

# An Adaptive Controller for a Dual-Thruster Autonomous Surface Vehicle

Kevin Becker<sup>1</sup>

## I. INTRODUCTION

Marine vehicle maneuvering is a unique area in control theory. This is because marine vehicles have many parameters which are all incredibly difficult to estimate. This environment has much to gain from adaptive controllers.

## II. OVERVIEW

At the MIT Marine Autonomy Lab, other researchers have done work on adaptive controllers as well. Using some supporting code reduced the amount of new software required to test vehicles on the water. Additionally, this software stack has been tested and works with other adaptive controllers.

### A. State-Space Selection

For any controls project it is important to determine a proper state-space. If important variables are ignored, the state-transition function will be incorrect, and the model will be inaccurate. Too many variables may result in an unnecessarily complicated, potentially uncontrollable model.

The vehicle used in this project has two thrusters and six degrees of freedom (see figure 1). However, certain degrees of freedom can be removed from the model.

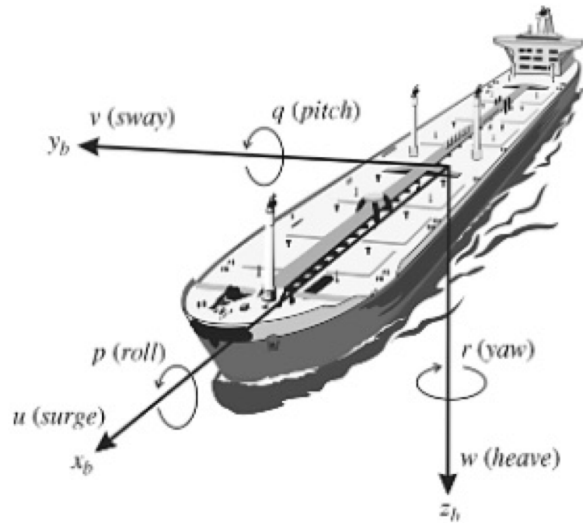


Fig. 1: Coordinate System used for this project. Source: [1].

Variations in draft (heave motions, in the Z direction) may be excluded since the vehicle is stable in this axis. The heave may have an impact on the vessel model since the added mass and frictional resistance are impacted by the draft. For simplicity, one may assume that these values are constant for a given sea state and amount of cargo carried. The adaptive controller will naturally determine the time-averaged effect heave has on added mass and drag.

The pitch and roll of the vessel may be excluded for the similar reasons. However, it requires the additional assumption that the thrusters will remain fully submerged, which is not always true. Excessive pitch motions have resulted in the thrusters exiting the water, sending water vertically, thereby reducing their effect on surge of the vessel. This issue is minimized by avoiding particular Heron vehicles and operating in conditions with smaller waves.

<sup>1</sup> Department of Mechanical Engineering, Massachusetts Institute of Technology, Cambridge, MA 02139, USA kevin00@mit.edu

This leaves three degrees of freedom, as shown in figure 2. Since the vessel is moving and a body-fixed coordinate system is used, the system will be controlled by the first time-derivatives of each axis. This is possible since the vessel dynamics are not a function of  $X$ ,  $Y$ , or  $\psi$ , but it is a function of their time derivatives.

The number of state variables (3) is still greater than the number of actuators (2), which is a sufficient condition for proving that a system is underactuated. It is trivial to see that the  $Y$  direction is uncontrollable. However, since the drag in sway is the opposite sign of  $V_y$ , it is stabilizable to  $V_y = 0$ .

The controls problem for this includes traversing waypoints, where it is reasonable to keep the desired  $V_y$  zero.

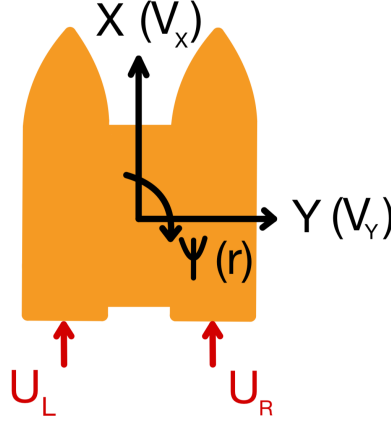


Fig. 2: Body-Fixed coordinate system demonstrating the state-space used for this project, where  $U_L$  is the left thruster input and  $U_R$  is the right thruster input. The variables in parenthesis are the first time-derivatives of each coordinate axis. For example, in this system,  $\psi$  is the yaw, and  $r$  is the yaw rate.

Infrastructure for converting desired heading and desired speed to desired  $V_x$ ,  $V_y$ , and  $r$ , has been completed by Tyler Paine []. The code infrastructure may be seen in figure 3. The same pipeline was used for this project, however a different adaptive controller was used.

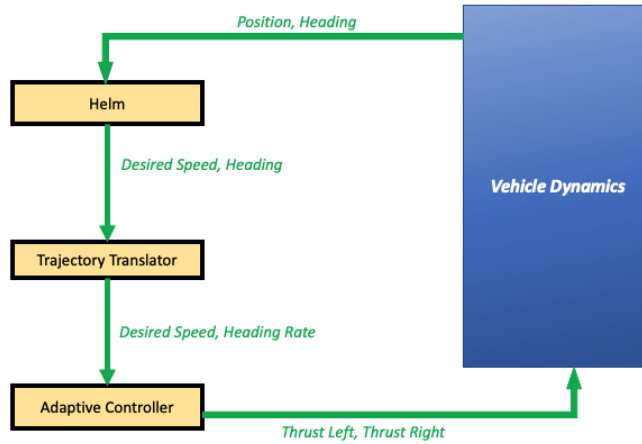


Fig. 3: Adaptive Controller Flow Diagram used other adaptive control projects at the MIT Marine Autonomy Lab.

### III. 3 DEGREE OF FREEDOM VEHICLE MODEL

A Taylor series vehicle dynamics model, as described in [1] was used as the vehicle model.

#### A. Assumptions

The assumptions made in this model are as follows:

- 1) Only first-order acceleration terms are considered.
- 2) The vessel is symmetric about the  $X$  axis.

- 3) Acceleration is not coupled to the velocity.
- 4) A 2nd order Taylor expansion about the operating condition.
- 5) Wind, waves, and current are negligible.

Assumptions 1-3 match Abkowitz (1964), as referenced in [1]. Abkowitz recommends a 3rd order Taylor expansion. This was reduced to a 2nd order expansion since the vessel has a maximum speed of 2 meters per second. Another alteration is that the model in Abkowitz used a rudder as the only control input. Instead, the thruster inputs  $U_L$  and  $U_R$  were used. Due to the low speeds, the thruster inputs were assumed to be directly proportional to the commanded percent thrust.

The resulting model is shown in (1).

$$\begin{aligned} F_x &= X_u u + X_{\dot{u}} \dot{u} + X_{ur} u |r| + X_{uu} u |u| + X_{uv} u |v| \\ F_y &= Y_v v + Y_{\dot{v}} \dot{v} + Y_{vr} v |r| + Y_{vu} v |u| + Y_{vv} v |v| \\ M_z &= N_r r + N_{\dot{r}} \dot{r} + N_{rr} r |r| + N_{ru} r |u| + N_{rv} r |v| \end{aligned} \quad (1)$$

Breaking equations in (1) into matrix form and solving for the acceleration gives (2).

$$\begin{aligned} Y_{Full} &= \begin{pmatrix} u |u| & u & u |v| & u |r| \\ v |v| & v & v |u| & v |r| \\ r |r| & r & r |u| & r |v| \end{pmatrix} A_{Full} = \begin{pmatrix} \frac{X_{uu}}{X_{\dot{u}}} & \frac{X_u}{X_{\dot{u}}} & \frac{X_{uv}}{X_{\dot{u}}} & \frac{X_{ur}}{X_{\dot{u}}} \\ \frac{Y_{vv}}{Y_{\dot{v}}} & \frac{Y_v}{Y_{\dot{v}}} & \frac{Y_{vu}}{Y_{\dot{v}}} & \frac{Y_{vr}}{Y_{\dot{v}}} \\ \frac{N_{rr}}{N_{\dot{r}}} & \frac{N_r}{N_{\dot{r}}} & \frac{N_{ru}}{N_{\dot{r}}} & \frac{N_{rv}}{N_{\dot{r}}} \end{pmatrix} \\ \begin{pmatrix} \dot{u} \\ \dot{v} \\ \dot{r} \end{pmatrix} &= Y_{Full} * A_{Full}^T + \frac{F}{acceleration} \end{aligned} \quad (2)$$

Since wind, waves, and current are assumed negligible, the only external forces are the control inputs. Including these control inputs results in equation (3).

$$\begin{pmatrix} \dot{u} \\ \dot{v} \\ \dot{r} \end{pmatrix} = Y_{Full} * A_{Full}^T + \begin{pmatrix} \frac{1}{X_{\dot{u}}} & \frac{1}{X_{\dot{u}}} \\ 0 & 0 \\ \frac{beam}{2*N_{\dot{r}}} & -\frac{beam}{2*N_{\dot{r}}} \end{pmatrix} * \begin{pmatrix} U_L \\ U_R \end{pmatrix} \quad (3)$$

The "B" matrix is clearly not full rank, and the control inputs have no impact on  $\dot{v}$ , as expected. For this reason, we can use the pseudo-inverse of the control mapping matrix to reduce the prediction in the model, since we only care about  $u$  and  $r$ . Therefore, our controller predicts the model shown in (4).

$$\begin{aligned} Y_{Controller} &= \begin{pmatrix} u |u| & u & u |v| & u |r| \\ r |r| & r & r |u| & r |v| \end{pmatrix} \\ A_{Controller} &= \begin{pmatrix} \frac{X_{uu}}{X_{\dot{u}}} & \frac{X_u}{X_{\dot{u}}} & \frac{X_{uv}}{X_{\dot{u}}} & \frac{X_{ur}}{X_{\dot{u}}} \\ \frac{N_{rr}}{N_{\dot{r}}} & \frac{N_r}{N_{\dot{r}}} & \frac{N_{ru}}{N_{\dot{r}}} & \frac{N_{rv}}{N_{\dot{r}}} \end{pmatrix} \end{aligned} \quad (4)$$

#### IV. CONTROLLER DESIGN

##### A. PD Controller

By cancelling the dynamics with the output and applying a PD controller, the end results is the following control law shown in equation (5).

$$\begin{aligned} \begin{pmatrix} U_L \\ U_R \end{pmatrix} &= \begin{pmatrix} X_{\dot{u}} & 0 & \frac{2}{beam} * N_{\dot{r}} \\ X_{\dot{u}} & 0 & -\frac{2}{beam} * N_{\dot{r}} \end{pmatrix} * (Y_{Full} * A_{Full} - k * s) \\ s &= \begin{pmatrix} \dot{u} - \dot{u}_d + \lambda_u * (u - u_d) \\ \dot{r} - \dot{r}_d + \lambda_r * (r - r_d) \end{pmatrix} \\ k &= \begin{pmatrix} k_u & 0 \\ 0 & k_r \end{pmatrix} \\ \dot{\hat{a}} &= -\Gamma * Y^T * B^{-1} * s \end{aligned} \quad (5)$$

To simplify the model and abstract away  $X_{\dot{u}}$  or  $\frac{2}{beam} * N_{\dot{r}}$ , control law can be simplified to (6). Equation (6) is the implemented control law.



Fig. 4: The *Clearpath Robotics Heron* USVs are used for testing these controllers on the water.

$$\begin{aligned}
 \begin{pmatrix} U_L \\ U_R \end{pmatrix} &= X_{\dot{u}} * \begin{pmatrix} 1 & 0 & \frac{2}{beam} * \frac{N_{\dot{r}}}{X_{\dot{u}}} \\ 1 & 0 & -\frac{2}{beam} * \frac{N_{\dot{r}}}{X_{\dot{u}}} \end{pmatrix} * (Y_{Full} * A_{Full} - k * s) \\
 s &= \begin{pmatrix} \dot{u} - \dot{u}_d + \lambda_u * (u - u_d) \\ \dot{r} - \dot{r}_d + \lambda_r * (r - r_d) \end{pmatrix} \\
 k &= \begin{pmatrix} k_u & 0 \\ 0 & k_r \end{pmatrix} \\
 \dot{\hat{a}} &= -\Gamma * Y^T * B^{-1} * s
 \end{aligned} \tag{6}$$

From there,  $X_{\dot{u}}$  could be abstracted away into the  $A_{Controller}$  and  $k$  matrices. This is acceptable since  $A_{Controller}$  is adaptive, and  $k$  is tuned by the user. This results in one parameter,  $\frac{2*N_{\dot{r}}}{beam*X_{\dot{u}}}$  requiring user intervention. This value was set to 100, since after the controller constants ( $k$  and  $\lambda$ ) were tuned for this value, it would not matter.

#### B. Sliding Mode controller

The sliding mode controller, as implemented, uses a similar control law. Sliding mode controllers typically use the control law  $u = \hat{u} - (F + \eta) * sgn(s)$  where  $\eta$  is some positive value and  $F$  is the maximum error in the model. To simplify the implementation,  $F$  was kept constant and  $F + \eta$  was replaced by the matrix  $k$ . For improved robustness,  $sgn(s)$  was replaced with a saturation function and a boundary layer. The different  $s$  variables are shown below:

$$\begin{aligned}
 s_{sat,i} &= sat(s_i \phi_i) \\
 s_{\Delta i} &= s_i - \phi_i * s_{sat,i}
 \end{aligned} \tag{7}$$

For the control law,  $F + \eta$  was replaced with  $k$ . The resulting sliding-mode controller uses equation (8).

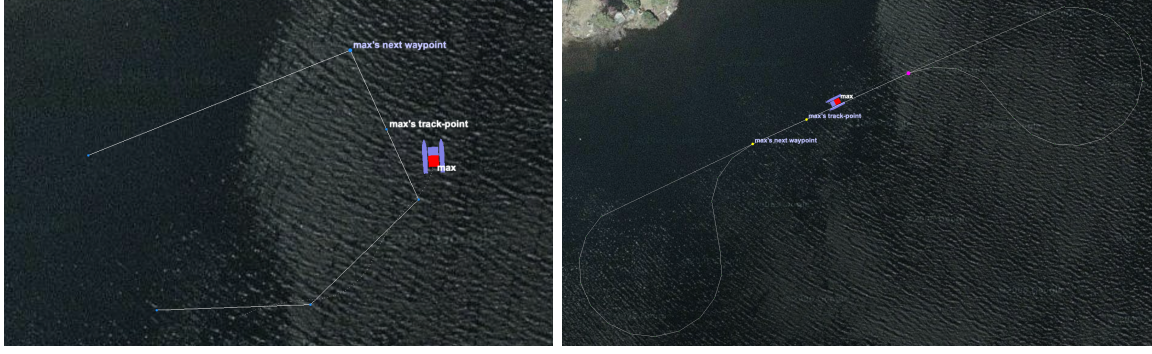
$$\begin{aligned}
 \begin{pmatrix} U_L \\ U_R \end{pmatrix} &= \begin{pmatrix} X_{\dot{u}} & 0 & \frac{2*N_{\dot{r}}}{beam} \\ X_{\dot{u}} & 0 & -\frac{2*N_{\dot{r}}}{beam} \end{pmatrix} * (Y_{Controller} * A_{Controller} - k * s_{\Delta}) \\
 \dot{\hat{a}} &= -\Gamma * Y^T * B^{-1} * s_{\Delta}
 \end{aligned} \tag{8}$$

Note the use of  $s_{\Delta}$  instead of  $s$  in this adaptation law.

### V. THE ROLE OF SIMULATION

Simulations were used to debug the controller and its supporting software. However, the simulator in question is does not have an underlying vehicle model. Rather, it attempts to visually mimic the vehicle movement as a means of testing higher-level autonomy instead of low-level controls. Therefore, simulation results were not included in this paper.





(a) The homeplate pattern.

(b) Legrun pattern.

Fig. 5: The two waypoint patterns used in the experiments.

## VI. EXPERIMENTAL SETUP

1) *Waypoints and Location*: Fear of collisions with crew shells and sailboats reduced the amount of on-water data. Therefore, adaptation vectors and boundary layers were manually adjusted during the experiment to enable quick, but stable convergence.

A vehicle was given one of two sets of waypoints to follow, as shown in Figure 5.

All tests were conducted at the "Battelle Autonomy Lab" on the Charles River.

2) *Controller Initial Conditions*: Before the adaptive controllers were run, the non-adaptive controllers underwent simple tuning. The following were kept for all experiments:

$$\begin{aligned}\lambda_u &= 5, \lambda_r = 1 \\ k_u &= 100, k_r = 10\end{aligned}\tag{9}$$

All tests initialized  $\hat{\mathbf{a}}(t=0) = \mathbf{0}$ . Additionally, the controller operated at a frequency of 10 Hertz for all experiments. To obtain the acceleration values, a low-pass filter was used, such that:

$$\begin{aligned}\dot{u}_t &= (1 - \alpha_u) * \dot{u}_{t-1} + \alpha_u * (u_t - u_{t-1}) \\ \dot{r}_t &= (1 - \alpha_r) * \dot{r}_{t-1} + \alpha_r * (r_t - r_{t-1})\end{aligned}\tag{10}$$

A value of  $\alpha_u = \alpha_r = 0.2$  produced good results, and underwent no further tuning.

## VII. RESULTS

### A. Adaptive PD Controller, Homeplate Pattern

1) *Homeplate pattern*: The results shown in Figure 6 has the following  $\Gamma$  values:

$\Gamma = 0.01 * I$  for  $0 < t < 170$  seconds  $\Gamma = 0.1 * I$  for  $170 < t$  seconds. This increase occurred since the parameter vector was taking a large time to converge.

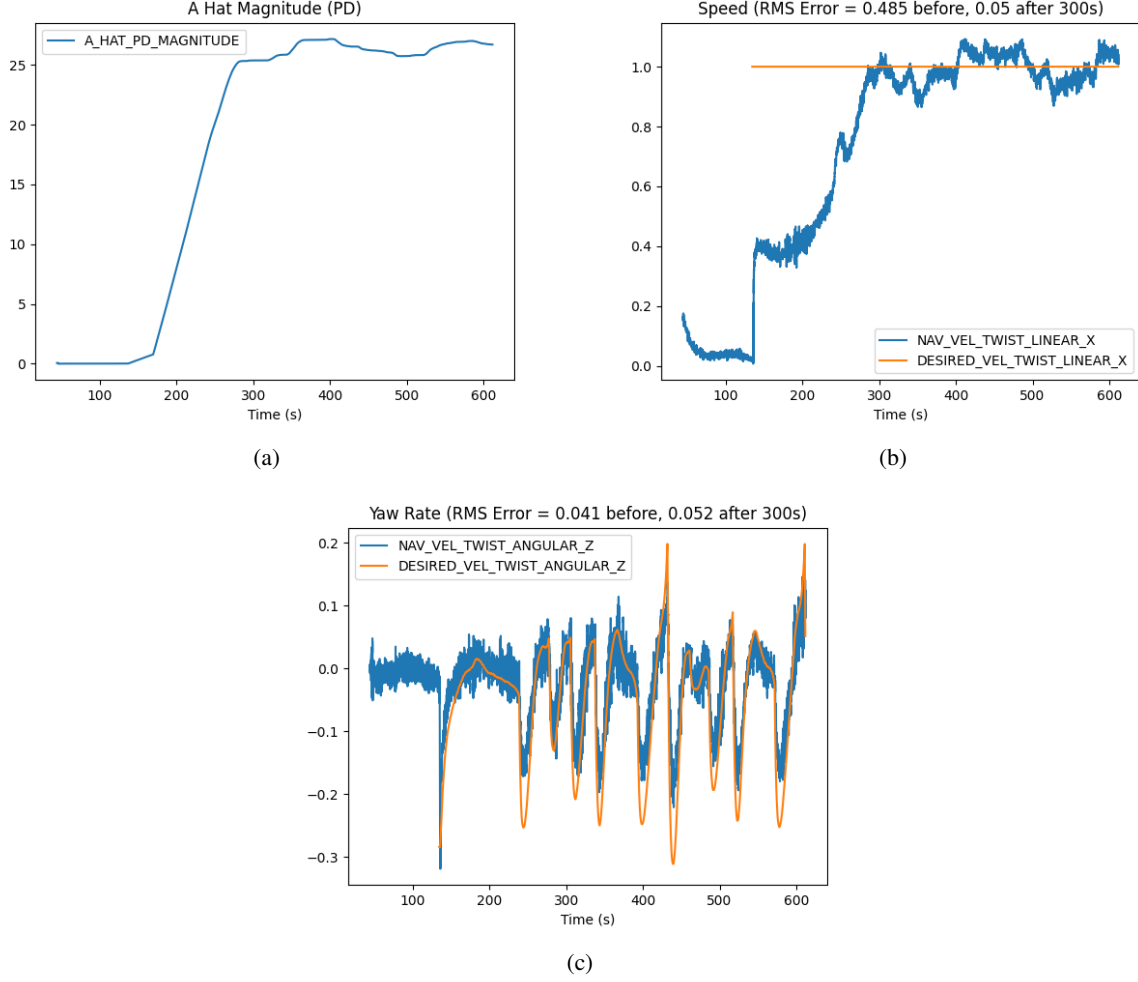


Fig. 6: Plotting the magnitude of the adaptation vector,  $\hat{a}$  (6a), speed in surge (6b), and yaw rate (6c) over time running the PD controller on the homeplate pattern. All units are in meters per second or radians per second.

### B. Adaptive Sliding Mode Controller, Homeplate Pattern

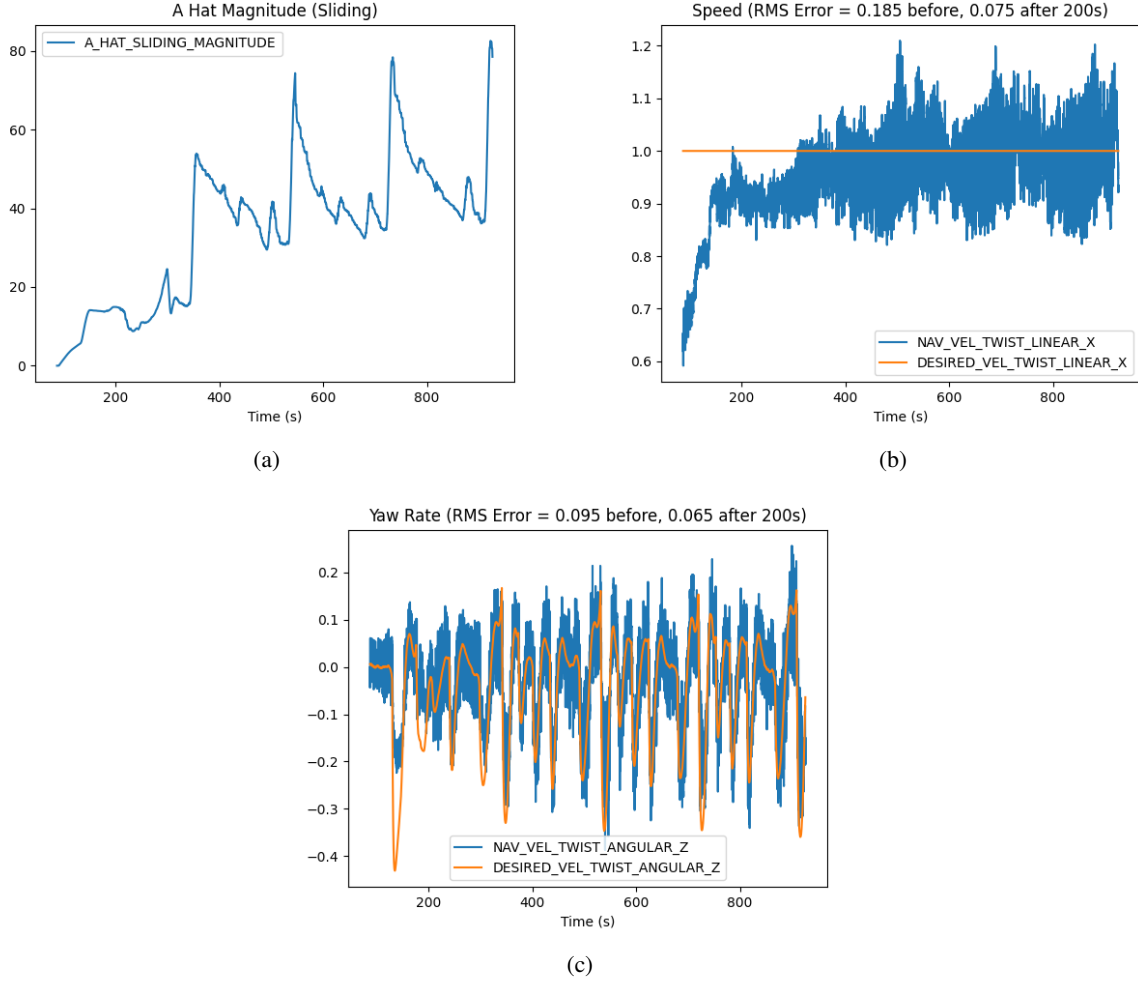


Fig. 7: Sliding mode controller running the homeplate pattern. Plotting the magnitude of the adaptation vector,  $\hat{a}$  (7a), speed in surge (7b), and yaw rate (7c) over time. All units are in meters per second or radians per second.

The results shown in Figure 7 has the following  $\Gamma$  values:

$$\Gamma = 0.1 * I \text{ for } 0 < t < 202 \text{ seconds}$$

$$\Gamma = 1.0 * I \text{ for } 202 < t \text{ seconds}$$

The boundary layer,  $\phi$ , was:

$$\phi = [0.500, 0.020]^T \text{ for } 0 < t < 341 \text{ seconds}$$

$$\phi = [0.050, 0.002]^T \text{ for } 341 < t \text{ seconds. This increase took place due to the relatively large steady-state speed error.}$$

The adaption in these plots appears is cyclic and possibly unstable. For this reason, a longer run with  $\Gamma = 0.1 * I$  and  $\phi = [0.500, 0.020]^T$  should be conducted to determine the stability.

### C. Adaptive Sliding Mode Controller, Legrun Pattern

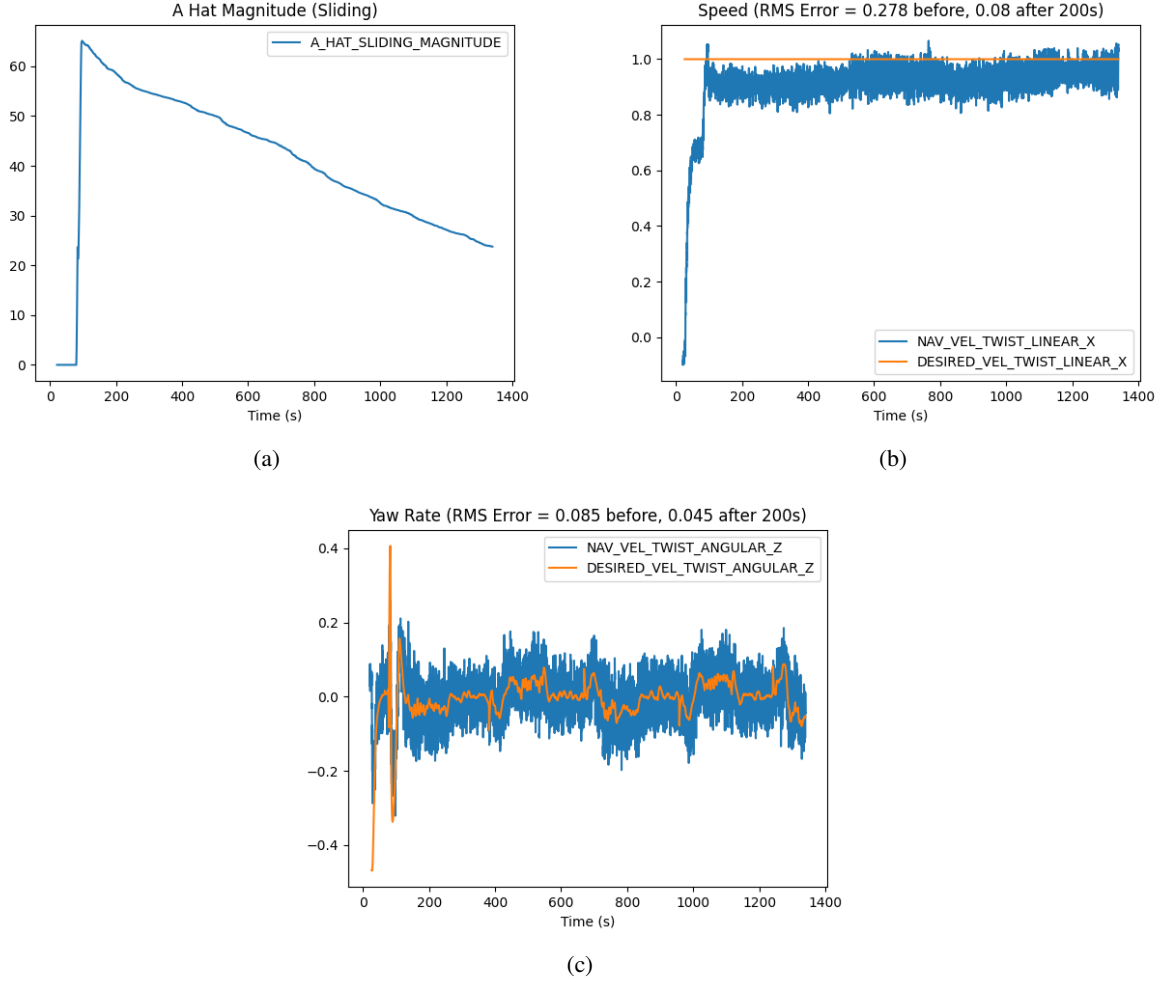


Fig. 8: Sliding mode controller running the legrun pattern. Plotting the magnitude of the adaptation vector,  $\hat{a}$  (8a), speed in surge (8b), and yaw rate (8c) over time. All units are in meters per second or radians per second.

The results shown in Figure 8 has the following  $\Gamma$  values:

$\Gamma = 0.0 * I$  for  $0 < t < 64.5$  seconds

$\Gamma = 1.0 * I$  for  $64.5 < t < 92.5$  seconds. Accidentally used a learning rate too large.

$\Gamma = 0.1 * I$  for  $93 < t$  seconds. Fixed the learning rate for a more stable controller.

The boundary layer,  $\phi$ , was kept constant at  $\phi = [0.0500, 0.002]^T$ .

$\Gamma = 0.1 * I$

## VIII. CONCLUSIONS

Both controllers saw notable improvements for matching the desired velocity, however there were minimal improvements for matching the yaw rate.

For the sliding mode controller, a boundary layer of  $\phi = [0.0500, 0.002]^T$  worked well. For both controllers, a  $\Gamma = 0.1 * I$  produced stable results which converged at an acceptably high rate.

### A. Sliding-Mode Performance

The sliding mode controller in 8b exhibited some steady-state speed error such that the speed remained at the lower end of the boundary layer. This makes sense, since adaptation stopped once the vehicle was in the boundary layer. Over time, as noise and other perturbations put the vehicle outside the boundary layer, it slowly began to integrate the last bit of speed

adaptation. However, the speed still exhibited notably more chatter than the PD controller. However, this chatter may have also been due to sensor noise chattering around the boundary layer. More tuning and testing should occur with the stable values of  $\Gamma = 0.1 * I$  and  $\phi = [0.0500, 0.002]^T$ . Adding a variable boundary layer would further improve these results.

#### B. PD Performance

The PD controller performed well. The root-mean square error in speed reduced tenfold, from 0.485 m/s before 300 seconds to 0.05 m/s after 300 seconds. The yaw rate however performed 25% worse after adaptation, increasing from 0.041 rad/s to 0.052 rad/s. This increase is due to the faster traversal speed, resulting in more frequent step inputs for the homeplate pattern's yaw rate. More frequent step inputs are expected to result in more error. To verify this claim, tests should be run where the learning rate should be initialized as:  $\Gamma = [1, 1, 1, 1, 0, 0, 0, 0]$ . Then, a full loop over the waypoints after  $\hat{a}$  converges,  $\Gamma$  should be set to  $\Gamma = I$  to see the improvement learning provides with maintaining a proper yaw rate.

#### C. Including a Drift Vector Term

In the homeplate pattern plots, shown in figures 6 and 7, it appears  $\hat{a}$  is cyclic with a period of approximately 200 seconds. This is the time for the vehicle to complete one cycle around the homeplate pattern. Cyclic behavior is expected, since the controller does not account for current or wind. Therefore,  $\hat{a}_{t=\infty}$  would be different for each heading. Including three more variables current/wind drift terms in the model would reduce the cyclic nature of the  $\hat{a}$  vector. The proposed improvement would come in the form of three variables representing an additional velocity vector and an additional yaw rate imparted on the vessel. This external velocity vector would need to be relative to a ground coordinate system.

#### D. Final Thoughts

Both controllers performed well. With an expanded model as described in VIII-C, these results would be even better. The sliding mode controller would likely also benefit from an adaptive boundary layer.

More tests would be beneficial to determine if the  $\hat{a}$  vectors converge to the same value for a given learning rate.

#### ACKNOWLEDGMENT

I would like to thank Tyler Paine for writing the overarching architecture for adaptive controllers at the lab, and for his assistance with on-water testing.

#### REFERENCES

- [1] T. Fossen, "Guidance and control of ocean vehicles," *John Wiley and Sons Ltd google schola*, vol. 2, pp. 3–27, 1994.