Oregon State University

Model Predictive Control for Robots in Ocean Waves

As presented by:
Daniel C. Fernández
M.S. Candidate
Robotics

About Me

Oregon State UNIVERSITY

- 1st year in Coastal/Ocean Engineering
- Summer 2014: Hinsdale Wave Research Lab
- Supported CEOAS and OOI glider groups
- 2nd year with Robotic Decision Making Lab

OSU Oregon State UNIVERSITY College of Engineering

CEOAS College of Earth, Ocean, and Atmospheric Sciences

OOI OCEAN OBSERVATORIES INITIATIVE

O.H. HINSDALE WAVE RESEARCH LABORATORY OREGON STATE UNIVERSITY

Motivation

- Ocean waves will displace a robot
- Wave disturbances lend to increased sensor drift

Source: *National Geographic*, 2012, R. D. Ballard

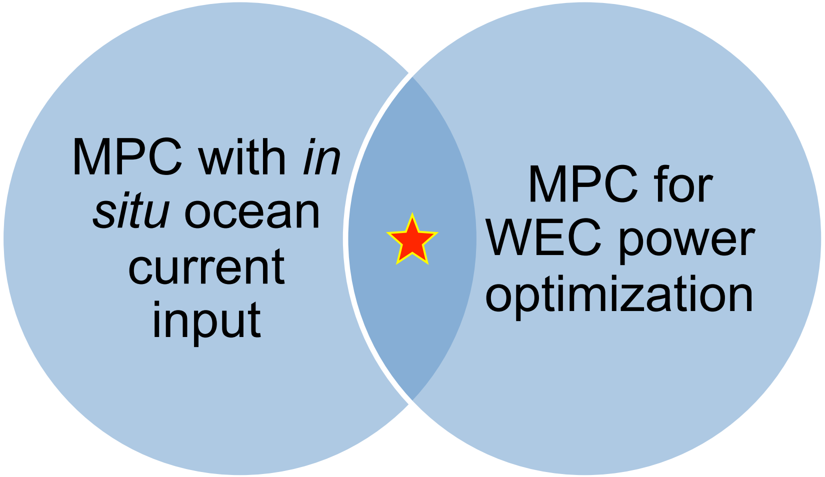
- Sensor drift reduces robotic observation quality
- Impending wave forces can be estimated
- **Objective: keep a station-keeping robot stationary under the influence of a wave field**

Motivation

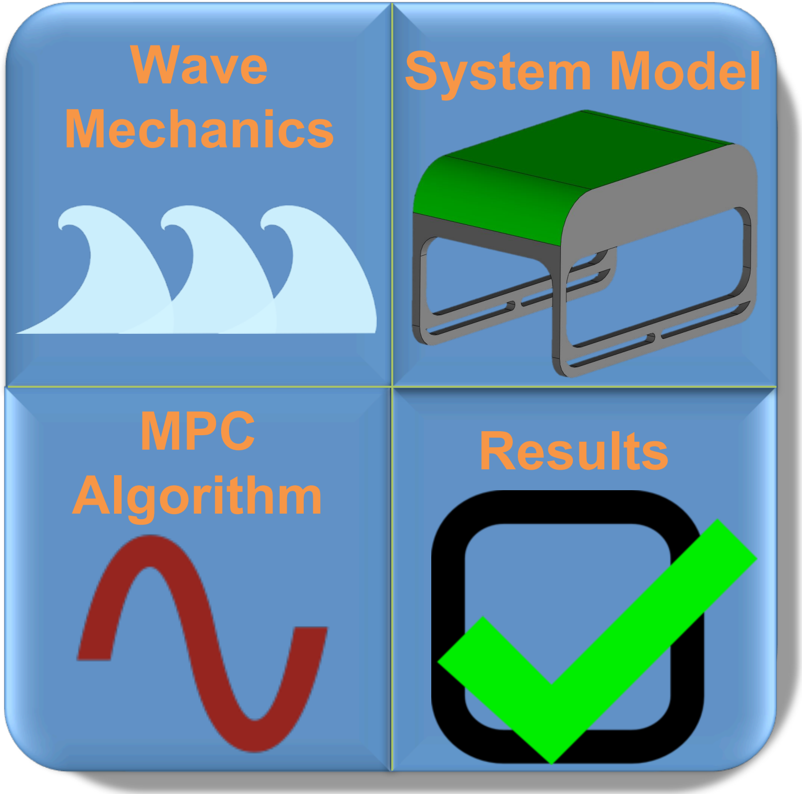
Related Work

Oregon State UNIVERSITY

- Model Predictive Control (MPC)
  - Path planning with *in situ* ocean currents (Medagoda, 2012)
  - Wave Energy Converter (WEC) optimization (Brekken, 2011)
  - Station-keeping under water waves (Heidel, 1998)

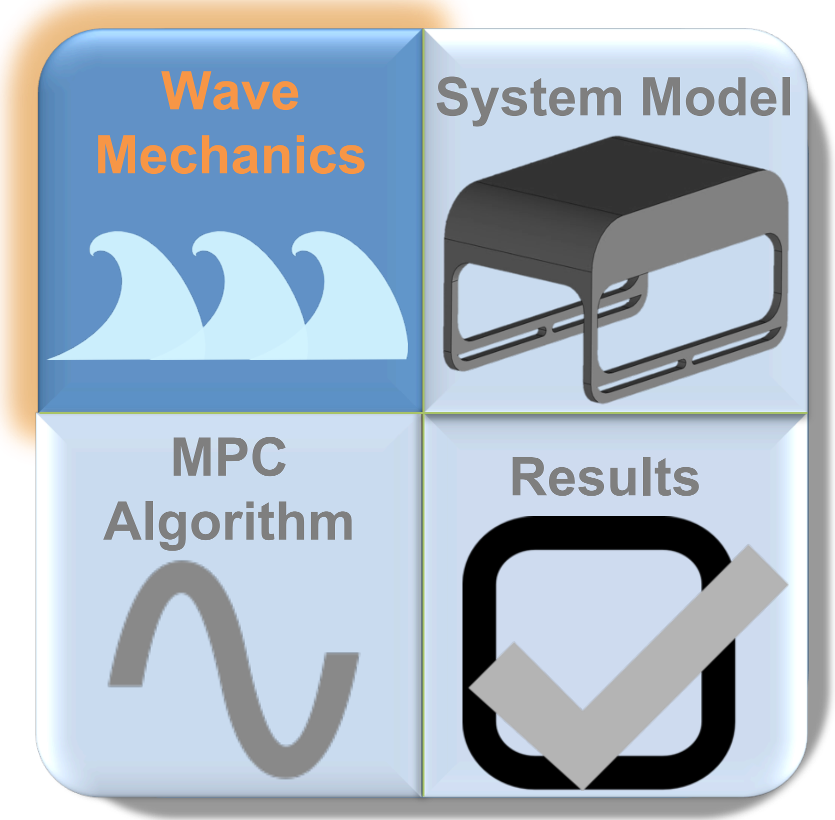


Outline

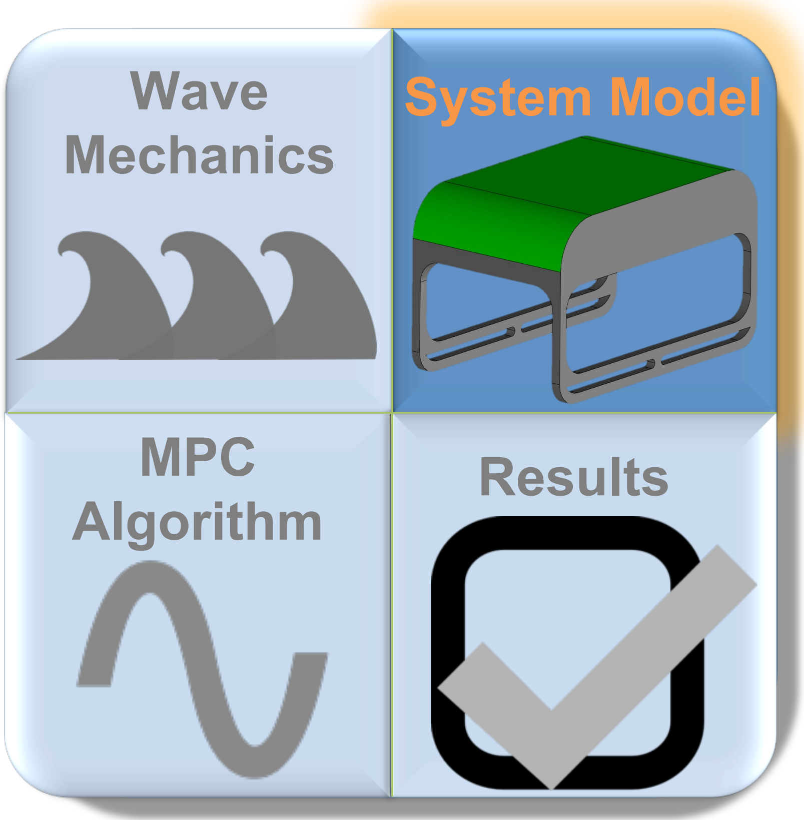


Outline

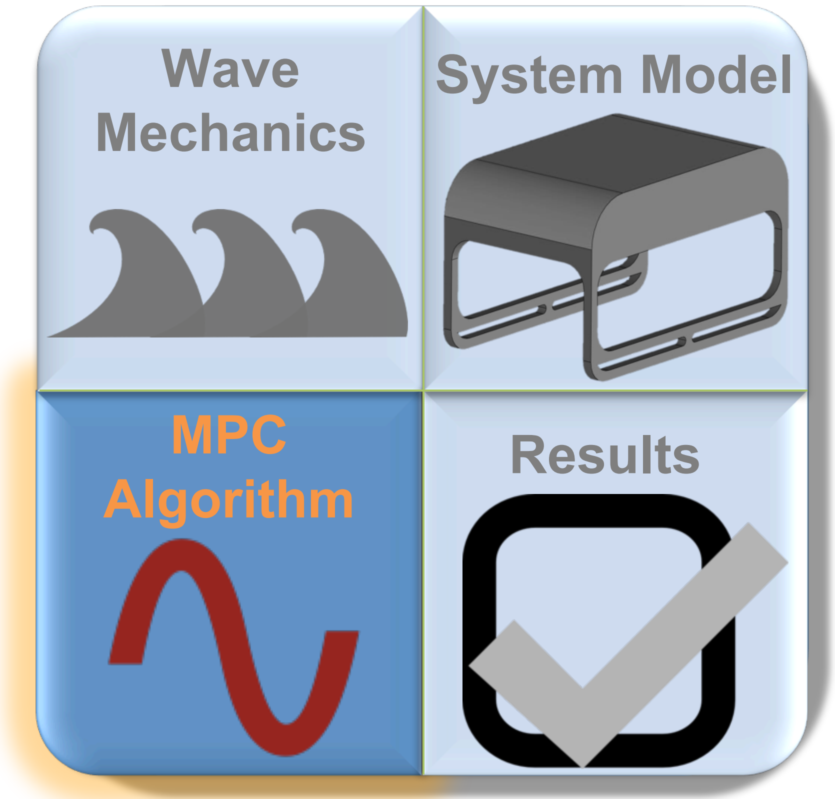
Oregon State University



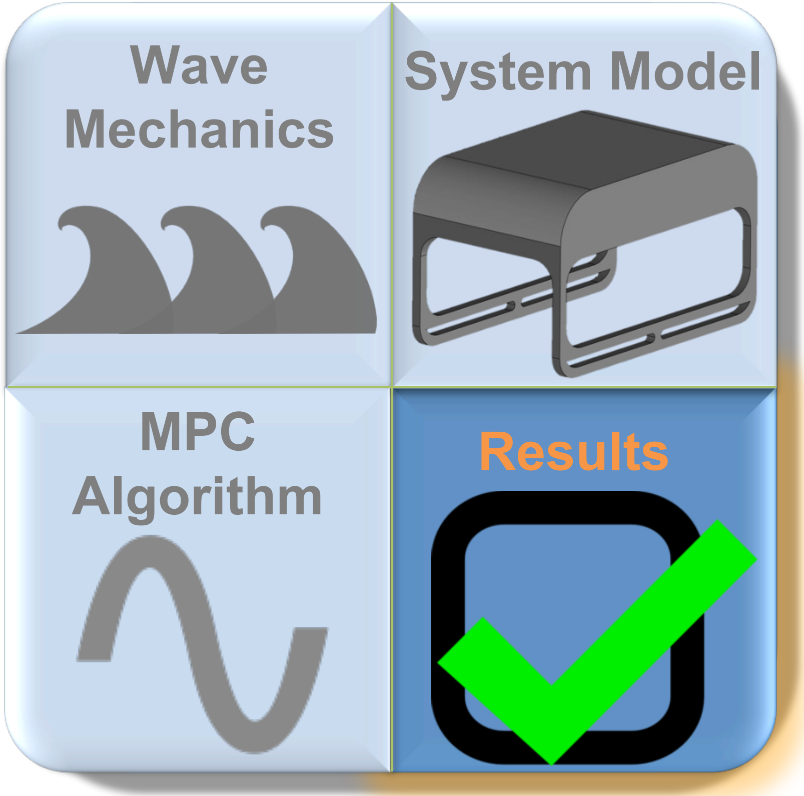
Outline



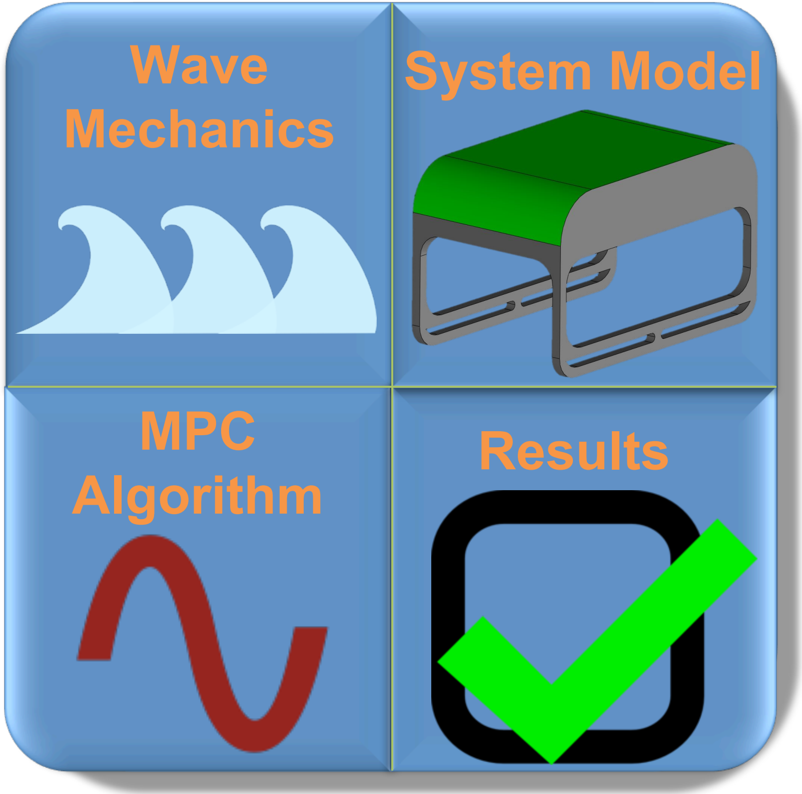
Outline



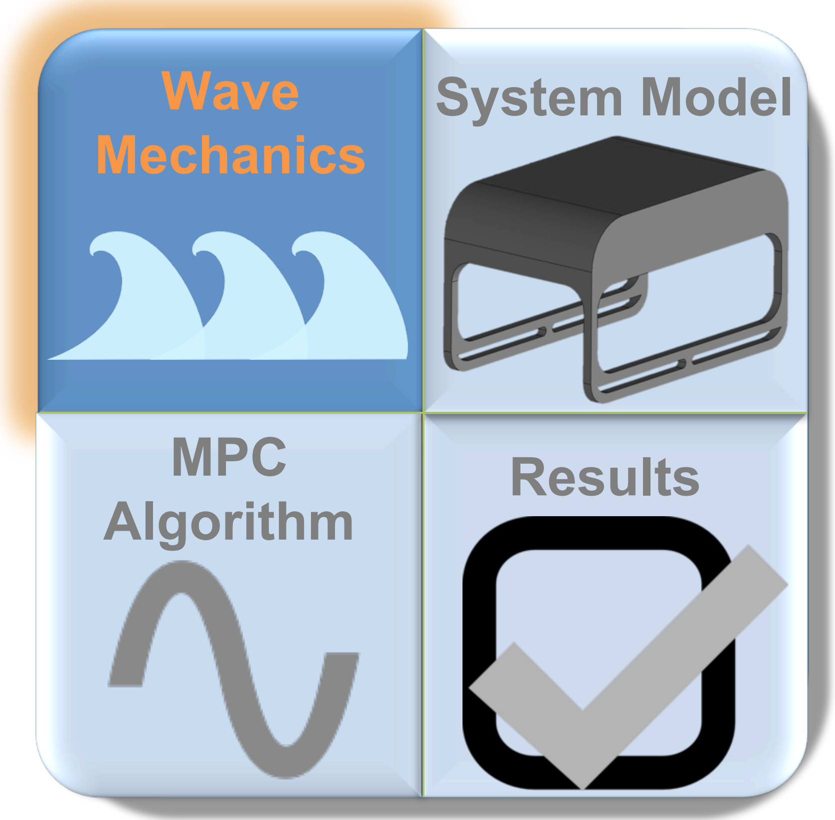
Outline



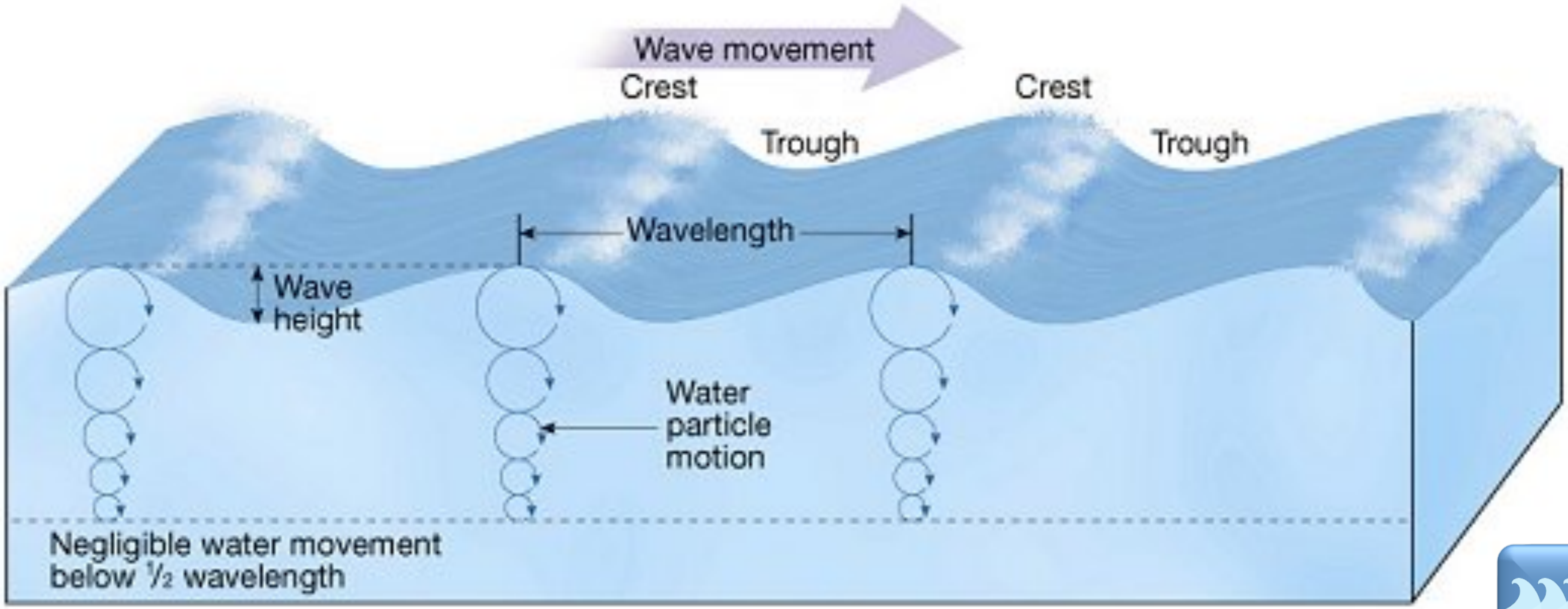
Outline



Outline

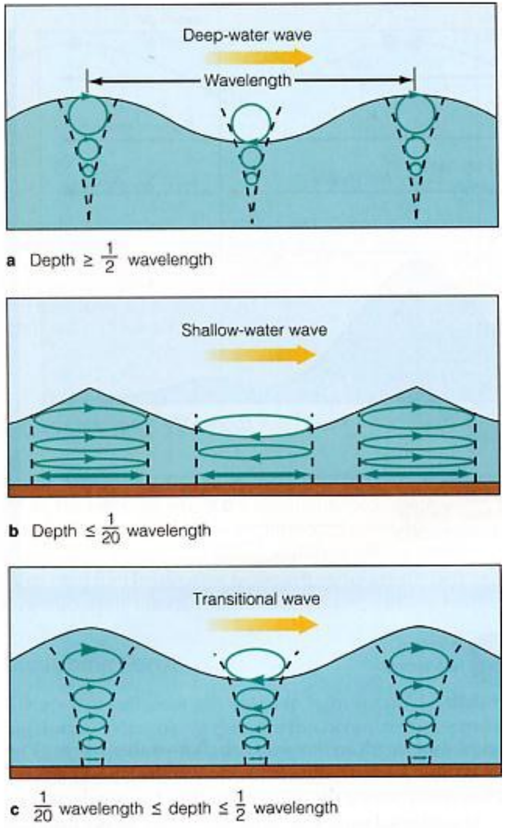


Parts of a Wave



Source: *Waves, Tides and Shallow-Water Processes*, 1999, J. Wright, A. Colling, D. Park

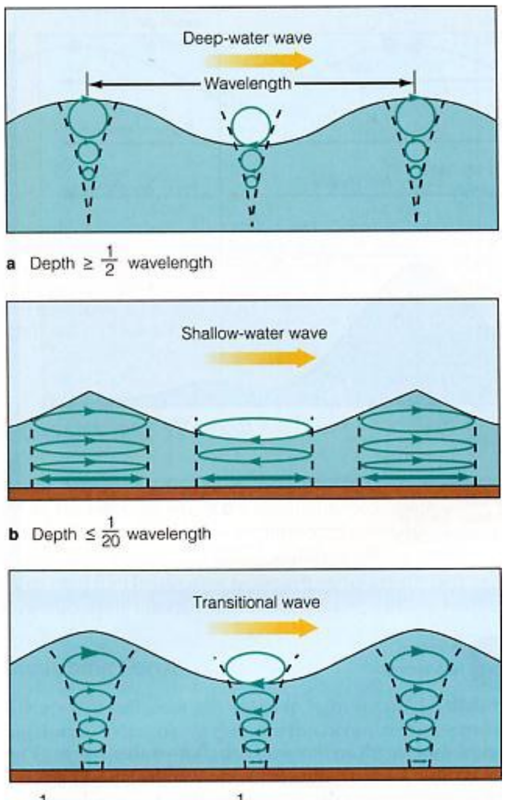
Water Motion under Waves



- Deep water wave
  - circular paths, exponential decrease with depth
  - $v_p = 1/2$ at $z = -L/9$, close to 0 at $z = -L/2$
- Shallow water wave
  - More lateral motion than vertical
  - More elliptical with depth, virtually lateral at depth
- Transitional water depths
  - Intermediate elliptical patterns
  - Majority of waves in this work

Source: *Waves, Tides and Shallow-Water Processes*, 1999, J. Wright, A. Colling, D. Park

Water Motion under Waves



a Depth $\geq \frac{1}{2}$ wavelength

b Depth $\leq \frac{1}{20}$ wavelength

c $\frac{1}{20}$ wavelength $\leq$ depth $\leq \frac{1}{2}$ wavelength

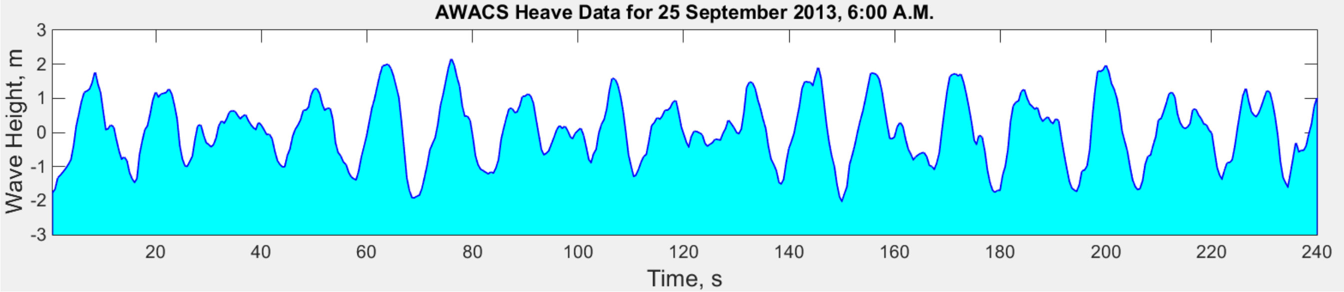
- Deep water wave
  - circular paths, exponential decrease with depth
  - $v_p = 1/2$ at $z = -L/9$, close to 0 at $z = -L/2$
- Shallow water wave
  - More lateral motion than vertical
  - More elliptical with depth, virtually lateral at depth
- Transitional water depths
  - Intermediate elliptical patterns
  - Majority of waves in this work

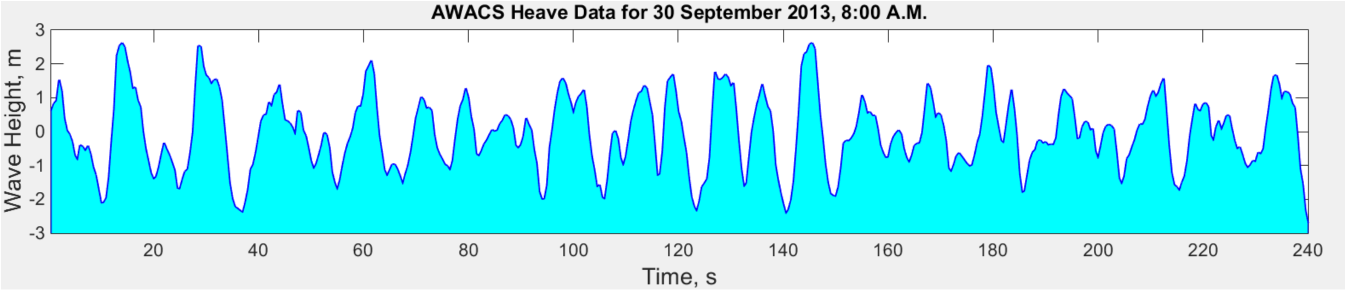
Source: *Waves, Tides and Shallow-Water Processes*, 1999, J. Wright, A. Colling, D. Park

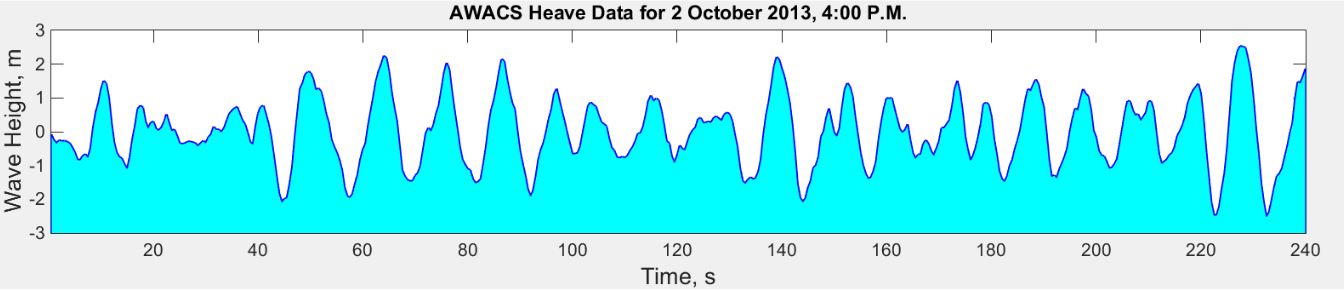
Simulation Setting (NETS)

- North Energy Test Site
- Operational depth: 50 m
- Validation of wave field
  - AWAC acoustic measurements
  - Deployed at NETS
  - August – October 2013
  - (600) 40 minute profiles (2 Hz)

Simulation Setting (NETS)

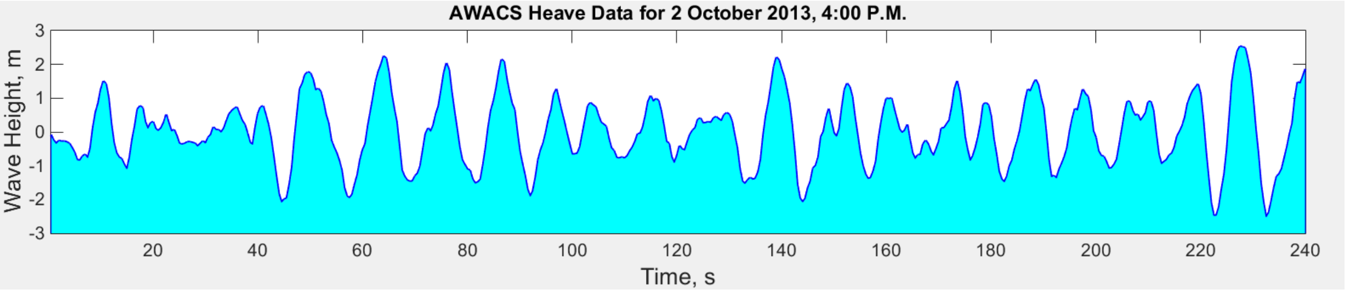




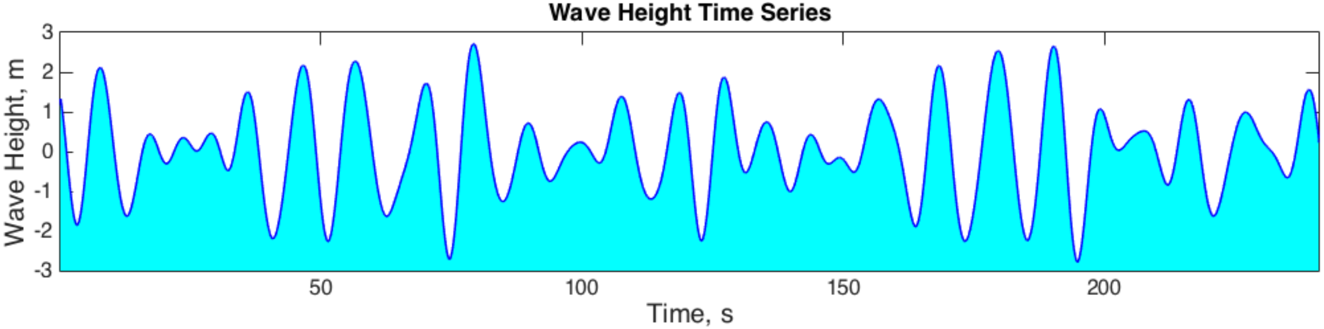


Simulation Setting (NETS)

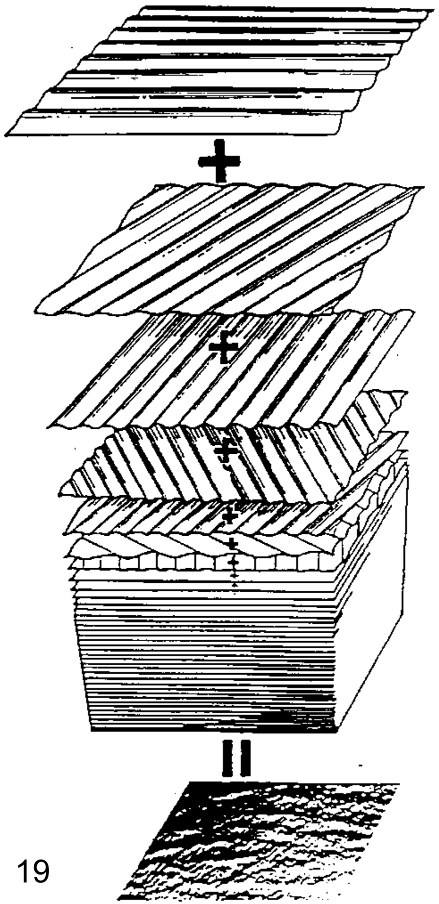
Real Data



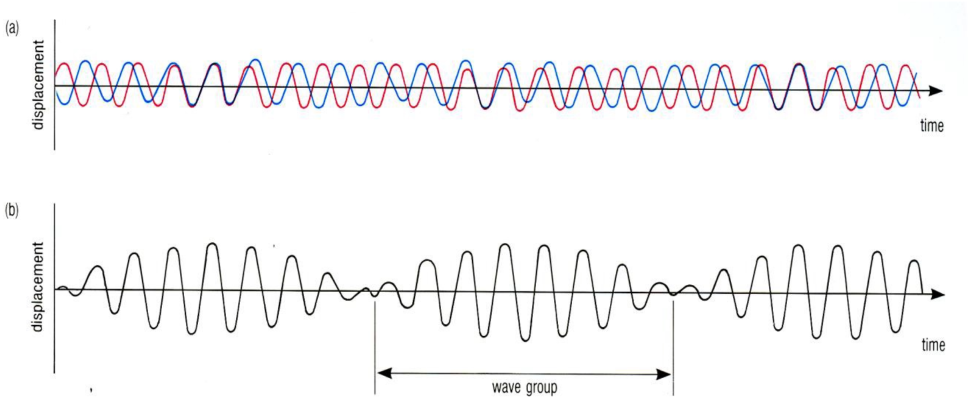
Simulated Data



Random Seas



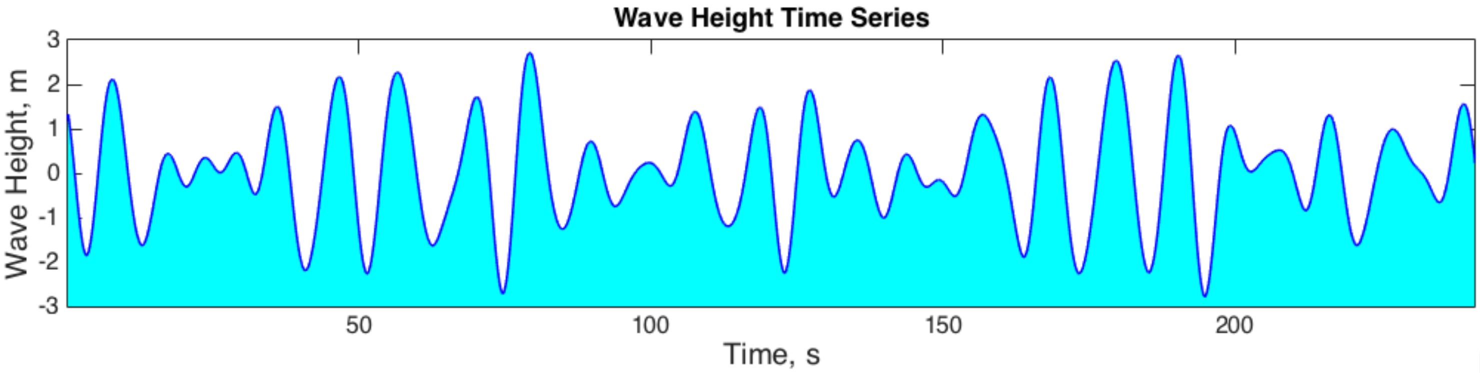
- Superposition
  - Sea surface can be represented by the sum of sinusoids with component periods (T), amplitudes (a), and phase offsets (φ)



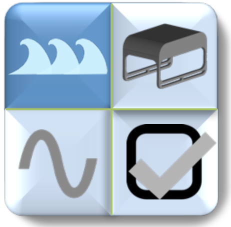
Source: *Waves, Tides and Shallow-Water Processes*, 1999, J. Wright, A. Colling, D. Park

Input Wave Field

Component Wave	1	2	3	4	5	6	7	8
Wave Period, T, s	10	8	12	11	6	7	9	25
Wave Height, H, m	1.8	0.9	1.6	1.3	0.4	0.5	1.1	0.7
Phase, ϕ, rad	$-\frac{\pi}{2}$	$-\frac{\pi}{4}$	$-\frac{5\pi}{8}$	$\frac{4\pi}{13}$	$-\frac{\pi}{15}$	$\frac{\pi}{3}$	$-\frac{\pi}{18}$	$-\frac{7\pi}{4}$



$$\eta(\mathbf{t}) = \sum \frac{H}{2} \cos(k\mathbf{x} - \omega\mathbf{t} + \phi)$$



Linear Wave Theory

- Assumes potential flow: $u = \nabla\phi$

- Dispersion Relation: $\omega^2 = gk\tanh(kd)$
- Wavelength: $L = \frac{gT^2}{2\pi}\left[\tanh\left(\left(\frac{\omega^2 d}{g}\right)^{3/4}\right)\right]^{2/3}$
- Wavenumber: $k = 2\pi/L$

Water Motion under Waves

- Lateral motion in transitional water

$$\mathbf{v}_{p,\mathbf{x}} = \frac{HgT}{2L}\frac{\cosh\frac{2\pi(\mathbf{z}+d)}{L}}{\cosh\frac{2\pi d}{L}}\cos(k\mathbf{x}-\omega\mathbf{t}+\phi),$$

$$\dot{\mathbf{v}}_{p,\mathbf{x}} = \frac{g\pi H}{L}\frac{\cosh\frac{2\pi(\mathbf{z}+d)}{L}}{\cosh\frac{2\pi d}{L}}\sin(k\mathbf{x}-\omega\mathbf{t}+\phi).$$

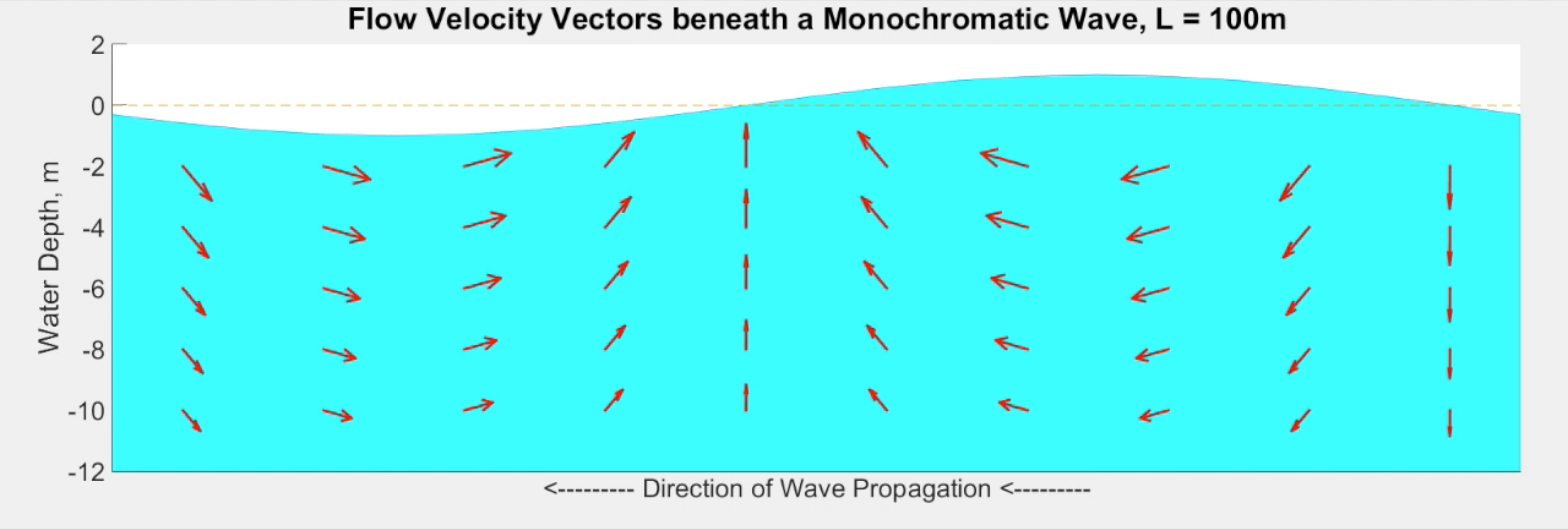
Water Motion under Waves

- Vertical motion in transitional water

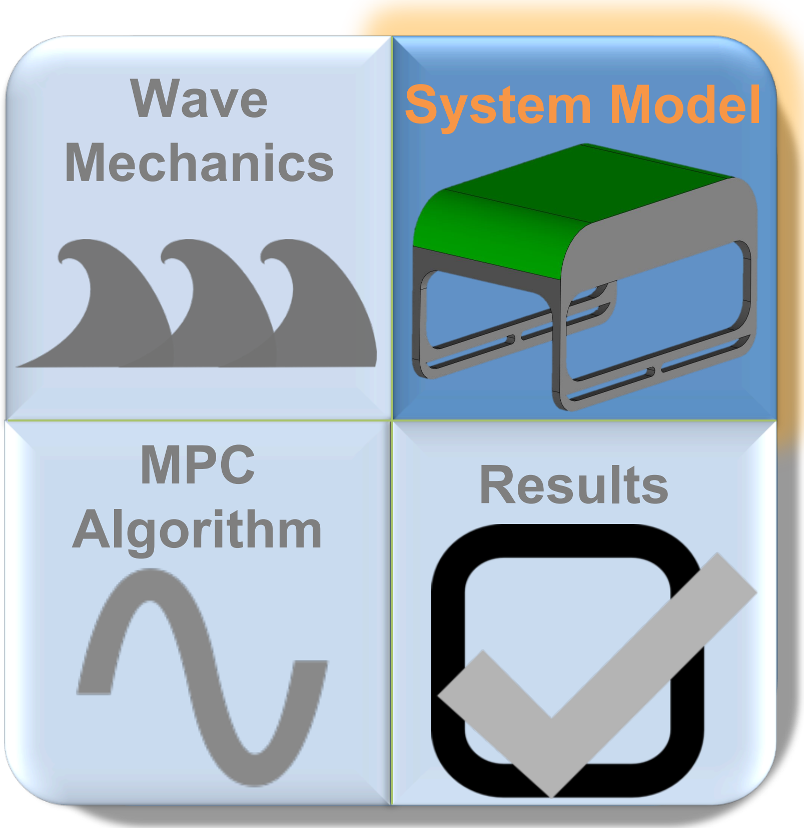
$$\mathbf{v}_{p,\mathbf{z}} = \frac{HgT}{2L} \frac{\sinh \frac{2\pi(\mathbf{z}+d)}{L}}{\cosh \frac{2\pi d}{L}} \sin(k\mathbf{x} - \omega\mathbf{t} + \phi),$$

$$\dot{\mathbf{v}}_{p,\mathbf{z}} = -\frac{g\pi H}{L} \frac{\sinh \frac{2\pi(\mathbf{z}+d)}{L}}{\cosh \frac{2\pi d}{L}} \cos(k\mathbf{x} - \omega\mathbf{t} + \phi).$$

Simulator (Fluid Motion)



Outline



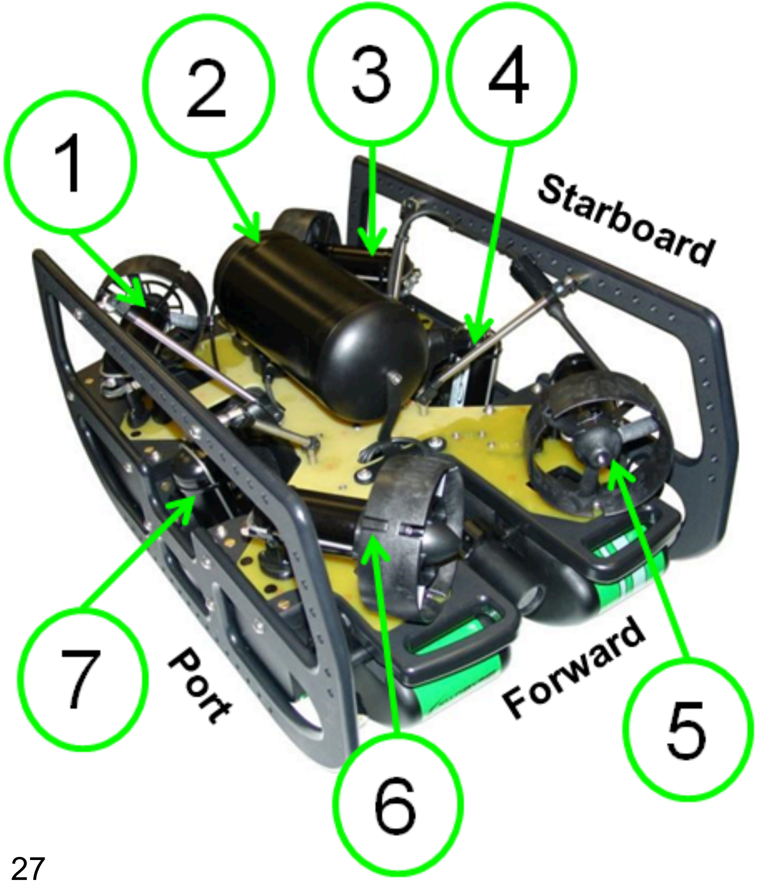
Remotely Operated Vehicle

Oregon State UNIVERSITY

- SeaBotix vLBV300
  - Payload: 10kg
  - Depth Rating: 300m
  - Doppler Velocity Log (DVL)
  - Inertial Measurement Unit (IMU)
  - (6) 100mm brushless DC thrusters
  - Low light color camera with 180° vertical tilt

Source: Teledyne SeaBotix Inc.

Remotely Operated Vehicle



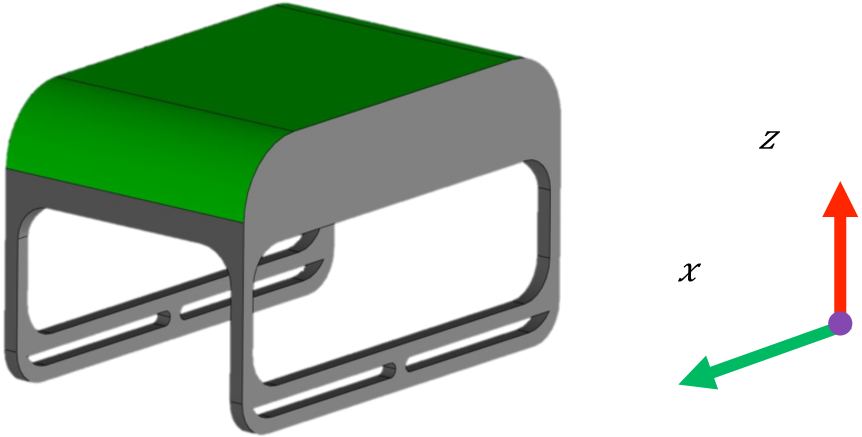
Source: Teledyne SeaBotix Inc.

1. Port aft thruster
2. Electronics tube
3. Starboard aft thruster
4. Starboard vertical thruster
5. Starboard forward thruster
6. Port forward thruster
7. Port vertical thruster

Remotely Operated Vehicle

- Dassault SolidWorks
- Ansys AQWA

Parameter	Symbol	Value
Density of Seawater	ρ_{sea}	$1030\ kg/m^3$
Incident Area, x	$A_{i,\mathbf{x}}$	$0.156\ m^2$
Incident Area, z	$A_{i,\mathbf{z}}$	$0.273\ m^2$
Moment of Inertia, x	I_{xx}	$0.62\ kg\ m^2$
Moment of Inertia, z	I_{zz}	$1.60\ kg\ m^2$
Dry Mass	m_{dry}	$22.2\ kg$
Added Mass, x	$m_{add,x}$	$8.1\ kg$
Added Mass, z	$m_{add,z}$	$36.7\ kg$
Drag Coefficient, x	$c_{d,x}$	0.84
Drag Coefficient, z	$c_{d,z}$	1.06
Max Thruster Force	T_{max}	$29.7\ N$
Thruster Angle, Forward	θ_f	35°
Thruster Angle, Aft	θ_a	45°
Thruster Angle, Vertical	θ_v	20°



Dynamics

Oregon State University

- System differential equation of motion:

$$\mathbf{M}\dot{\mathbf{v}}_a = \mathbf{F}_{thrust} + \mathbf{F}_d + \mathbf{F}_g + \mathbf{F}_c$$

- Simplify, sub inertia and drag relations:

$$m_{dry}\dot{\mathbf{v}}_a + m_{add}\dot{\mathbf{v}}_r = \mathbf{F}_{thrust} + \frac{1}{2}\rho_{sea}A_i c_d |\mathbf{v}_r| \mathbf{v}_r$$

Dynamics

- System differential equation of motion:

$$\mathbf{M}\dot{\mathbf{v}}_a = \mathbf{F}_{thrust} + \mathbf{F}_d + \cancel{\mathbf{F}_g} + \cancel{\mathbf{F}_c}$$

- Simplify, sub inertia and drag relations:

$$m_{dry}\dot{\mathbf{v}}_a + m_{add}\dot{\mathbf{v}}_r = \mathbf{F}_{thrust} + \frac{1}{2}\rho_{sea}A_i c_d|\mathbf{v}_r|\mathbf{v}_r$$

Dynamics

- Substitute $\boldsymbol{v}_p$ such that $\boldsymbol{v}_a = \boldsymbol{v}_r + \boldsymbol{v}_p$:

$$\begin{bmatrix} m_{dry} + m_{add,\mathrm{x}} \\ m_{dry} + m_{add,\mathrm{z}} \end{bmatrix} \begin{bmatrix} \ddot{\mathrm{x}} \\ \ddot{\mathrm{z}} \end{bmatrix} = \begin{bmatrix} \mathbf{F}_{thrust,\mathrm{x}} \\ \mathbf{F}_{thrust,\mathrm{z}} \end{bmatrix} + \begin{bmatrix} m_{add,\mathrm{x}} \\ m_{add,\mathrm{z}} \end{bmatrix} \begin{bmatrix} \dot{\mathbf{v}}_{p,\mathrm{x}} \\ \dot{\mathbf{v}}_{p,\mathrm{z}} \end{bmatrix} + \frac{\rho_{sea}}{2} \begin{bmatrix} A_{i,\mathrm{x}} C_{d,\mathrm{x}} \\ A_{i,\mathrm{z}} C_{d,\mathrm{z}} \end{bmatrix} \begin{bmatrix} |\dot{\mathrm{x}} - \mathbf{v}_{p,\mathrm{x}}|(\dot{\mathrm{x}} - \mathbf{v}_{p,\mathrm{x}}) \\ |\dot{\mathrm{z}} - \mathbf{v}_{p,\mathrm{z}}|(\dot{\mathrm{z}} - \mathbf{v}_{p,\mathrm{z}}) \end{bmatrix}$$

  - where: $\boldsymbol{v}_{a,\,x} = \boldsymbol{x}$ and $\boldsymbol{v}_{a,\,z} = \boldsymbol{z}$

Dynamics

- In state space form:

$$\dot{\Upsilon} = \begin{bmatrix} \dot{x} & \ddot{x} & \dot{z} & \ddot{z} \end{bmatrix}^T = A\Upsilon + Bu + D$$

  - where: $\boldsymbol{x} = \boldsymbol{v} \downarrow a, \boldsymbol{x}$ and $\boldsymbol{z} = \boldsymbol{v} \downarrow a, \boldsymbol{z}$

Dynamics

$$A\Upsilon = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \dot{\mathbf{x}} \\ \mathbf{z} \\ \dot{\mathbf{z}} \end{bmatrix}$$

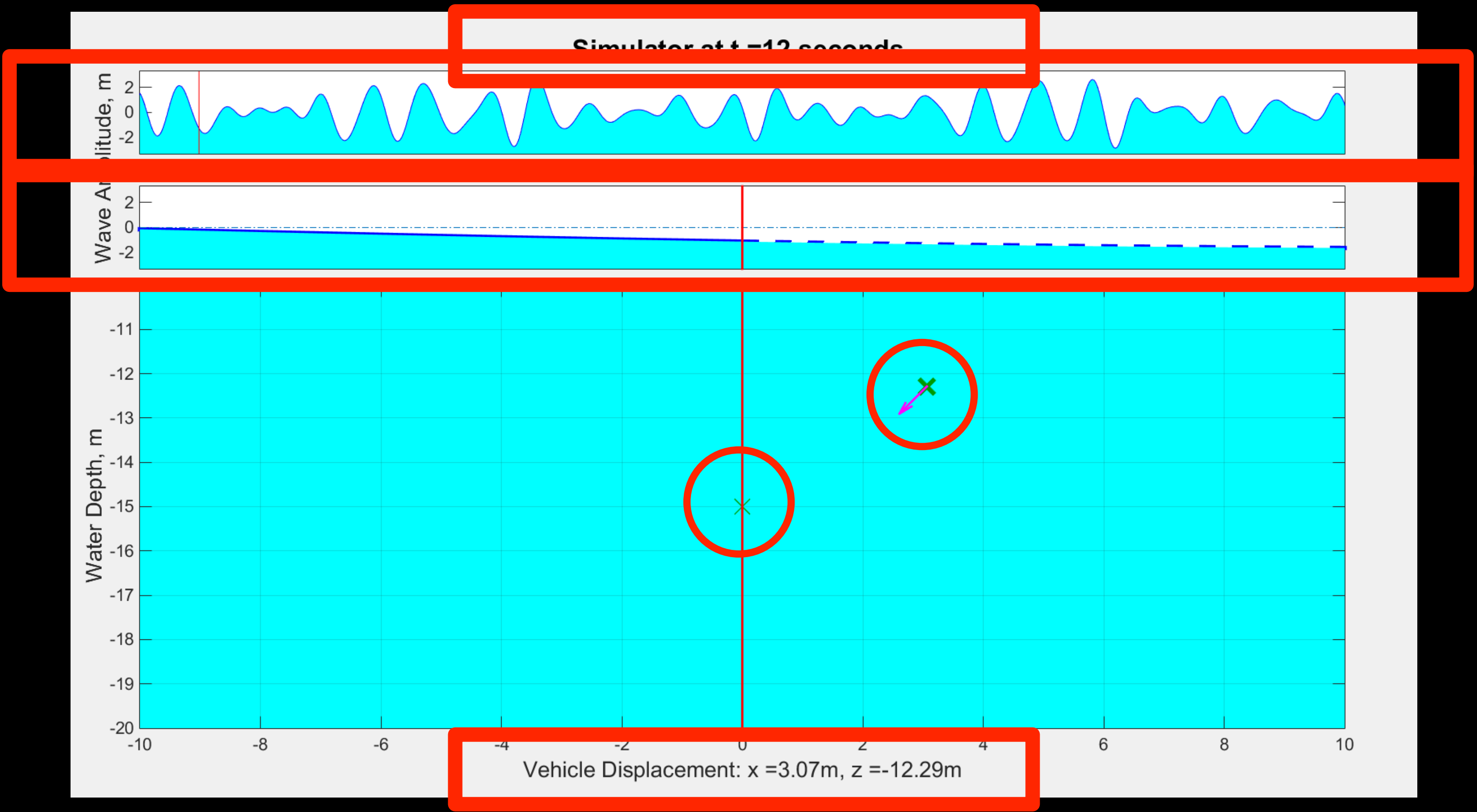
Dynamics

$$\mathbf{Bu} = \frac{T_{max}}{m_{dry}} \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ cos\theta_f & cos\theta_f & -cos\theta_a & -cos\theta_a & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -cos\theta_v & -cos\theta_v \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \end{bmatrix}$$

Dynamics

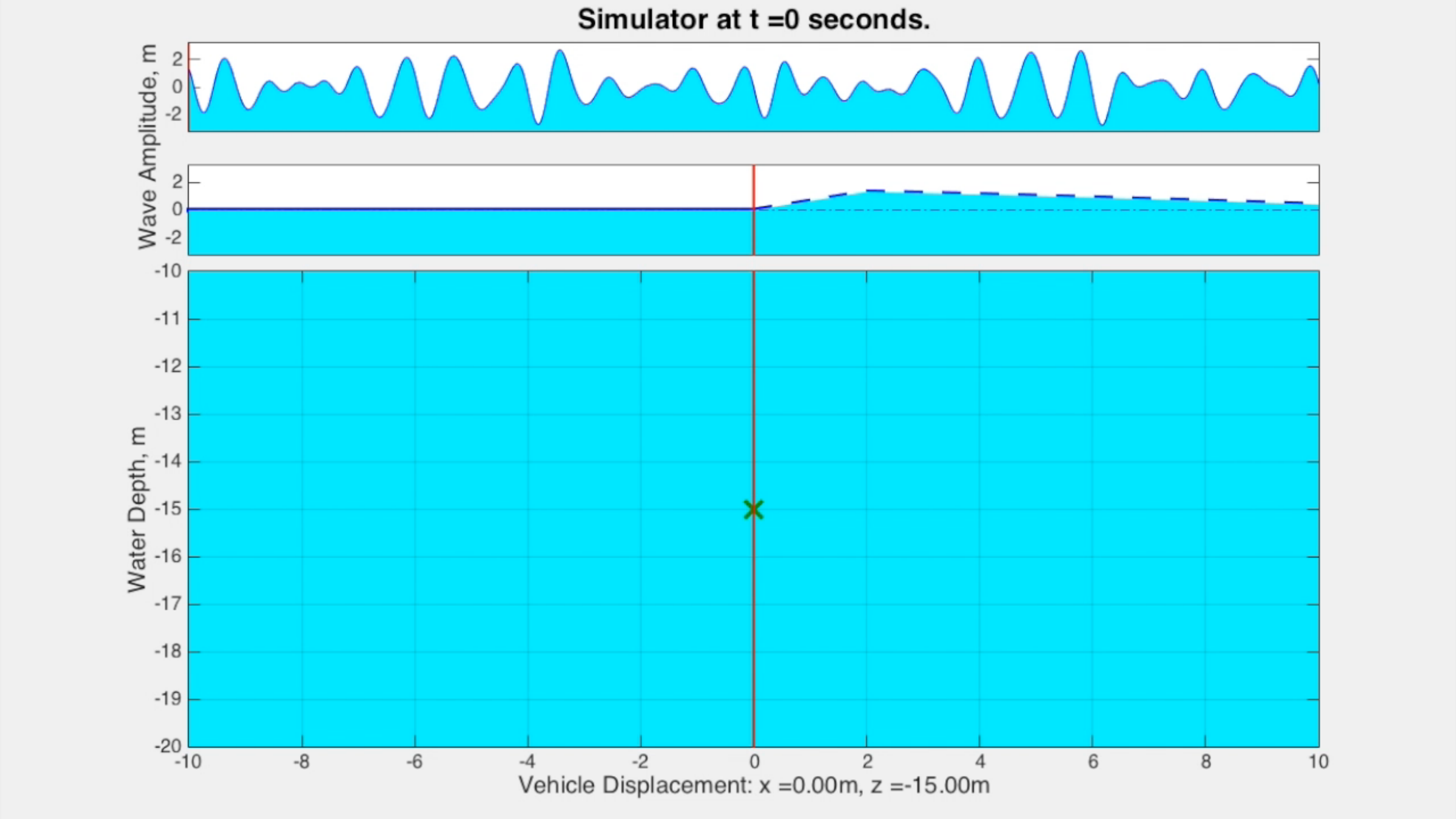
$$\mathbf{D} = \begin{bmatrix} 0 \\ \dfrac{\dot{\mathbf{v}}_{p,\mathbf{x}}}{m_{dry}} + \dfrac{\rho_{sea} A_{i,\mathbf{x}} C_{d,\mathbf{x}}}{2(m_{dry}+m_{add,\mathbf{x}})} |\dot{\mathbf{x}} - \mathbf{v}_{p,\mathbf{x}}|(\dot{\mathbf{x}} - \mathbf{v}_{p,\mathbf{x}}) \\ 0 \\ \dfrac{\dot{\mathbf{v}}_{p,\mathbf{z}}}{m_{dry}} + \dfrac{\rho_{sea} A_{i,\mathbf{x}} C_{d,\mathbf{z}}}{2(m_{dry}+m_{add,\mathbf{z}})} |\dot{\mathbf{z}} - \mathbf{v}_{p,\mathbf{z}}|(\dot{\mathbf{z}} - \mathbf{v}_{p,\mathbf{z}}) \end{bmatrix}$$

Simulator



Wave Amplitude, m
2
0
-2
Water Depth, m
-11
-12
-13
-14
-15
-16
-17
-18
-19
-20
-10
-8
-6
-4
-2
0
2
4
6
8
10
Vehicle Displacement: x =3.07m, z =-12.29m

Simulator (Drifting Robot)



Feedback Control

- Purely Reactive
- Use as base of comparison
- Position Derivative (PD) control

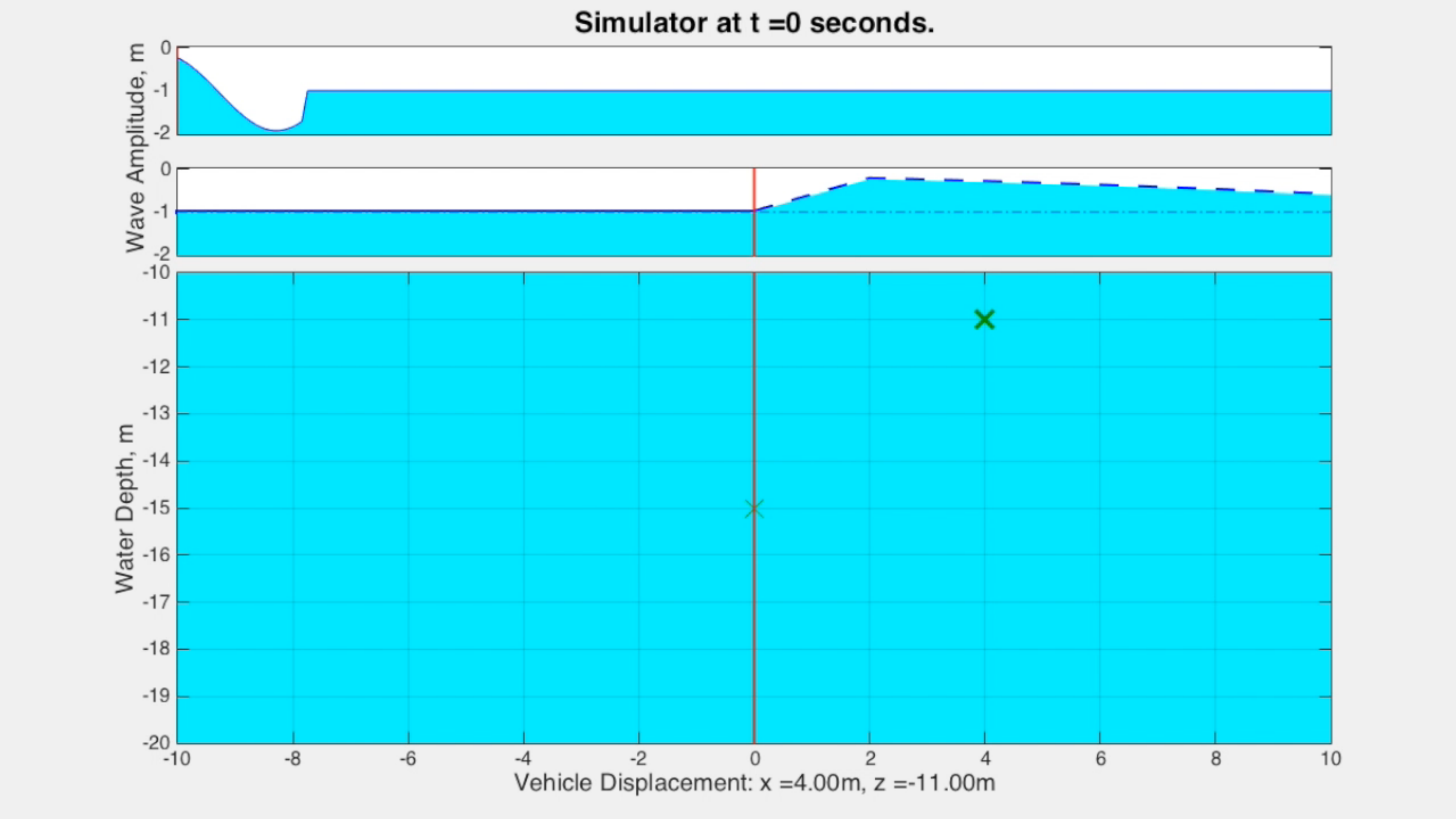
PD Controller

- Positional Error: $\epsilon_P = \boldsymbol{r}_{target} - \boldsymbol{r}_n$
- Derivative Error: $\epsilon_D = \epsilon_{P,n} - \epsilon_{P,(n-1)}$
- $[u_1 \;\; u_2 \;\; u_3 \;\; u_4]^T = K_{P,\boldsymbol{x}} \, \epsilon_{P,\boldsymbol{x}} + K_{D,\boldsymbol{x}} \, \epsilon_{D,\boldsymbol{x}}$
- $[u_5 \;\; u_6]^T = K_{P,\boldsymbol{z}} \, \epsilon_{P,\boldsymbol{z}} + K_{D,\boldsymbol{z}} \, \epsilon_{D,\boldsymbol{z}}$

PD Controller

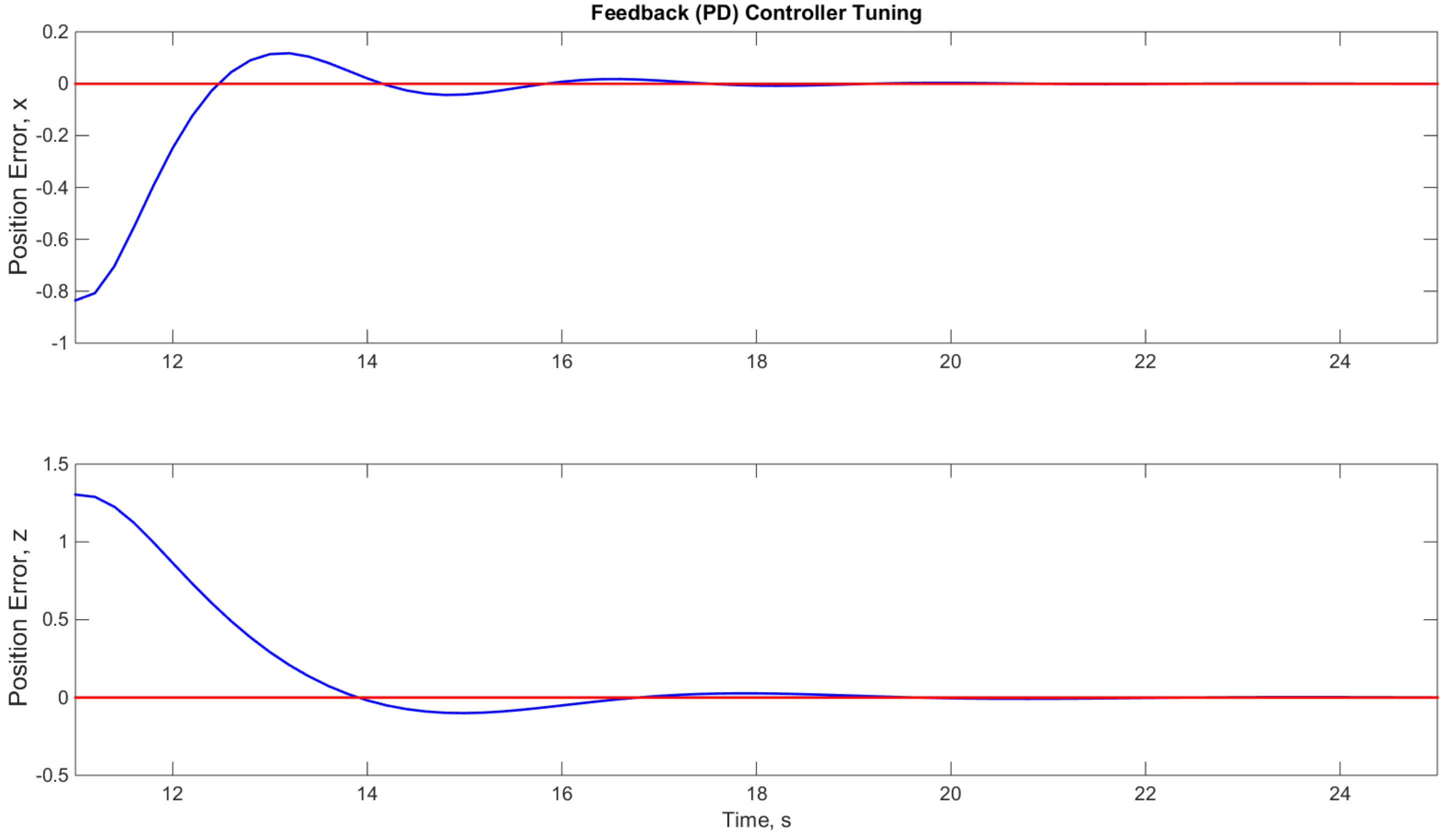
- Positional Error: $\epsilon_P = \mathbf{r}_{target} - \mathbf{r}_n$
- Derivative Error: $\epsilon_D = \epsilon_{P,n} - \epsilon_{P,(n-1)}$
- $[u_1 \; u_2 \; u_3 \; u_4]^T = K_{P,\mathbf{x}} \epsilon_{P,\mathbf{x}} + K_{D,\mathbf{x}} \epsilon_{D,\mathbf{x}}$
- $[u_5 \; u_6]^T = K_{P,\mathbf{z}} \epsilon_{P,\mathbf{z}} + K_{D,\mathbf{z}} \epsilon_{D,\mathbf{z}}$

PD Tuning



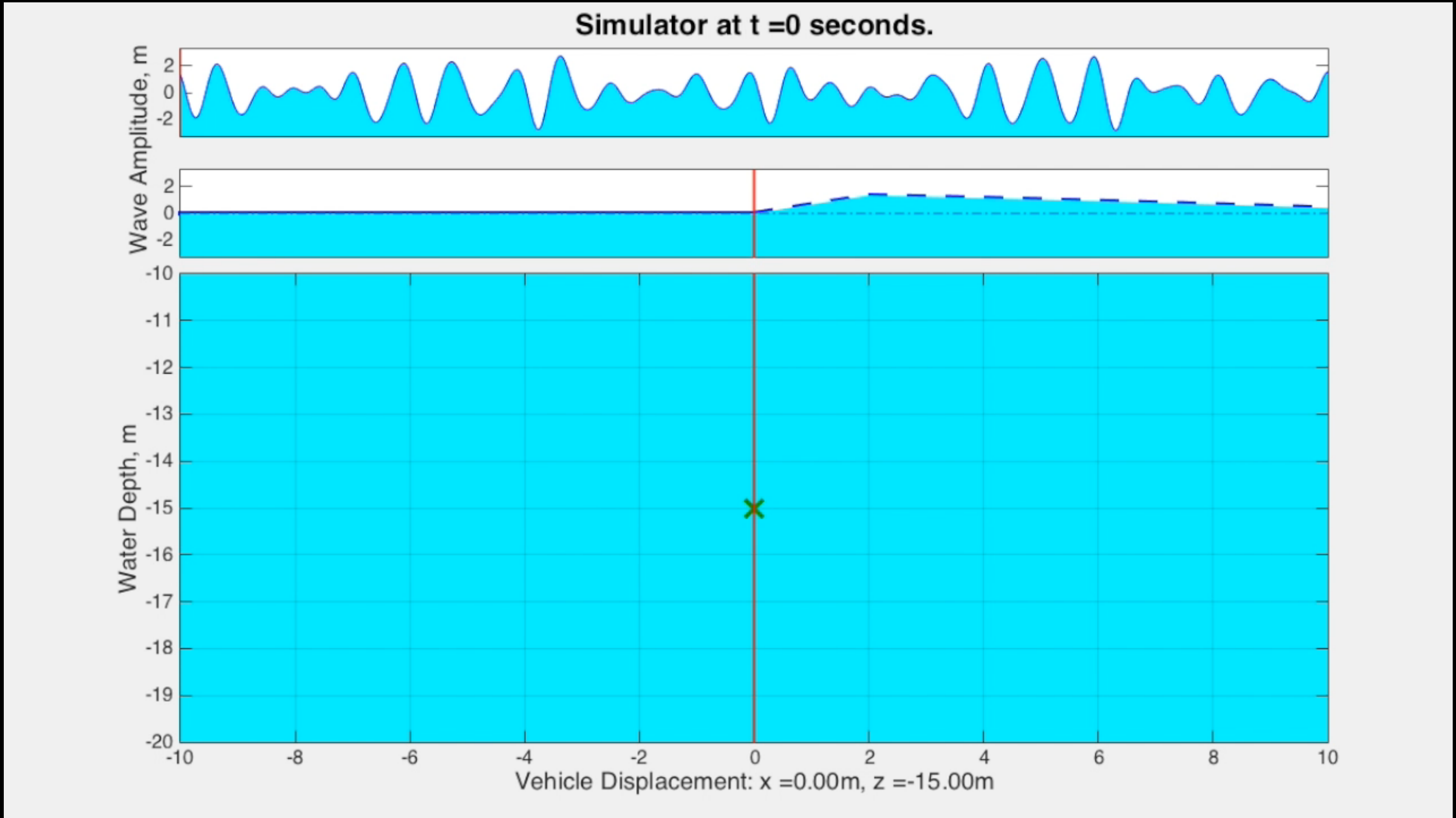
Simulator at t =0 seconds.
Wave Amplitude, m
0
-1
-2
0
-1
-2
Water Depth, m
-10
-11
-12
-13
-14
-15
-16
-17
-18
-19
-20
-10
-8
-6
-4
-2
0
2
4
6
8
10
Vehicle Displacement: x =4.00m, z =-11.00m

PD Tuning

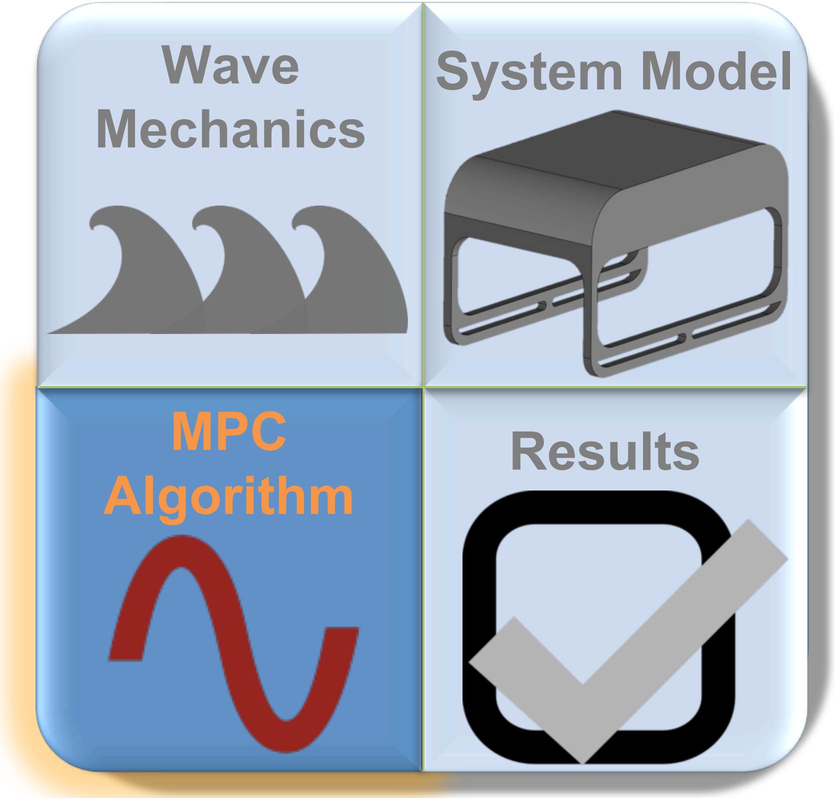


Feedback (PD) Controller Tuning
Position Error, x
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
12
14
16
18
20
22
24
Position Error, z
1.5
1
0.5
0
-0.5
Time, s

Simulator (PD Robot)



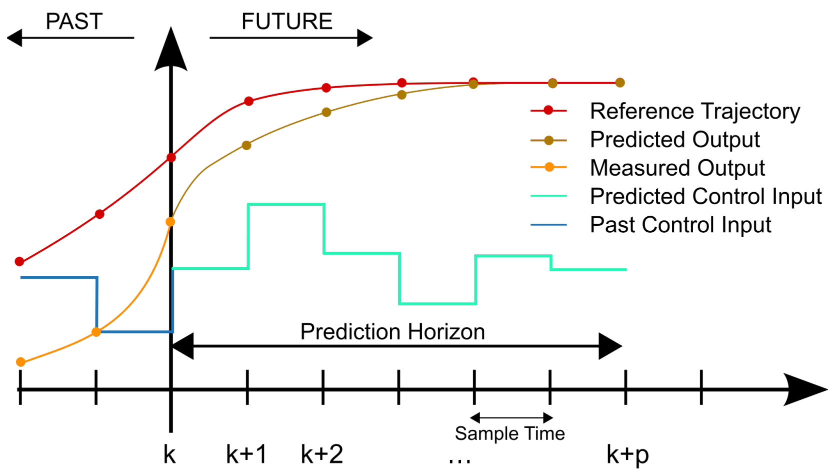
Outline



Model Predictive Control

Oregon State UNIVERSITY

- State estimator
- Cost function
- Receding horizon



- **Objective:** Find the control actions that minimize the distance between the desired and predicted states.

Cost Function

$$J = \sum_{k=1}^{N} \left[\boldsymbol{Y}_{target} - \boldsymbol{Y}_{k}(\boldsymbol{u}_{k}) \right]^{2} + \beta \boldsymbol{u}_{k}^{2}$$

$$\boldsymbol{u}_{1:N}^{*} = \arg\min_{\boldsymbol{u}_{1:N}} J(\boldsymbol{u}_{1:N})$$

Gradient Descent Optimization

Oregon State University

- Perturb new control input by Jacobian
  - Minimized as optimal control action is approached

$$\partial \boldsymbol{J}/\partial \boldsymbol{u} = \frac{\boldsymbol{J}_n - \boldsymbol{J}_{n-1}}{\boldsymbol{u}_n - \boldsymbol{u}_{n-1}}$$

Algorithm

Oregon State UNIVERSITY

Algorithm 1 MPC for ocean wave station-keeping where t is a time vector, λ contains all relevant wave spectra data, and Υ represents states in time and space.

procedure $MPC(\ t,\ \lambda,\ \text{robot},\ \Upsilon_{target}\ )$
    $n \leftarrow 1$
    $\eta \leftarrow \text{LOADSEASTATE}(\ t,\ \lambda,\ \Upsilon_{initial}\ )$
    while $n <$ simulatorOff **do**
        $\text{input} \leftarrow \text{GETFORECAST}(\ t,\ \text{robot},\ \lambda,\ \Upsilon_{target},\ n\ )$
        $\text{robot} \leftarrow \text{MOVEROBOT}(\ t,\ \text{robot},\ \lambda,\ \text{input},\ n\ )$
        $n \leftarrow n + 1$

Algorithm

Oregon State UNIVERSITY

Algorithm 1 MPC for ocean wave station-keeping where t is a time vector, λ contains all relevant wave spectra data, and Υ represents states in time and space.

procedure $MPC(\ t, \lambda,$ robot, $\Upsilon_{target}\ )$

    $n \leftarrow 1$

    $\eta \leftarrow$ LOADSEASTATE($t, \lambda, \Upsilon_{initial}$)

    while $n <$ simulatorOff **do**

        input $\leftarrow$ GETFORECAST(t, robot, λ, Υ_{target}, n)

        robot $\leftarrow$ MOVEROBOT(t, robot, λ, input, n)

        $n \leftarrow n + 1$

Algorithm

Algorithm 1 MPC for ocean wave station-keeping where t is a time vector, λ contains all relevant wave spectra data, and Υ represents states in time and space.

procedure $MPC(\ t, \lambda,$ robot, $\Upsilon_{target}\ )$
    $n \leftarrow 1$
    $\eta \leftarrow$ LOADSEASTATE($t, \lambda, \Upsilon_{initial}$)
    while $n <$ simulatorOff **do**
        input $\leftarrow$ GETFORECAST(t, robot, $\lambda, \Upsilon_{target}, n$)
        robot $\leftarrow$ MOVEROBOT(t, robot, λ, input, n)
        $n \leftarrow n + 1$

Algorithm

Algorithm 1 MPC for ocean wave station-keeping where t is a time vector, λ contains all relevant wave spectra data, and Υ represents states in time and space.

procedure $MPC(\ t,\ \lambda,\ \text{robot},\ \Upsilon_{target}\ )$
    $n \leftarrow 1$
    $\eta \leftarrow \textsc{LoadSeaState}(\ t,\ \lambda,\ \Upsilon_{initial}\ )$
    while $n <$ simulatorOff **do**
        input $\leftarrow \textsc{GetForecast}(\ t,\ \text{robot},\ \lambda,\ \Upsilon_{target},\ n\ )$
        robot $\leftarrow \textsc{MoveRobot}(\ t,\ \text{robot},\ \lambda,\ \text{input},\ n\ )$
        $n \leftarrow n + 1$

Algorithm

Algorithm 1 MPC for ocean wave station-keeping where t is a time vector, λ contains all relevant wave spectra data, and Υ represents states in time and space.

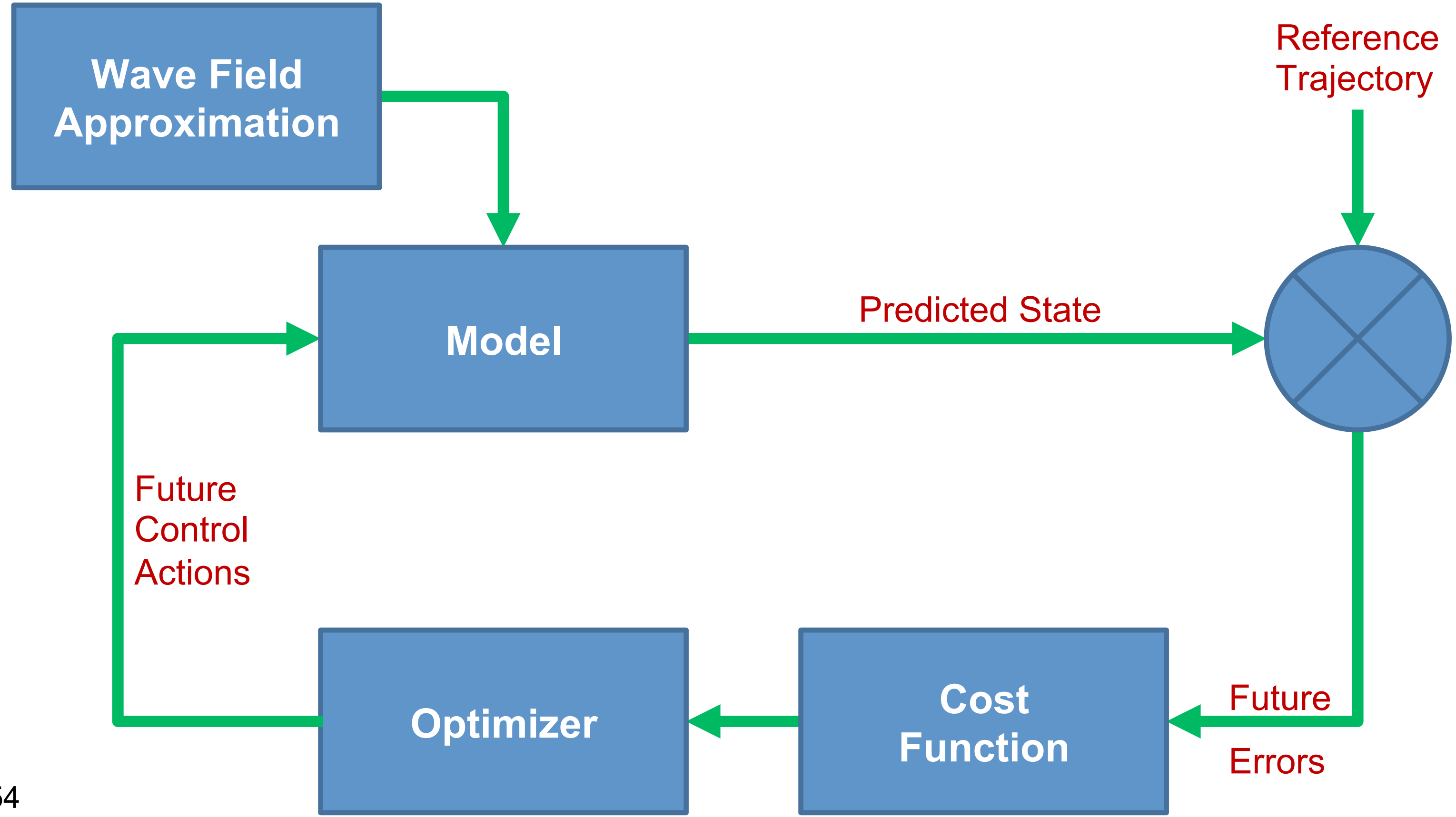
```
procedure MPC( t, λ, robot, Υ_target )
    n ← 1
    η ← LOADSEASTATE( t, λ, Υ_initial )
    while n < simulatorOff do
        input ← GETFORECAST( t, robot, λ, Υ_target, n )
        robot ← MOVEROBOT( t, robot, λ, input, n )
        n ← n + 1
```

Algorithm

Algorithm 1 MPC for ocean wave station-keeping where t is a time vector, λ contains all relevant wave spectra data, and Υ represents states in time and space.

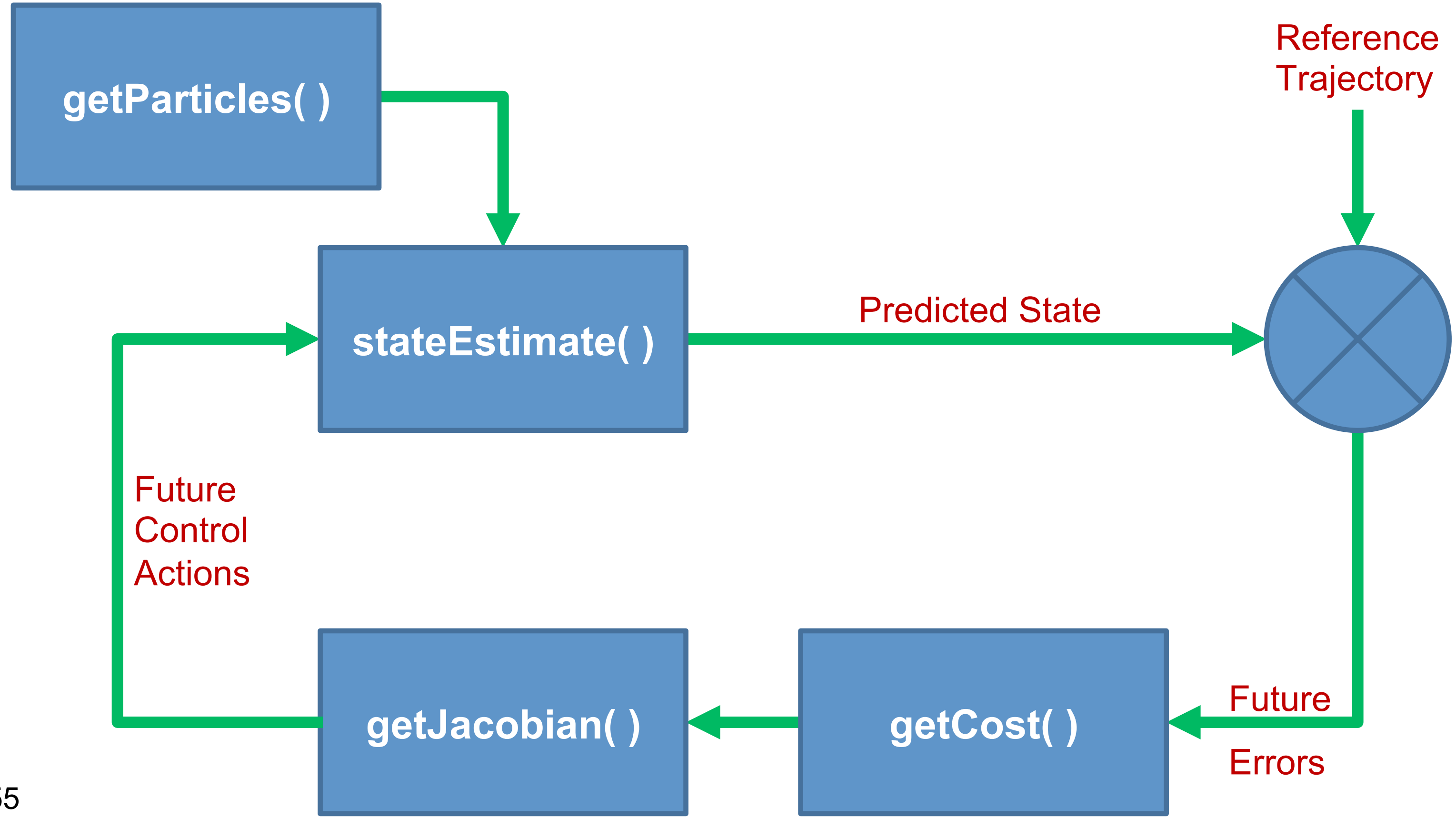
procedure $MPC(\; t, \; \lambda, \; \text{robot}, \; \Upsilon_{target} \;)$
    $n \leftarrow 1$
    $\eta \leftarrow$ LOADSEASTATE$(\; t, \; \lambda, \; \Upsilon_{initial} \;)$
    while $n <$ simulatorOff **do**
        input $\leftarrow$ GETFORECAST$(\; t, \; \text{robot}, \; \lambda, \; \Upsilon_{target}, \; n \;)$
        robot $\leftarrow$ MOVEROBOT$(\; t, \; \text{robot}, \; \lambda, \; \text{input}, \; n \;)$
        $n \leftarrow n + 1$

Algorithm



Wave Field Approximation
Reference Trajectory
Model
Predicted State
Future Control Actions
Optimizer
Cost Function
Future Errors

Algorithm


getParticles( )
Reference Trajectory
stateEstimate( )
Predicted State
Future Control Actions
getJacobian( )
getCost( )
Future Errors

Algorithm

while $i <$ maxIterations **and** $\delta >$ exitCriteria **do**

  $u_{i+1} \leftarrow u_i - \delta$

  for $k \in [\ 1,\ 2,\ ...,\ N\ ]$ **do**

    $\mathbf{v}_p, \dot{\mathbf{v}}_p \leftarrow$ GETPARTICLES(t, $\Upsilon_i(k)$, λ)

    $\Lambda \leftarrow \mathbf{v}_p, \dot{\mathbf{v}}_p$

    $\Upsilon_{i+1}(k) \leftarrow$ STATEESTIMATE(t, robot, Λ, $u_{n+1}(k)$)

    $J_{i+1}(k) \leftarrow$ GETCOST(Υ_{target}, $\Upsilon_{i+1}(k)$)

  $\delta \leftarrow$ GETJACOBIAN(J_i, J_{i+1}, u_i, u_{i+1})

  $u_i \leftarrow u_{i+1}$

  $J_i \leftarrow J_{i+1}$

  $\Upsilon_i \leftarrow \Upsilon_{i+1}$

  $i \leftarrow i + 1$

return u_{i+1}

Algorithm

while $i < \text{maxIterations}$ **and** $\delta > \text{exitCriteria}$ **do**

$u_{i+1} \leftarrow u_i - \delta$

for $k \in [\ 1,\ 2,\ \ldots,\ N\ ]$ **do**

$\mathbf{v}_p, \dot{\mathbf{v}}_p \leftarrow \textsc{getParticles}(\ t,\ \Upsilon_i(k),\ \lambda\ )$

$\Lambda \leftarrow \mathbf{v}_p, \dot{\mathbf{v}}_p$

$\Upsilon_{i+1}(k) \leftarrow \textsc{stateEstimate}(\ t,\ \text{robot},\ \Lambda,\ u_{n+1}(k)\ )$

$J_{i+1}(k) \leftarrow \textsc{getCost}(\ \Upsilon_{target},\ \Upsilon_{i+1}(k)\ )$

$\delta \leftarrow \textsc{getJacobian}(\ J_i,\ J_{i+1},\ u_i,\ u_{i+1}\ )$

$u_i \leftarrow u_{i+1}$

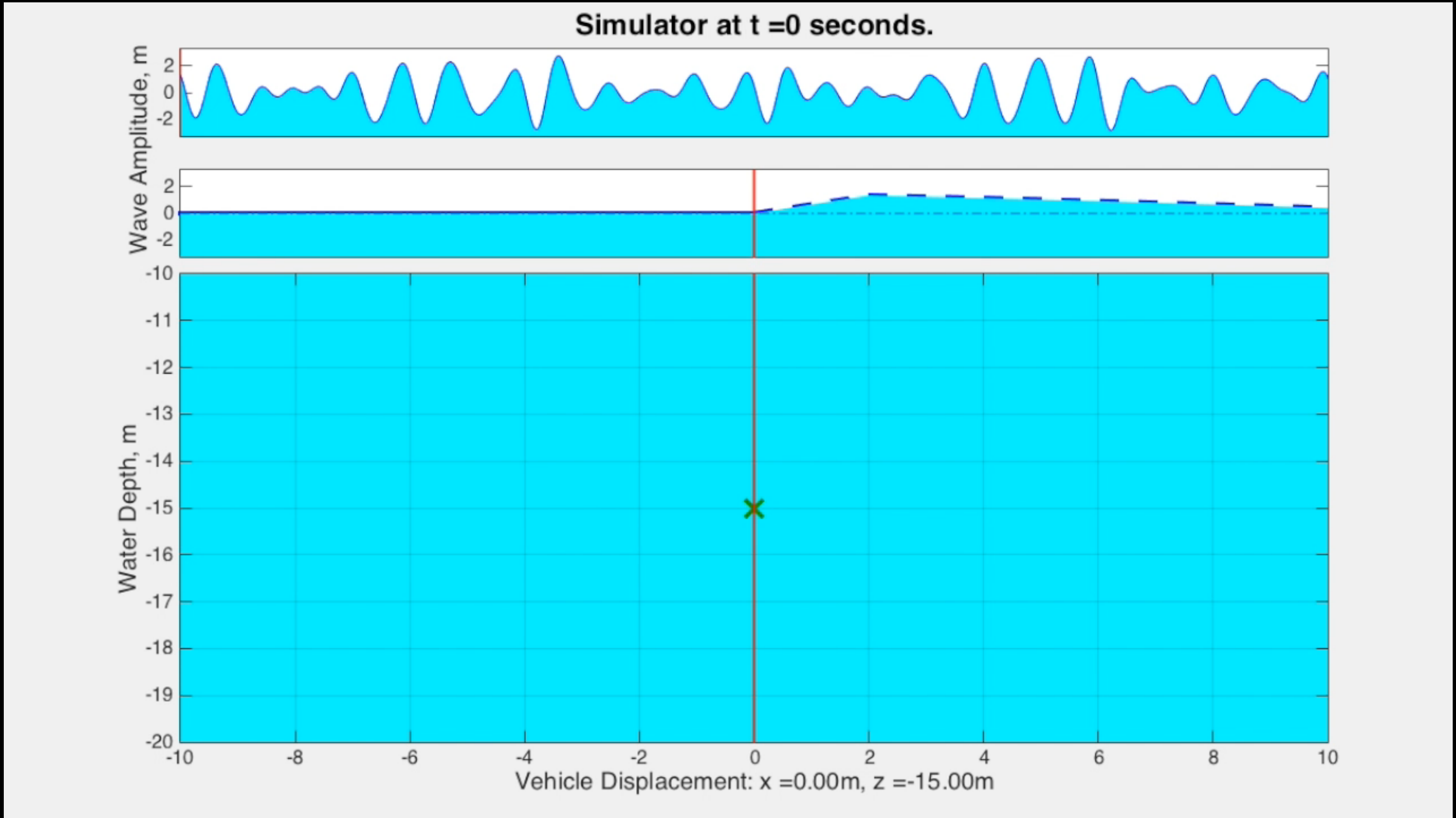
$J_i \leftarrow J_{i+1}$

$\Upsilon_i \leftarrow \Upsilon_{i+1}$

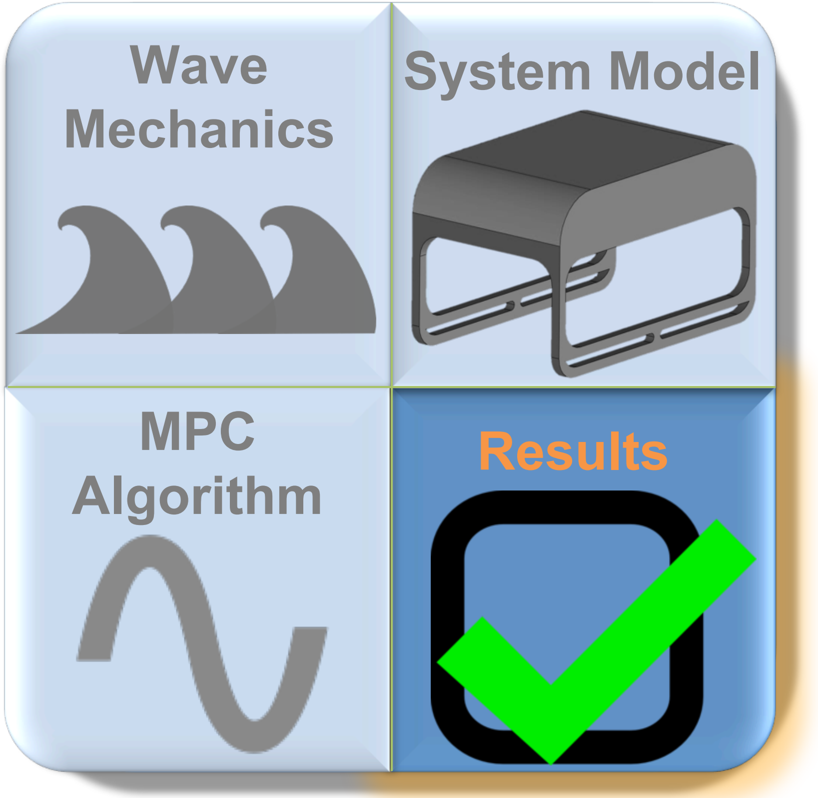
$i \leftarrow i + 1$

return u_{i+1}

Simulator (MPC Robot)



Outline



Results

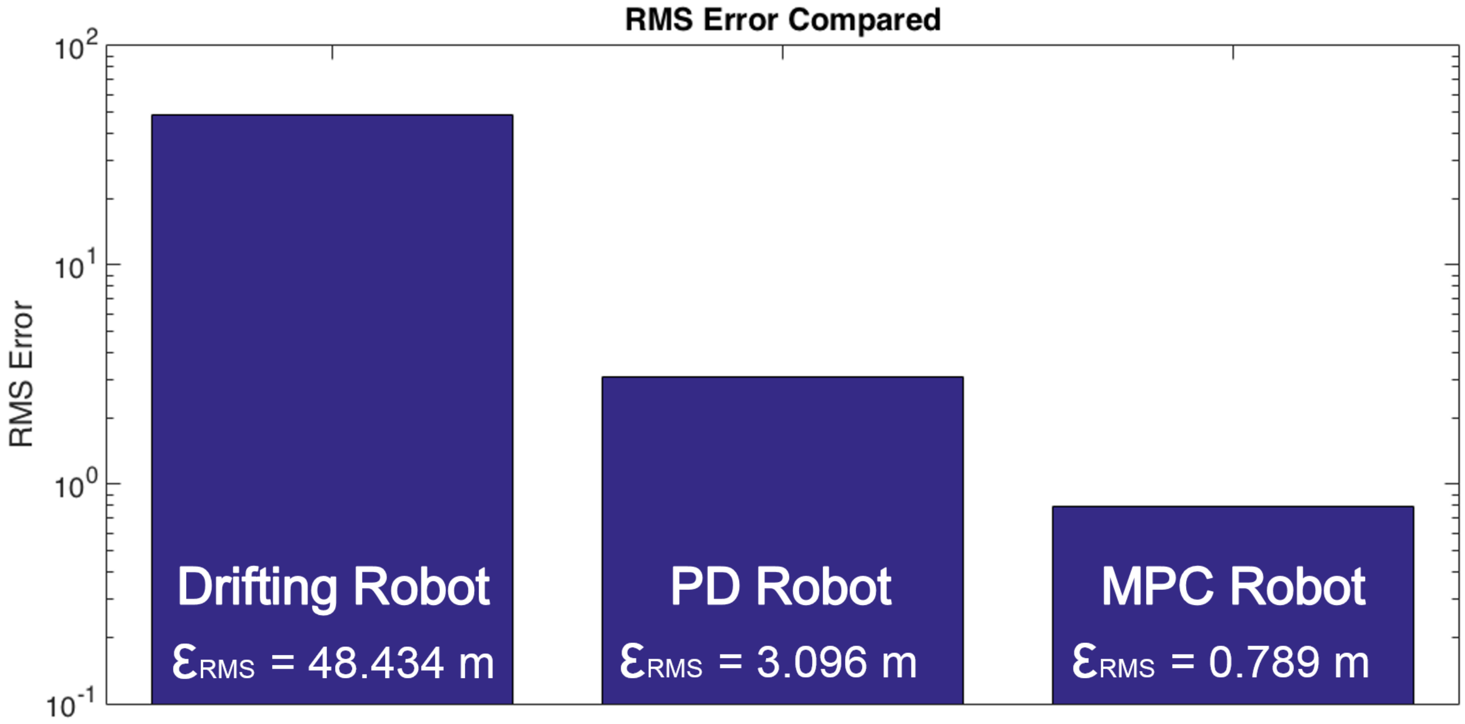
- Determine best performing horizon
- MPC performance versus PD control
- Resistance to noisy sensor observations

Ideal Horizon

Horizon, s	0.2	0.4	**0.8**	1.0	1.6	2.0	3.0
ϵ_{RMS}, m	5.02	2.11	**0.79**	0.65	0.29	0.05	$9.0E\text{-}6$
$\Sigma\, t_{Calc}$, s	1658	507.1	**93.8**	233.2	7808	21886	101252
$\bar{t}_{Calc}$, s	6.90	2.11	**0.39**	0.97	32.53	91.19	421.88

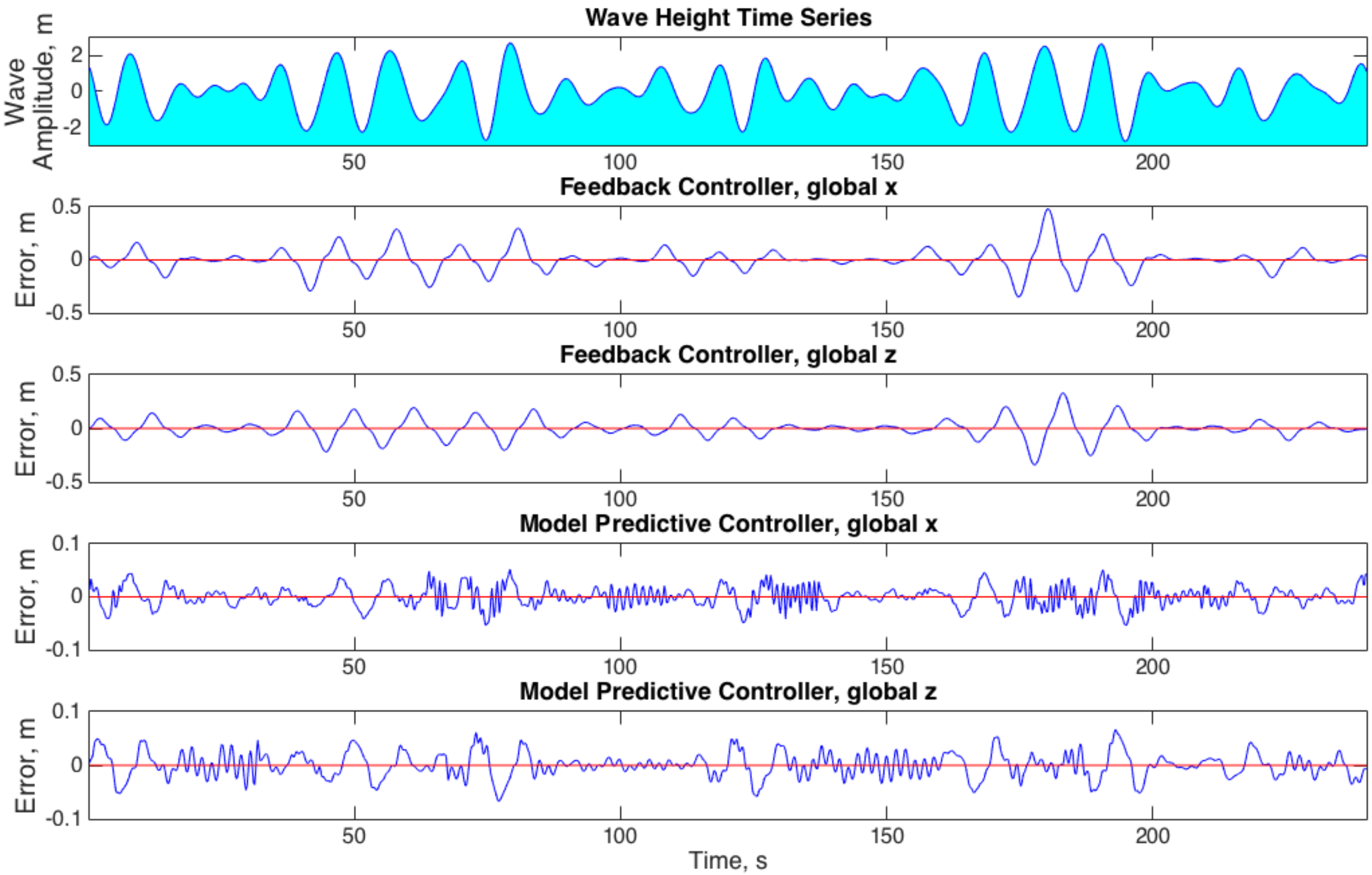
- 0.8 s best balance of low error and time
- Poor performers on low & high end
- Total time: 240 s discretized by 0.2 s

MPC Performance



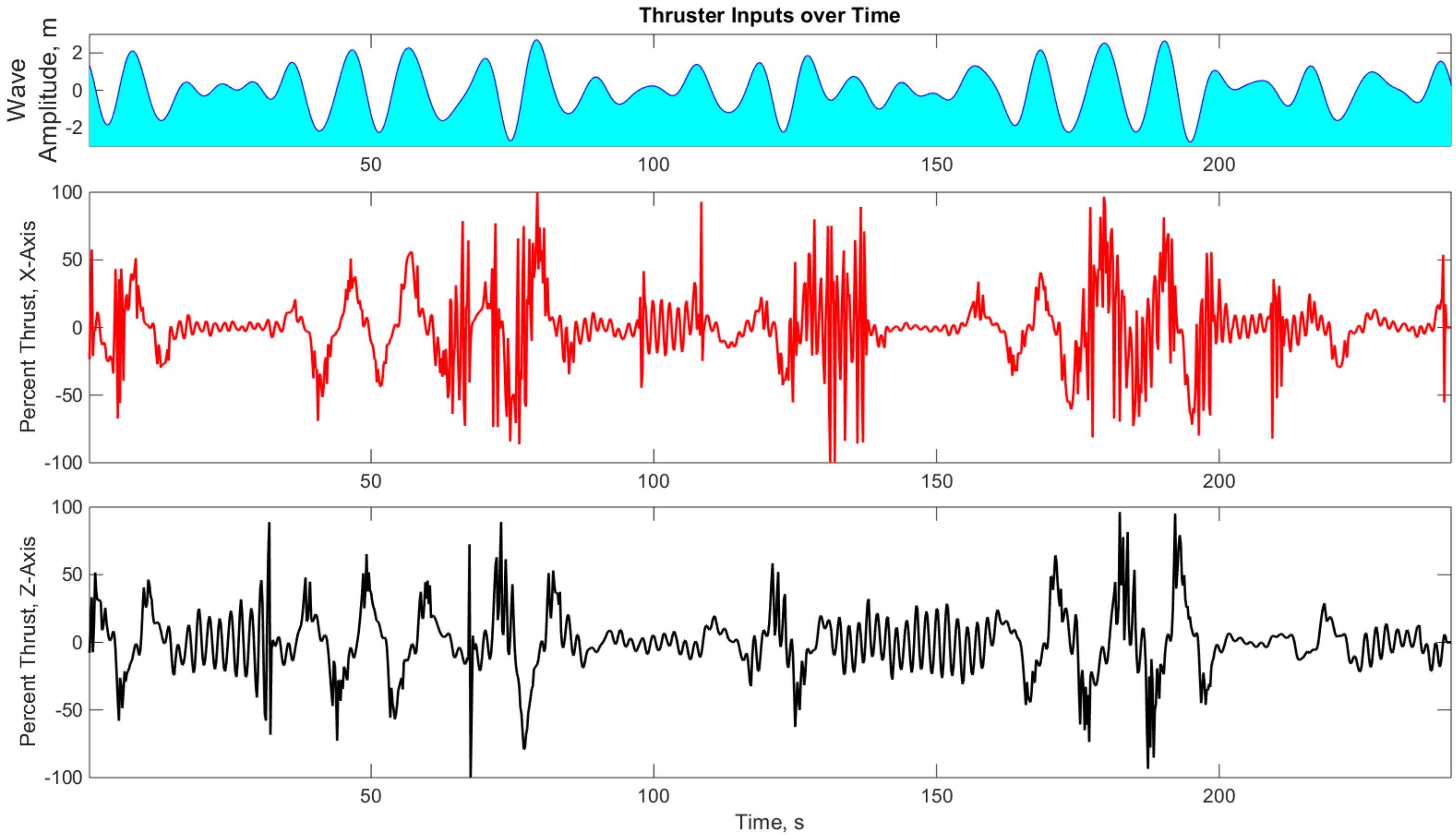
- 74% reduction in position error over PD

MPC Performance



MPC Performance

Oregon State UNIVERSITY

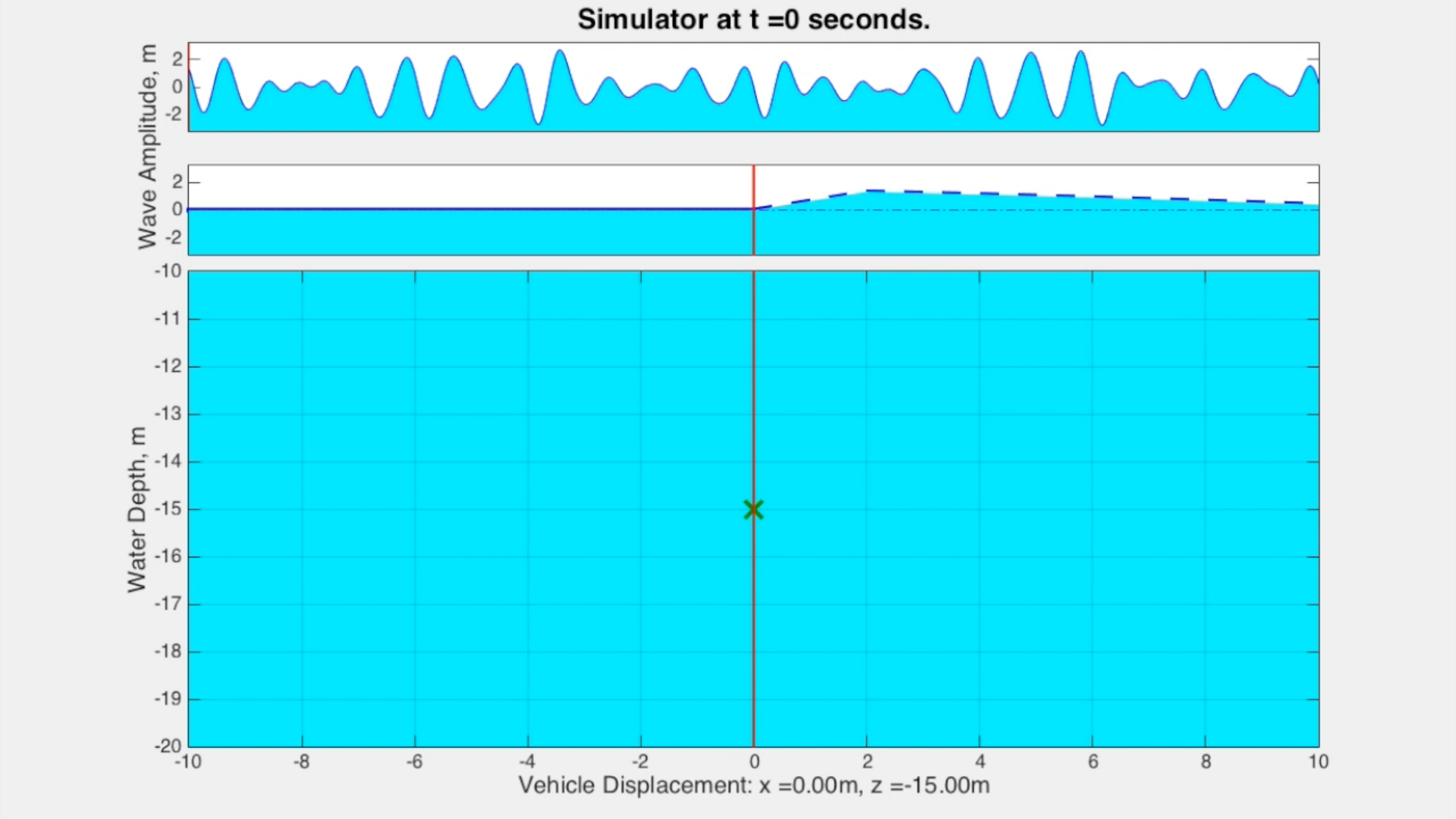


Impact of Gaussian Noise

Oregon State University

- Observations of perceived wave state
- *H* term assigned maximum variance
- Minimal localization noise assumed
  - Deterministic PD case

Simulator (MPC w/ noise)



Impact of Gaussian Noise

Oregon State University

- 50 simulations
    - *getForecast( )* gets new noisy wave field at n^{th} step
- 44% reduction over PD
    - $\epsilon_{RMS} = 1.737 m$
    - $\sigma = 0.059$
- Notable run time increase

Summary of Contributions

- A feedforward control (MPC) method that can **forecast** and **compensate** for impending wave forces
- Application of the MPC algorithm to a simulated station-keeping robot
- Comparison of the MPC algorithm against traditional feedback (PD) control
- Algorithm resistance to noisy sensor observations of wave field parameters.
- Recommendations for choosing a prediction horizon

Future Work

- Real-time wave prediction methods
- Neuro-Evolutionary control methods
  - Hydrodynamic simulation software packages
- System dynamics expanded
- More efficient optimization

Sponsors and Affiliations

Oregon State UNIVERSITY

DEPARTMENT OF ENERGY • UNITED STATES OF AMERICA •

PCC Precision Castparts Corp.

NNMREC

Oregon State University Robotics Club
oregonstate.edu/groups/osurc

OSU Oregon State UNIVERSITY
College of Engineering

CEOAS
College of Earth, Ocean, and Atmospheric Sciences

W.M.KECK FOUNDATION

Office of Naval Research ONR Science & Technology

OOI
OCEAN OBSERVATORIES INITIATIVE

O.H. HINSDALE WAVE RESEARCH LABORATORY
OREGON STATE UNIVERSITY

Questions?

Oregon State UNIVERSITY