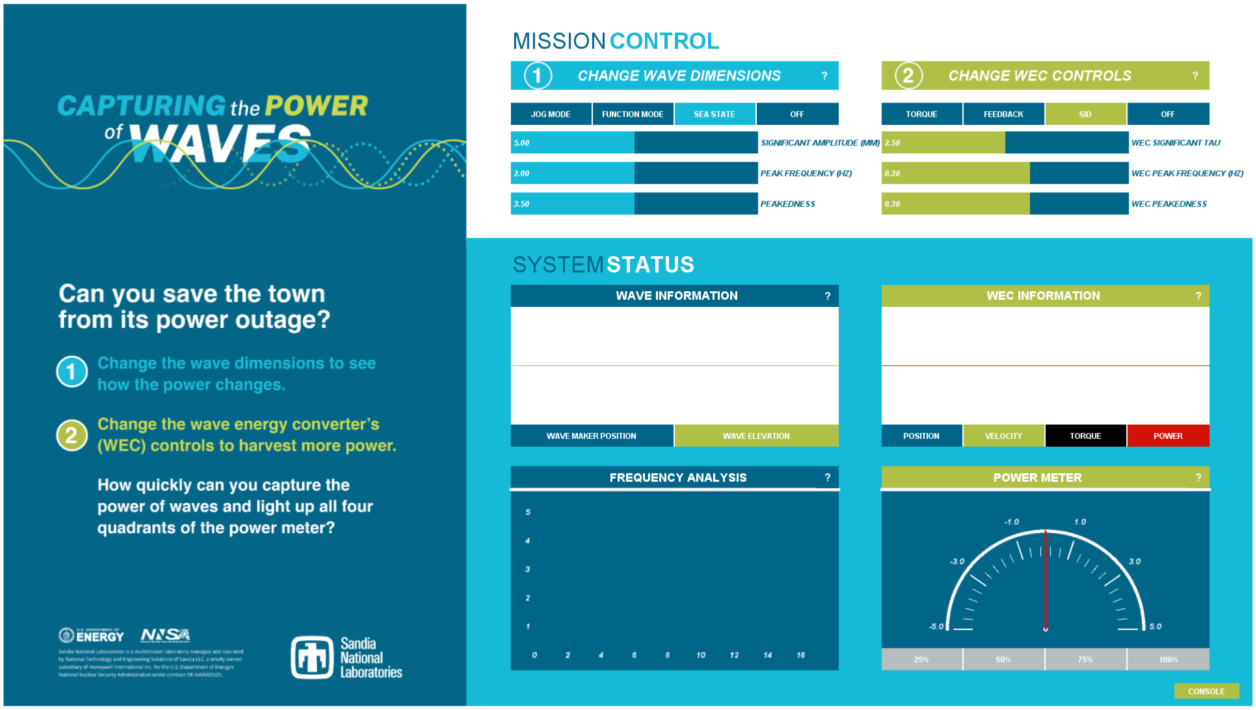

Figure 5: Graphical user interface (GUI) screenshot.

cational wave tank and WEC system to be fully documented with an open-source design and software package. This will enabled interested researchers and students to efficiently develop replicates and variations of the SIWEED design for their own purposes.

- **Portability:** To allow for transport to conferences and outreach events, the SIWEED has been specifically designed to allow for easy transportation.
- **WEC control:** In addition to demonstrating the physics of ocean waves, the SIWEED allows users to interact with a WEC control system to better understand the principles of reactive feedback control to maximize power absorption for the waves.
- **Student-led design:** The SIWEED project has been a student-led design project involving undergraduate engineering students.
- **Curriculum:** The SIWEED repository includes a curriculum for students K-12 based on the Next Generation Science Standards [6]. The repository contains a presentation and a document intended to be given to the teacher beforehand. The presentation explains water power and Sandia National Labs' role within it, and then goes into more detail about wave energy converters and how the SIWEED demonstration works. The teacher resources document contains general information about the project, and a multitude of resources surrounding wave energy education, including videos, worksheets, vocabulary words, general topics, and Next Generation Science standards for each grade level.

3. Design files

Design file-name	File type	Open source license	Location of the file(s)
Final SIWEED Town	.SLDASM	GNU GENERAL PUBLIC LICENSE	*online repository*
Hardware Hat	.SLDPRT	GNU GENERAL PUBLIC LICENSE	*online repository*
WEC_Controller/ interrupts.ino	.ino	GNU GENERAL PUBLIC LICENSE	*online repository*
WEC_Controller/ Serial.ino	.ino	GNU GENERAL PUBLIC LICENSE	*online repository*
WEC_Controller/ WEC_Controller.ino	.ino	GNU GENERAL PUBLIC LICENSE	*online repository*
WaveMaker_Controller/ interrupts.ino	.ino	GNU GENERAL PUBLIC LICENSE	*online repository*
WaveMaker_Controller/ Serial.ino	.ino	GNU GENERAL PUBLIC LICENSE	*online repository*
WaveMaker_Controller/ WaveMaker_Controller.ino	.ino	GNU GENERAL PUBLIC LICENSE	*online repository*
dataLogging.pde	.pde	GNU GENERAL PUBLIC LICENSE	*online repository*
Meter.pde	.pde	GNU GENERAL PUBLIC LICENSE	*online repository*
Serial.pde	.pde	GNU GENERAL PUBLIC LICENSE	*online repository*
siweedGUI.pde	.pde	GNU GENERAL PUBLIC LICENSE	*online repository*
UI.pde	.pde	GNU GENERAL PUBLIC LICENSE	*online repository*
UnitTests.pde	.pde	GNU GENERAL PUBLIC LICENSE	*online repository*
fft.java	.java	GNU GENERAL PUBLIC LICENSE	*online repository*
Complex.java	.java	GNU GENERAL PUBLIC LICENSE	*online repository*

The referenced online repository is here: https://zenodo.org/record/7502670.
CAD Model: The CAD assembly includes all functional physical parts of the SIWEED system, with accurate dimensions that can be referenced during fabrication and assembly. Aesthetic details, like the town are intentionally vague in the CAD model. Physical implementation of artistic aspects are not important for functionality, but are thought to add significantly to audience engagement. The files in the repository are Solidworks models, with all dependencies included in a compressed

archive. There is an additional part labeled "Hardware Hat". This was a late addition that was 3D printed to cover potential pinch points on the model, and prevent sunlight from interfering with the infrared limit switches. Because it's a late addition, the part is included in the repository, but not in the final assembly.

Processing Sketches: The Processing sketch is made up of multiple files combined in the processing IDE as separate tabs. This includes the `.pde` files and the `.java` files. This is done mostly for organization purposes, but all files are necessary for the sketch to run properly. It is important to note that the sketch is also dependent on the Console library and ControlP5 library, and various graphics and text files that support the GUI. These files are not listed here but are all included in the online repository.

Arduino Sketches: Similarly to the Processing sketch, the two Arduino sketches are broken into multiple files for organization. Once again, all files are required for the sketch to function. It is important to note that the sketches are also dependent on various imported libraries, which are included in the online repository, but not listed here. Some of these libraries are modified from their original versions, so it is important that they are pulled from this project repository, and not the libraries' original sources.

4. Bill of materials

The complete Bill of Materials can be found here: https://zenodo.org/record/7502670. Note that a SIWEED system can be assembled with any tank on hand (e.g., a fish tank) and any Windows PC capable of running the Processing IDE; as these components can contribute significantly to system cost, the BOM lists the tank and computer separately from the remainder of the components.

5. Build instructions

A functionally similar version of this wavetank can be reconstructed using the System Layout diagram in Figure 4, the bill of materials, and the CAD model shown in Figure 1, which is also included in the design files. Many parts can be substituted with similar ones, such as the laptop, transistors, power supply, or the tank itself. Software specific hardware, such as the Arduinos or Encoder Buffers should not be substituted. It is important to consider safety during reconstruction: Electrical isolation and grounding, as well as guarding pinch points, is essential for safe usage.

Software setup:

- Download the latest version of Arduino IDE and Processing IDE.
- Download the Master SIWEED Repository.
- Change the Sketchbook Location for the Arduino IDE to `\siweed\Arduino`.
- Change Sketchbook Location for the Processing IDE to `\siweed\Processing`.
- Change Arduino IDE board to "Arduino Due Programming Port".
- Upload the sketches to the Arduinos.
- A more detailed version of these instructions can be found in the README file of the repository.

6. Operation instructions

After performing the setup described in the build instructions, the project can be operated as follows:

- Power on the wavetank, ensuring any kill switches are closed, and any pinch points are clear.
- Open processing.exe.
- File > Open > siweedGUI.pde
- (optional)Edit modifiers. These are booleans at the top of the code that will do things like enable basic mode, debugging, or data logging.
- Click "Run".
- The GUI will open, allow some time for it to load.
- In the console, which will either be in the GUI, made visible by pressing the console button in the bottom right corner, or in the Processing console, depending on your boolean modifiers, unit tests should be visible that can be used to verify that everything is functioning properly. Note that the color of the console button in the GUI is representative of the serial checksum status. When the received checksum matches the calculated checksum(functioning nominally), the button will be a green/yellow. Otherwise it will be gray, indicating one of the Arduinos' checksum does not match, and the system is out of sync. This will result in the state of the GUI not accurately representing the state of the physical system.

7. Verification and characterization

A series of analyses were conducted on the hardware and software to confirm the proper functionality of the SIWEED. These tests ensure that individual components, and the systems they rely on, behave as expected. What is not included here is a series of software unit tests that run at the start of every execution of the GUI. These test specific software functions as well as the integrity of some of the hardware and the serial communications. Results of these tests can be found on a per execution basis in the GUI console. During nominal operation all unit tests have proven to pass.

7.1 Torque constant verification:

The factory torque constant was verified by logging the position and current of the free floating WEC while commanding a slow sinusoidal current. The position displacement time series was converted to a time series of the applied physical torque based on the hydrostatic force due to submerging/lifting the WEC:

$$\tau = \sigma \cdot \gamma \cdot d \cdot \pi \cdot r^2 \tag{2}$$

Where τ is torque, σ is the radius of the pinion gear, γ is the specific weight of water, d is the displacement of the buoy in the water, and r is the radius of the buoy. It is important to note that the torque values at this point did not account for friction or hysteresis. Plotting this torque data against the recorded commanded current values gives Figure 6. The "fit1" line represents the best linear fit of the entire data set, but the data points at the highest and lowest currents are affected more significantly by static friction, so omitting them from the data set improves the torque

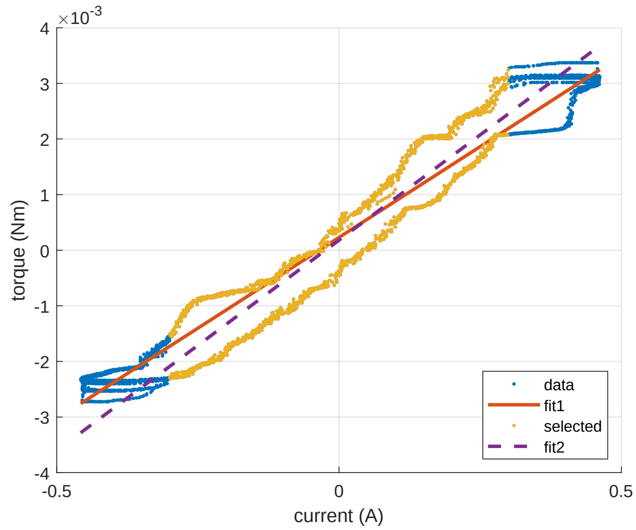

Figure 6: Torque vs commanded current, with linear fits of the entire and partial data set.

constant estimation, as shown with the "fit2" line in Figure 6. This provides an estimated torque constant of about 7.52 mNm/A, which is within 4% of the factory stated constant of 7.8 mNm/A.

Since torque is being estimated from displacement and the frequency is low (inertial and viscous effects are small), the friction shows up as a "constant" offset from the ideal torque constant and doesn't affect the slope. This allows the friction to be seen as half the offset between the data points as the wave maker moves up, and the points as it moves down. Plotting the torque error using the calculated torque constant against the current gives Figure 7. The average offset shows that the friction is about 0.45 mNm in this operational envelope.

7.2 Wave Probes:

The readings from the wave probes could be verified by attaching them to the wave maker, which has a very precise encoder that allows for accurate measurement of changes in displacement. Moving the wave maker with the wave probe attached and comparing the displacement data from both of them was used to make sure the wave probe was providing accurate readings.

It was found having two wave probes as close at they are, roughly 1/,ft apart, was problematic. Changing the depth of one probe seemed to also adjust the reading of the nearby, stationary probe. It is believed this is caused by a sort of capacitive coupling, but a solution was never pursued. Known work-arounds are only using one probe, or powering the second probe off during operation.

7.3 WEC feedback control

The proportional derivative (PD) controller functions on the following formula:

$$\tau = k_p \cdot p + -k_d \cdot v \tag{3}$$

Here, τ is the calculated torque, k_p is the user defined proportional gain, p is the position in meters, k_d is the user defined derivative gain, and v is the velocity, in meters per second, calculated by the change in position over the last 10 milliseconds. Using this control method allows the WEC to be controlled as a spring and damper, with the proportional gain k_p adjusting the strength of the spring, and the derivative gain k_d adjusting the strength of the damper. Unlike a physical spring,

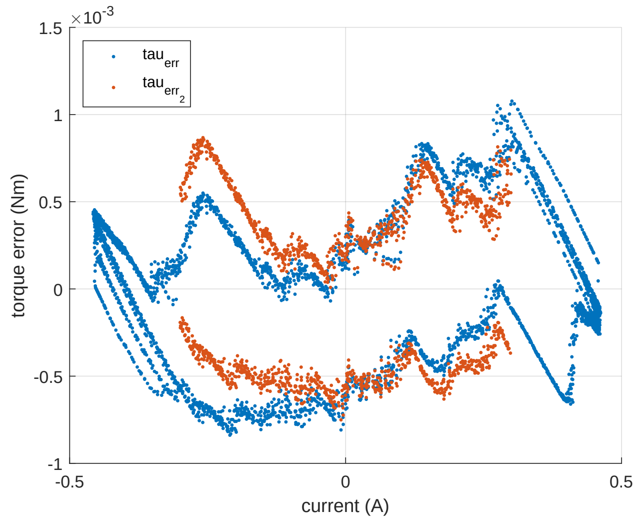

Figure 7: Torque error vs commanded current.

this "spring" can both push device towards center, or push it away, depending on the sign of the gain. A positive k_p will act as a destabilizing force, whereas a negative k_p will center the WEC to the waterline, similarly to a physical spring.

This calculation is done on the Arduino Due micro controller responsible for controlling the WEC, which commands an output torque through PWM to the WEC motor controller. The position and velocity are determined with an encoder, while the k_p and k_d inputs are set by the user in the GUI, which the Arduino receives through USB serial.

By logging the inputs and output of the control system through a range of user inputs, its performance can be verified. Initial tests showed a delay of three samples (100 ms) in the control loop. This was found to be an error with the serial communication checksum function, and has since been fixed. Control loop response times can now be verified as less than 33 ms, as the data logging control loop runs at 30 Hz and shows a delay of one sample or less.

Figure 8 shows the controller during typical operation. The commanded torque was calculated as per Equation 3, while the measured torque was taken from an analog output on the WEC motor controller that measures current, then multiplying it by the known torque constant(see Section 7.1). This shows us how the PD control algorithm and motor controller react to our given inputs. As the linearity of the plot shows, the measured output torque closely matches the expected torque. The standard deviation of the torque during user testing showed to by 3.82e−4 Nm.

There is a theoretical possibility to experience saturation near the limits of our torque output. This is the result of a protection written into the WEC motor controller, and can be adjusted by what is referred to as the "Thermal Time Constant". When commanded to apply a torque over what the controller deems as safe, the motor controller will apply this torque for only some time before saturating at the maximum nominal torque. This was observed during lab limit testing, occurring around 3.5e−3 Nm, but not during any user tests, as the commanded torque was never that high. If it were to become and issue, the the range of possible k_p and k_d commands and torque jog commands could be made more limiting in the GUI to avoid any saturation triggers.

While fitting closely, there is still error present in this control loop. Figure 9 shows a histogram of the torque error, calculated by subtracting the measured torque from the expected torque. In these particular user tests, the RMS error was 9.84e−5 Nm. For context, the standard deviation of the torque measurement was 3.82e−4 Nm.

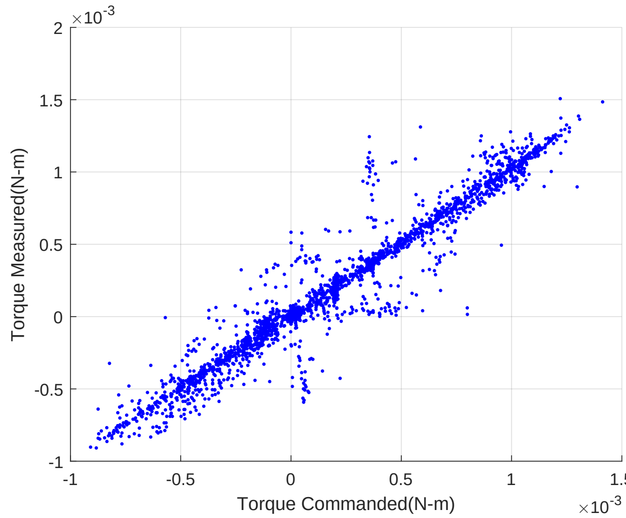

Figure 8: Torque measured vs torque commanded.

Part of what has to be considered with this error is noise in the analog signal that is used to read amperage and calculate measured torque. To characterize this, zero torque can be commanded, and then the measurement analyzed. For simplified data collection, user testing data sets were truncated to only sections where the commanded torque was zero. Figure 10 shows a histogram of the torque when a zero command is sent. It can be easily seen that there is a slight positive offset in the noise, which is assumed to come from inaccuracy in the analog reporting on the motor controller, or the analog reading on the Arduino Due. This offset varies from test to test, and can be negative. Using this method of analyzing only data points where the torque command is zero over multiple tests gives a noise band of about $8e{-5}$ Nm, where 90% of sampes lie between -$4e{-5}$ Nm and $4e{-5}$ Nm.

8. Declaration of interest

Declarations of interest: none

9. Conclusion

The SIWEED was developed to communicate the importance of controls within wave energy devices while adhering to the open source philosophy. This paper has provided the resources necessary to understand and even reproduce the SIWEED, while demonstrating the viability of the system. With this information in hand, the owners and operators of SIWEED are now empowered to educate others about wave energy, and any interested parties are enabled to recreate this project.

Should others wish to improve the SIWEED design, the wave probes module would act as an excellent start. Because of what was believed to be capacitive coupling as described in Section 7.2, and limited sensor resolution, the produced waves were never fully characterized to the user. Communication of wave speed and a more precise wave height measurement in the GUI could prove beneficial to the user.

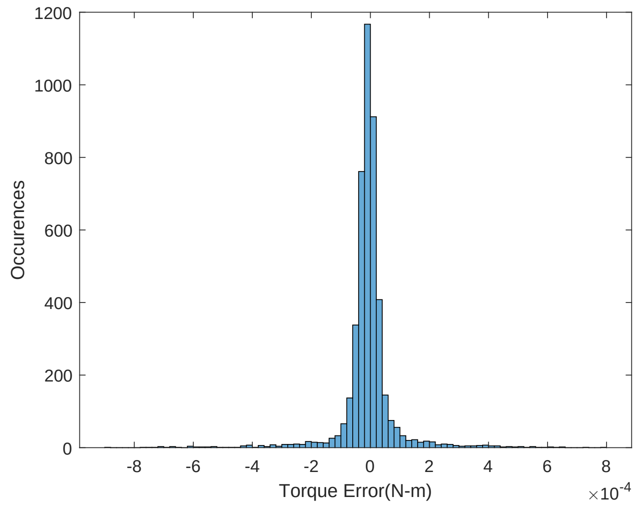

Figure 9: Error histogram.

Acknowledgment

Sandia National Laboratories is a multi-mission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525. This paper describes objective technical results and analysis. Any subjective views or opinions that might be expressed in the paper do not necessarily represent the views of the U.S. Department of Energy or the United States Government.

References

[1] Giorgio Bacelli and Ryan G. Coe. Comments on control of wave energy converters. *IEEE Transactions on Control Systems Technology*, 29(1):478–481, Jan 2021. doi: 10.1109/TCST.2020.2965916. URL `https://ieeexplore.ieee.org/document/9005201`.

[2] Ryan G. Coe, Giorgio Bacelli, David Patterson, and David G. Wilson. Advanced WEC Dynamics & Controls FY16 testing report. Technical Report SAND2016-10094, Sandia National Labs, Albuquerque, NM, October 2016. URL `https://mhkdr.openei.org/submissions/151`.

[3] Ryan G. Coe, Giorgio Bacelli, and Dominic Forbush. A practical approach to wave energy modeling and control. *Renewable and Sustainable Energy Reviews*, 142:110791, 2021. ISSN 1364-0321. doi: 10.1016/j.rser.2021.110791. URL `https://www.sciencedirect.com/science/article/pii/S1364032121000861`.

[4] Jae Min Hyun. Simplified analysis of a plunger-type wavemaker performance. *Journal of Hydronautics*, 10(3):89–94, 1976.

[5] Ivan. New educational tank in the national museum of scotland's energy laboratory. URL `http://www4.edesign.co.uk/2016/07/new-tank-at-the-national-museum-of-scotlands-new-energy-laboratory/`.

[6] NextGenScience. Next generation science standards. URL `https://www.nextgenscience.org/`.

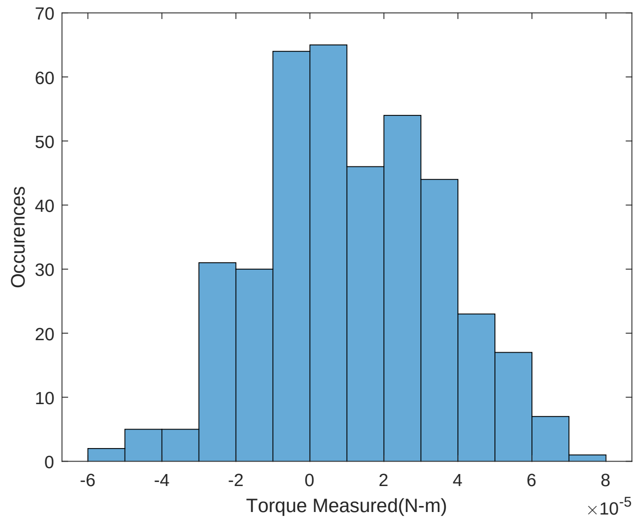

Figure 10: Noise histogram.

[7] JBA Trust. Wave tank. URL `https://www.jbatrust.org/how-we-help/physical-models/wave-tank/`.

[8] Matthew Lawrence Unger. *Creating a flexible, Web-enabled learning and research facility at the MIT Towing Tank*. PhD thesis, Massachusetts Institute of Technology, 2006.