UNIVERSITY OF TECHNOLOGY SYDNEY

FACULTY OF ENGINEERING AND INFORMATION TECHNOLOGY

43019 – DESIGN IN MECHANICAL AND MECHATRONIC SYSTEMS

SUMMARY OF SHIP DYNAMIC THEORY AND SIMULINK MODEL DOCUMENTATION FOR AUTOMATIC MANEUVERING CAPABILITIES PROJECT

Prepared by UTS undergraduate engineering team:

Anson Antony

Charles Craig

Ho Minh Quang Ngo

Minh Quan Vu

Trung Le

Xinyi Wu

With the guidance of:

Industrial partner from the naval sector of UTS

Table of Contents

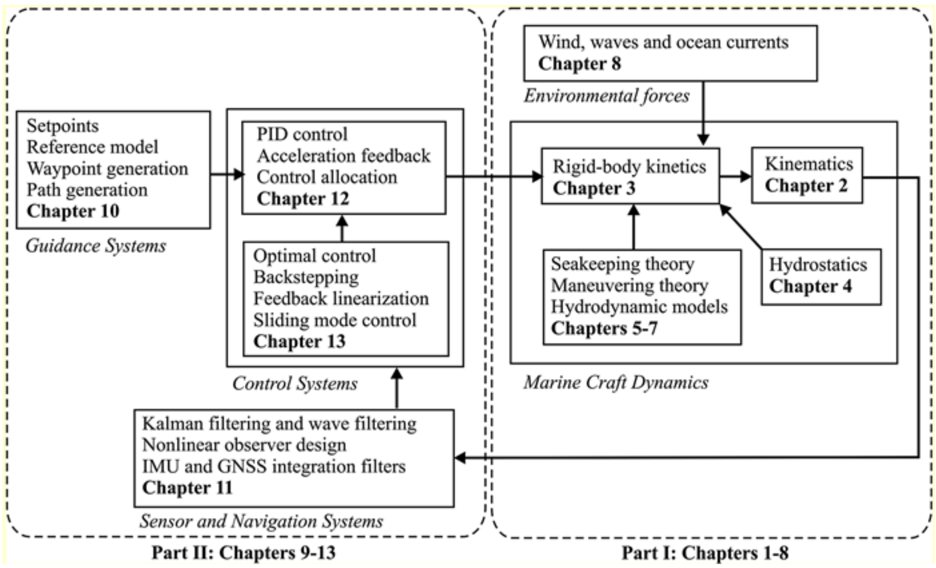
Table of Figures

1 Project Background

The project is part of the UTS Engineering subject 43019 Design for Mechanical and Mechatronic Systems. It is a project-based subject to promote engineering students' abilities to apply university knowledge, skills and professionally manage a real-world project. Spring 2022 is the first generation of students for this project. The objective of this project was to construct and test a Simulink model for a ship autopilot. The document serves an overview of the theory involved in creation of the models, the usage of our simulation, and an introduction for any team that may build upon this project in the future.

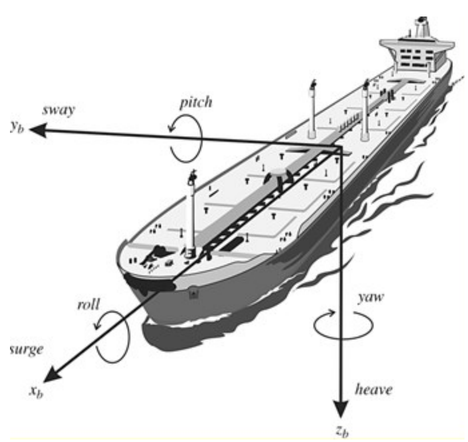
2 Nomenclature and Fundamental Theory

The Handbook outline:



2.1 Hydrodynamics

Degree of freedom and motion: refer to the Handbook of Marine Craft Hydrodynamics and Motion Control, page 6. The model currently developed is the 3 DOF model, in the surge, sway and yaw coordinates.



Classical Models in Navel Architecture:

- Maneuvering Theory: the study of a ship moving at constant speed in calm water
- Seakeeping Theory: keep the ship stationary at a specific point in the sea

Reference frame: refer to the Handbook of Marine Craft Hydrodynamics and Motion Control (Thor I. Fossen), page 19:

$$\boldsymbol{\eta} = \begin{bmatrix} \boldsymbol{p}_{b/n}^{n} \text{ (or } \boldsymbol{p}_{b/n}^{e}) \\ \boldsymbol{\Theta}_{nb} \end{bmatrix}, \quad \boldsymbol{\nu} = \begin{bmatrix} \boldsymbol{v}_{b/n}^{b} \\ \boldsymbol{\omega}_{b/n}^{b} \end{bmatrix}, \quad \boldsymbol{\tau} = \begin{bmatrix} \boldsymbol{f}_{b}^{b} \\ \boldsymbol{m}_{b}^{b} \end{bmatrix}$$

In the scope of the model developed, the distance travelled is small relative to the Earth (according to Fossen, it is the scenario of traveling within a constant latitude and longitude). Therefore, we use the flat earth navigation instead of global navigation (the position vector from NED to BODY in η is expressed in the NED coordinate)

2.2 Control

Autopilot (or automatic pilot): a device for controlling an aircraft, marine craft or other vehicles without constant human intervention (Fossen, 2011)

In the Handbook, Fossen (2011) stated that modern control systems are based on a variety of design techniques such as PID control, linear quadratic regulator (LQR) and stochastic control, H∞ control. In the actual model, we try to implement PID and potentially LQR. As specified by our client, the PID is the common design method used in the ship control industry at that time.

Motion control system: usually constructed as three independent blocks denoted as the guidance, navigation and control (GNC) systems (Fossen, 2011):

- **Guidance:** continuously compute the reference (desire) position, velocity and acceleration of a marine craft to be used by the Motion Control System
- **Navigation:** directing a craft by determining its position/attitude, course and distance travelled
- **Contro**l: Determine the necessary control forces and moments to be provided by the craft in order to satisfy a certain Control Objective

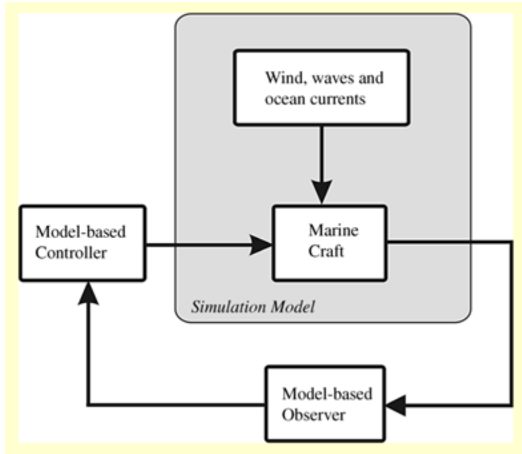


Figure 1. GNC model (Figure 1.4)

The control can be either closed-loop (feedforward and feedback) or open-loop (only feedforward):

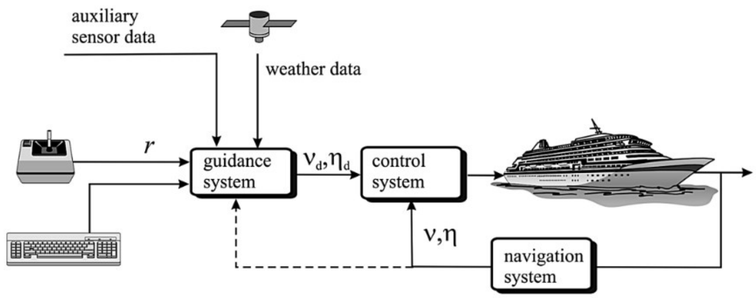


Figure 10.1 In closed-loop guidance (dotted line) the states are fed back to the guidance system while open-loop guidance only uses sensor and reference signal inputs.

Important control objectives, according to Fossen (2011):

- **Setpoint regulation:** constant input (setpoint) provided by human operation
- **Trajectory-tracking control:** The position and velocity of the marine craft are made to track desired time-varying position and velocity reference signals
- **Path-following control:** Follow a predefined path independent of time (no temporal constraint)

Weathervaning: positioning the ship at a favourable angle towards wind, waves and current (The Nautical Institute, n.d)

Dynamic Positioning: maintaining a vessel's position and heading by using its own propeller and thrusters (The Nautical Institute, n.d)

2.3 Material Guide to The Simulink Model

This section provides a guide on how the formulas on material books are applied to the model. External resources, which are research papers, articles and the client's practical insights are not mentioned in this section. Some Simulink blocks are developed in reference to some models in the Marine Systems Simulator (MSS) Matlab toolbox (Fossen, 2004). They include the Fin/Rudder Machinery and Supply Vessel (3 DOF Dynamics) blocks.

Supply Vessel block (input force τ , output position vector η and velocity vector ν): refer to section 7.1 Maneuvering Models (3 DOF) on page. 133 in the Handbook of Marine Craft Hydrodynamics and Motion Control (Thor I. Fossen)

Propeller block (input velocity vector ν, propeller angle α and propeller angular velocity RPM, ouput force τ):

- **Thrust calculation block** (input propeller angular speed RPM (round per minute) and n (round per sec), propeller diameter D and advance ratio J, output thrust T):

  $$T_p = K_T \rho n^2 D^4$$ (Pivano et al., 2007)

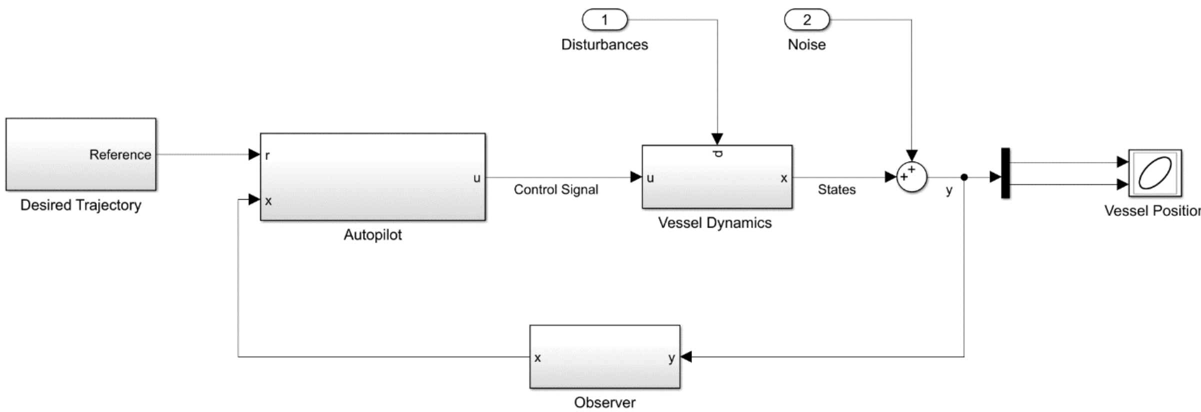
- **Advance Ratio J Calculation block** (input propeller angular speed n, distance from propeller to center axis y_p, to half vessel length x_p, velocity vector ν, output advance ratio J)

Rotation matrix in yaw blocks (input angle in NED ψ and velocity vector in BODY ν , output velocity vector in NED ν)

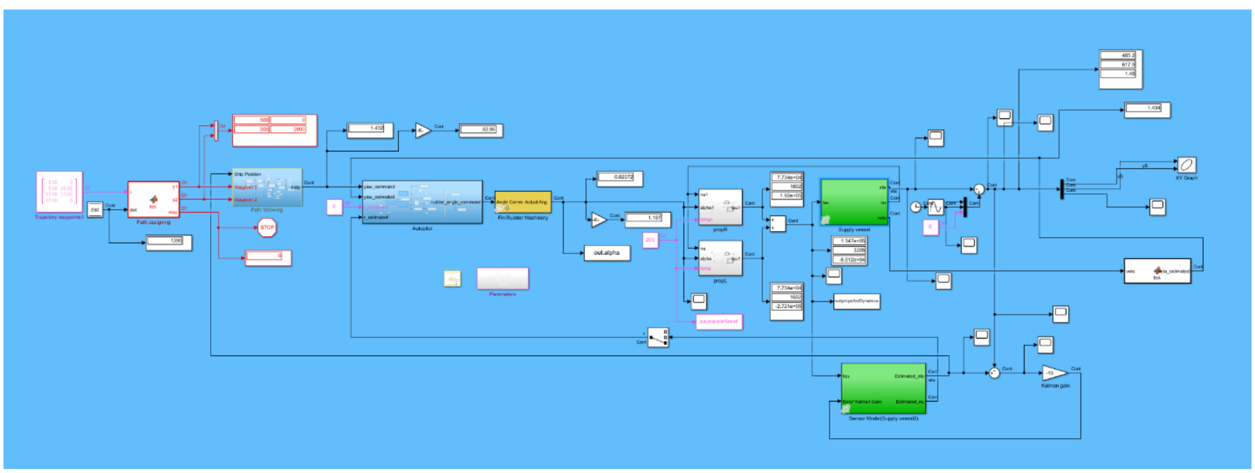
3 Implementation

3.1 Overall control architecture

The architecture of the model includes all the fundamental building blocks of a common control system:

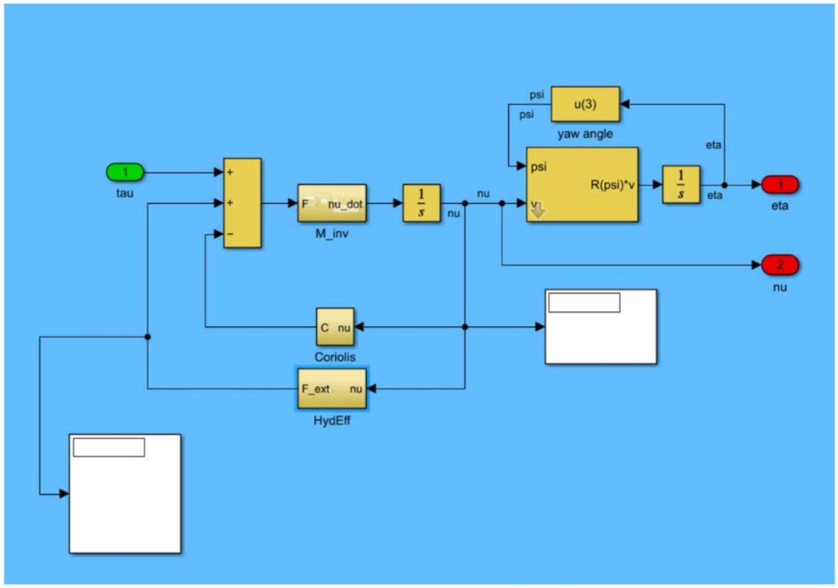


Below is the whole architecture put into practice:



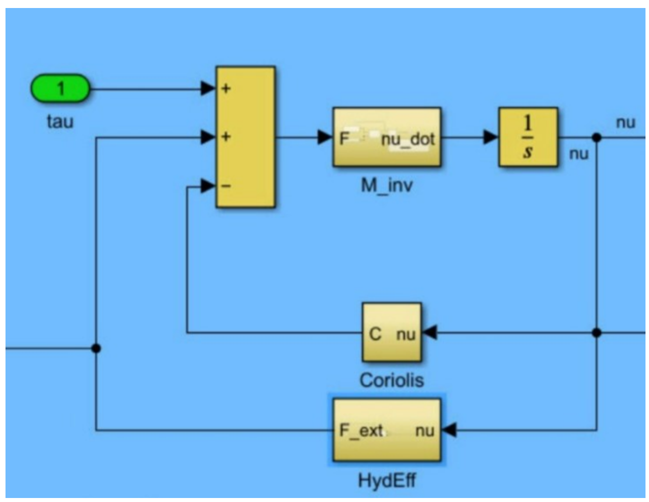
3.2 Dynamics

The submodule that was first constructed is the dynamics of vessels, which takes input force τ and outputs position vector η and velocity vector ν



3.2.1 Ship dynamics

Inside the dynamics module, this group of blocks perform calculation of the vessel dynamics.



This part takes in the propulsion force from the propellers and outputs velocity of the vessel. Here the velocity is measured in the body frame. The formular for this part can be referred to section 7.1 Maneuvering Models (3 DOF) on page. 133 in the Handbook of Marine Craft Hydrodynamics and Motion Control (Thor I. Fossen):

The 3 DOF horizontal plane models for maneuvering are based on the rigid-body kinetics:

$$\boldsymbol{M}_{RB}\dot{\boldsymbol{\nu}} + \boldsymbol{C}_{RB}(\boldsymbol{\nu})\boldsymbol{\nu} = \boldsymbol{\tau}_{RB} \tag{7.1}$$

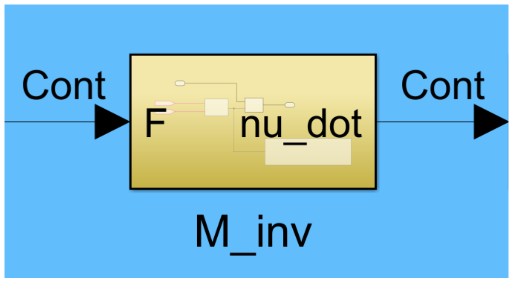
where

$$\boldsymbol{\tau}_{RB} = \boldsymbol{\tau}_{\text{hyd}} + \boldsymbol{\tau}_{\text{hs}} + \boldsymbol{\tau}_{\text{wind}} + \boldsymbol{\tau}_{\text{wave}} + \boldsymbol{\tau} \tag{7.2}$$

From this formula, simplification is advised by the client as hydrostatic force and other forces are negligible. To apply into the block model, the formula is derived:

$$M^{-1}\left[\left(\tau_{RB}+\tau_{wind}+\tau_{wave}-C_{RB}\left(\nu\right)\nu\right)\right]=\dot{\nu}$$

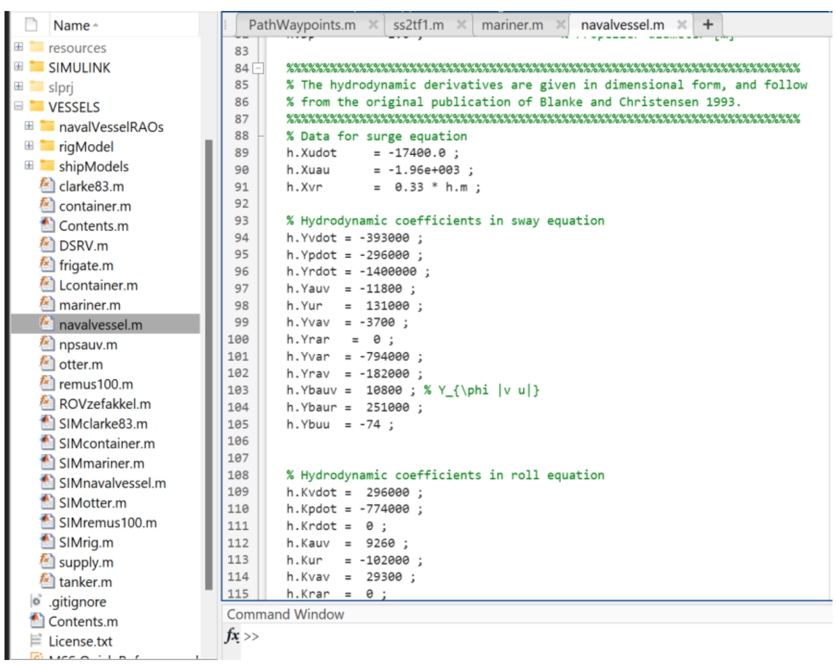
Below is the block to calculate nu_dot (derivative of velocity):



It is provided by calculating the inverse matrix of M, which can be referred to equation 7.16 on page 136 in the Handbook of Marine Craft Hydrodynamics and Motion Control (Thor I. Fossen):

$$\boldsymbol{M}=\begin{bmatrix} m-X_{\dot{u}} & 0 & 0 \\ 0 & m-Y_{\dot{v}} & mx_g-Y_{\dot{r}} \\ 0 & mx_g-Y_{\dot{r}} & I_z-N_{\dot{r}} \end{bmatrix}$$

The construction of the matrix can be partly referred to the MSS toolbox, in VESSEL/shipModels/navelvessel.m:



In the actual model, this calculation is simplified, using the parameters required in equation 7.16 above only. The value of these parameters are empirical, and is taken form the toolbox to implement into the model. Therefore, some physical specifications represent the vessel model from the toolbox only.

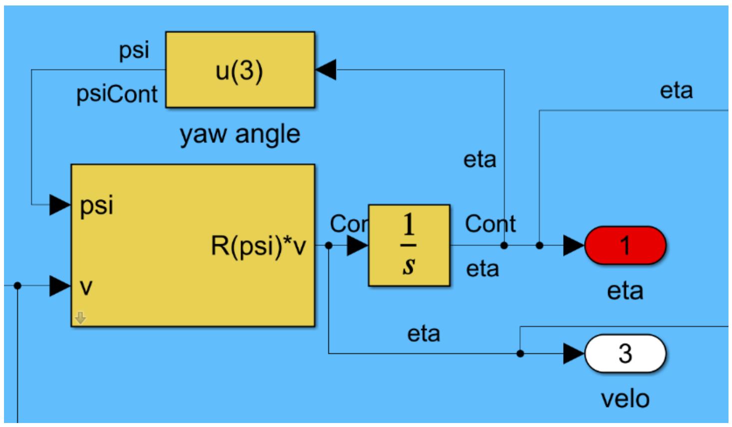
3.2.2 Coriolis effect

The Coriolis block is constructed to take account for the Coriolis force on the vessel. It is based on equation 7.13, on page 135 in the Handbook of Marine Craft Hydrodynamics and Motion Control (Thor I. Fossen):

$$\boldsymbol{C}_{RB}(\boldsymbol{\nu}) = \begin{bmatrix} 0 & 0 & -m(x_g r + v) \\ 0 & 0 & mu \\ m(x_g r + v) & -mu & 0 \end{bmatrix}$$

3.2.3 Velocity transformation

Velocity in the body frame is converted into the NED frame by this block group:



This part is based on the basic rotational matrix, with the formula

$$\dot{\eta} = R(\psi).v$$

For further information, reader can refer to section 2.2 Transformations between BODY and NED in Handbook of Marine Craft Hydrodynamics and Motion Control (Thor I. Fossen).

3.3 Pod propeller system

3.3.1 Abstract

A pod system is a propulsion system (azimuthal thruster/propulsion system) where the vessel's propeller and its electric drive motor are mounted in a pod/gondola under the vessel's stern. The horizontal orientation of this pod (around a vertical axis), thereby, can be controlled by a steering system (which should have a steering motor) mounted directly on the ship's hull.

Due to the propeller's "rotatable" feature, the propulsion supplied to the ship can be manipulated not only in magnitude but also in attitude, which integrates the propulsive function to that of the steering function of the vessel.

It is easy to recognize that a pod system eliminates the rudder that inherently exists in the traditional propulsion system. Not only that, but this propulsion system also overcomes many critical drawbacks from the traditional propeller and drive shaft systems, such as large space-consumed drive shafts, sophisticated gearing and shaft bearing systems, and inadequate quality of watertight seals in hull-damage situations.

Therefore, compared to the traditional one, the pod propulsion system has greater maneuverability which allows steering large ships in confined spaces without tugs' assistance, as well as improved dynamic positioning by applying an adequate number of pod units.

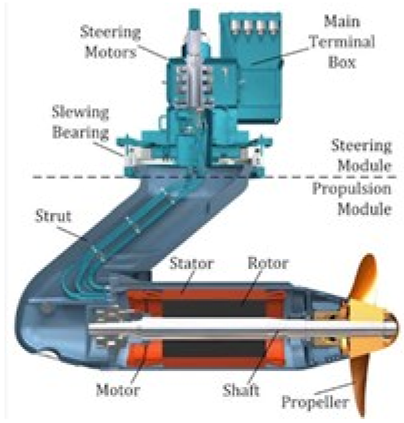


Figure 5.3.1 Typical Propulsion Pod (Navy Matters, 2021)

3.3.2 Theoretical Basis and Assumptions

Because the pod propeller has the manipulability for orientation, its control parameters are not only the speed command (RPM) like the traditional propeller in cooperation with the rudder, but also the angle command *alpha (α)*. *Alpha* is the angle between the vessel's surge axis and the propeller's shaft axis, which limits from -90° to 90° and becomes zero when the surge axis and the propeller's axis are parallel.

Based on Van Lammeren et al. (1969), we can have an approximation for the output thrust of the propeller:

$$T_p = K_T \rho \omega^2 D^4 \tag{5.3.2}$$

In which, the thrust coefficient K_T is the propeller characteristic, ρ is the water density (kg/m^3), ω is the speed (Rounds-per-second), and D is the propeller's disc diameter.

Note that the only parameter we can control to manipulate the thrust output is the speed command (ω), but the thrust coefficient K_T will also have different responses based on the current vessel's speed. Here, we can take into account the advanced number J, which is the ratio between the ambient inflow velocity of the water(u_a) to the propeller and the propeller's angular speed times the disc diameter: $J = u_a/(\omega D)$ Thrust coefficient K_T, thus, can be considered as a function of advanced number J, $K_T = f(J)$. However, we hardly determine a symbolic equation because there are too many influencing variables, so the usual way is to obtain the K_T experimentally. Therefore, a look-up table for K_T by J based on empirical records is applied in this project and can be found in Appendix.

A more practical way to get the value of thrust is through the propeller thrust/torque relationship because, in practice, the propeller torque, based on torque coefficient K_Q, is easier to calculate or measure. However, within the scope of this project, we temporarily put that aside to assume that we have a relatively precise correlation between K_T and J in order to build a dynamic model for the propeller system.

3.3.3 Simulink Model

We will go from the outermost block for the pod-propeller system to the critical inner parts.

1. General Block

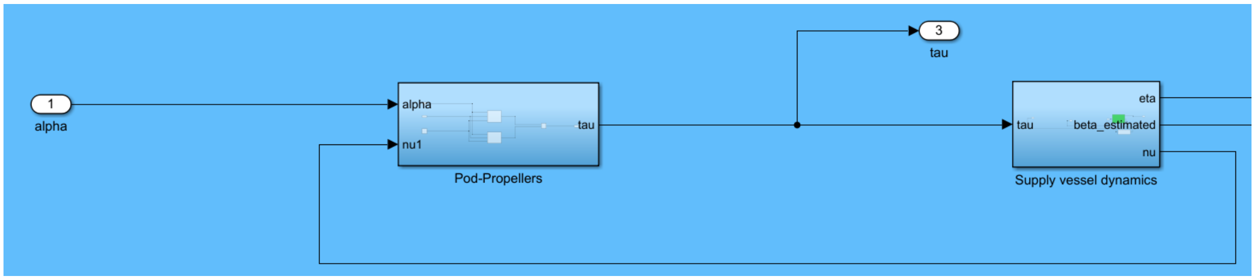


Figure 5.3.3.1. Pod-propellers system with the vessel dynamics

The pod-propellers block takes two input arguments:

- The Alpha angle, which is the angle between the propeller's axis with the vessel's x-axis.
- The Nu vector ([u, v, r]), which is the velocity vector of the vessel performed in the body frame.

Then, it outputs Tau vector, which comprises forces in x, y and torque in z of the vessel's body frame.

2. Two Propellers

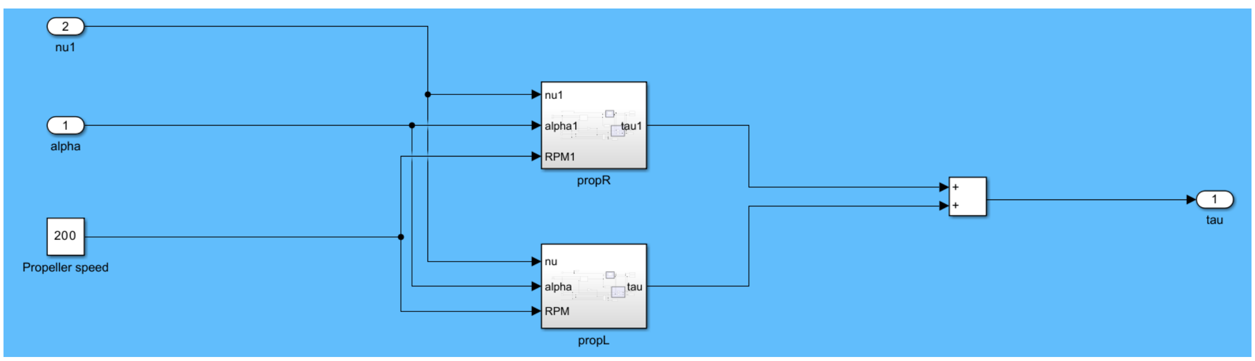


Figure 5.3.3.2. Two propellers on two sides

Inside the big Pod-propellers block, there are two identical propeller blocks denoted *propR* and *propL* for the propeller on the right and left. The Tau vector is generated by taking the sum of the two Tau components from each propeller. Here, we input the desired speed for each propeller, which is a constant of 200 RPM.

3. Single Propeller

Below is the structure inside a single propeller block. Two output components of this structure are highlighted in purple, which are the forces in x, y and torque in z of the body frame. They combine to produce the Tau component (from a single propeller) for the vessel.

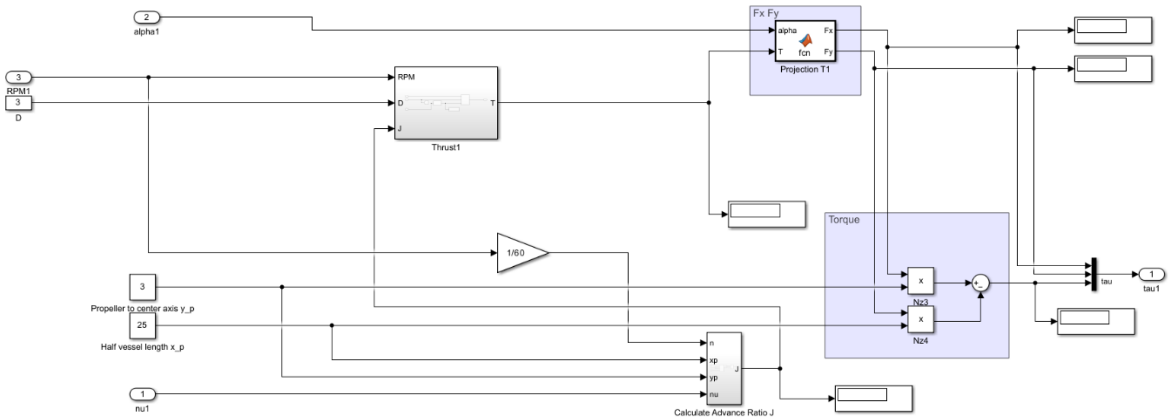
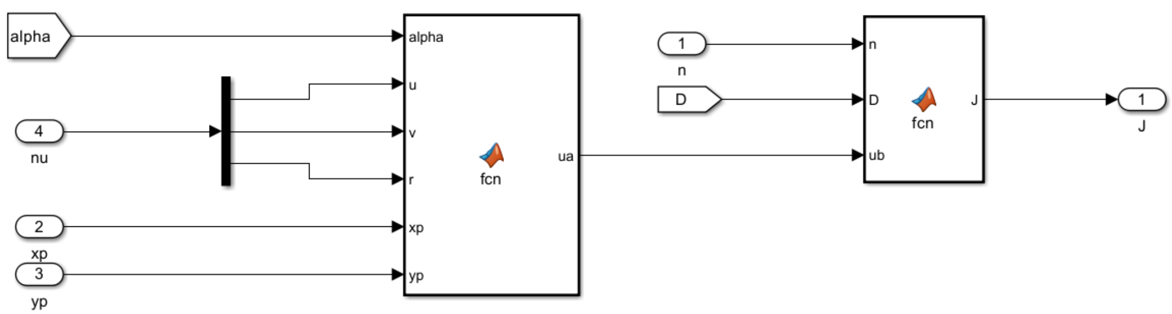


Figure 5.3.3.3. Single Propeller structure

The input arguments required for this structure are the *approximate half-length* of the vessel and the *distance from the propeller position to the ship's centre axis* (coincides with the surge axis). The half-length is an approximation of the distance from the propeller to the vessel's Center of Gravity. These parameters are required for geometrical calculations, such as projecting the thrust and velocity vector into different axes, and calculating the advance ratio J.

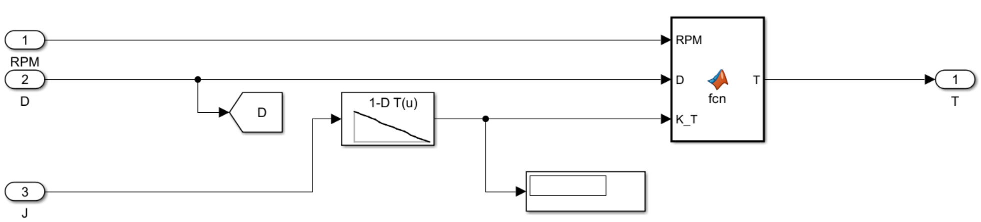
Advance Ratio



We have two blocks for calculating the advance ratio J:

- The first is to calculate the current velocity of the propeller (Cartesian velocity vector) with respect to the vessel's CoG velocity vector. Then, projecting this velocity onto the vessel's heading direction.
- The second block calculates the advance ratio J based on the formula specified in 5.3.2.

Thrust



The thrust output of the propeller is constructed from the equation 5.3.2. Here we include a look-up table for the Thrust coefficient K_T based on the value of the advance ratio J. The thrust then is projected in corresponding axes to get forces in x and y directions, and torque in z direction (of the vessel).

3.4 Autopilot system

3.4.1 Abstract

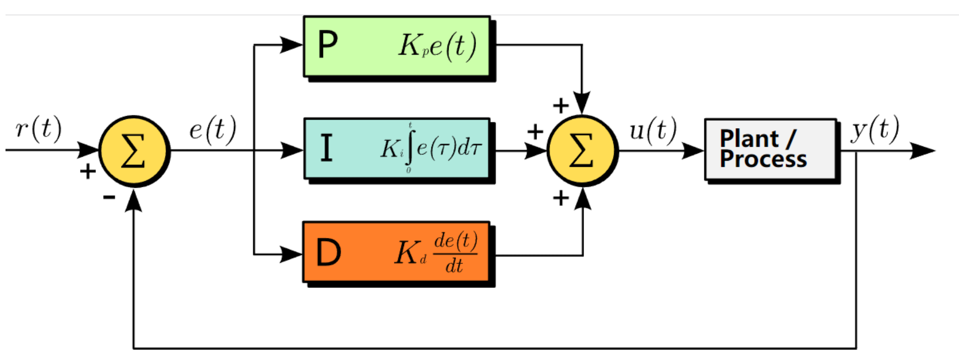
Autopilot system is a more important part in project-a naval or supply vessel, because as mentioned before, all the researches include hydrodynamics, propeller system are the advanced preparations for the autopilot system.

Autopilot is a device to control marine craft without constant human intervention. The function of this system is not only restricted to maintain the fixed course, but also execute the complicated operation like, turning and docking.

In 1922, the auto ship-steering mechanism built by Elmer Sperry successfully simulated the operation of skilled pilot, including compensating for different sea state with the feedback control and automatic gain adjustment. Then Nocholas Minorsky detailedly researched the position feedback control system and put forward three-term control law, proportional-integral-derivative (PID) control that is still be used until today. Therefore, the core of Autopilot system is PID control.

3.4.2 The basic theory of PID control

The flow of PID control algorithm simply use feedback to detect the error signal and control the controlled variables through the error signal. The basic proportional-integral-derivative flow diagram is shown below:



The input is r(t) and output is y(t), then the error can be calculated err(t)=r(t)-y(t). Finally, the PID formula can be expressed:

$$\mathrm{U}(t) = kp(err(t) + \frac{1}{T_I}\int err(t)dt + \frac{T_D derr(t)}{dt})$$

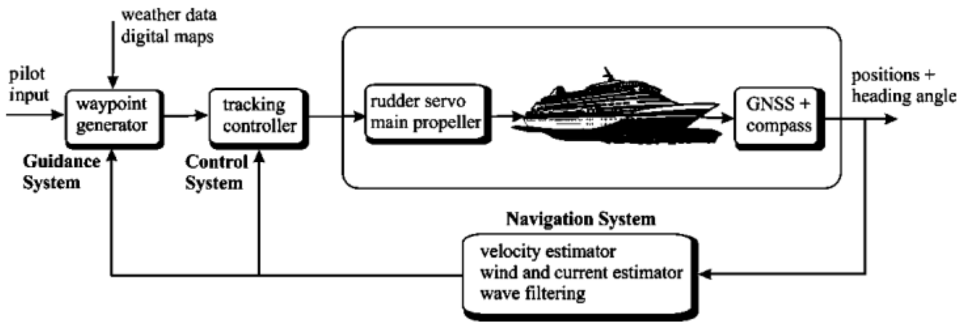
In this formula, Kp is proportional gain, Kp/Ti is integral gain and Kp*Td is derivative gain.

1. The proportional controller proportional to reflect the signal error of control system, once the error appears, proportional controller will decrease the error immediately, because the output y(t) is proportional to the input error err(t). The effect of proportional controller is not only decided by err(t), but also the value of Kp. The smaller the proportional coefficient Kp is, the smaller the control effect is and the slower the system response is. On the contrary, the larger the proportional coefficient Kp is, the stronger the control effect is, and the faster the system response is. However, the value of Kp cannot be too big that will cause the overshoot and oscillation.
2. Steady state error always exists due to the error between value of expectation and measured value, so that the integral must be added in PID control to show the variation tendency and introduce into an effective early correction signal. As usual, the the early correction signal is used to counteract one of a resistance.
3. The derivatives controller is used to control the output tendency of PI and predict the tendency in the future. The resulting damping of this output is to counteract the output of PI so that their output does not change too quickly. When the output of PI leads to the decrease of error and close to the expectation value, derivatives controller will prevent this deviation. On the contrary, if the output of PI is overcharging that leads to the increase of error, the differential will also prevent this increase. The reasonable value of Kd will prevent the overshooting.

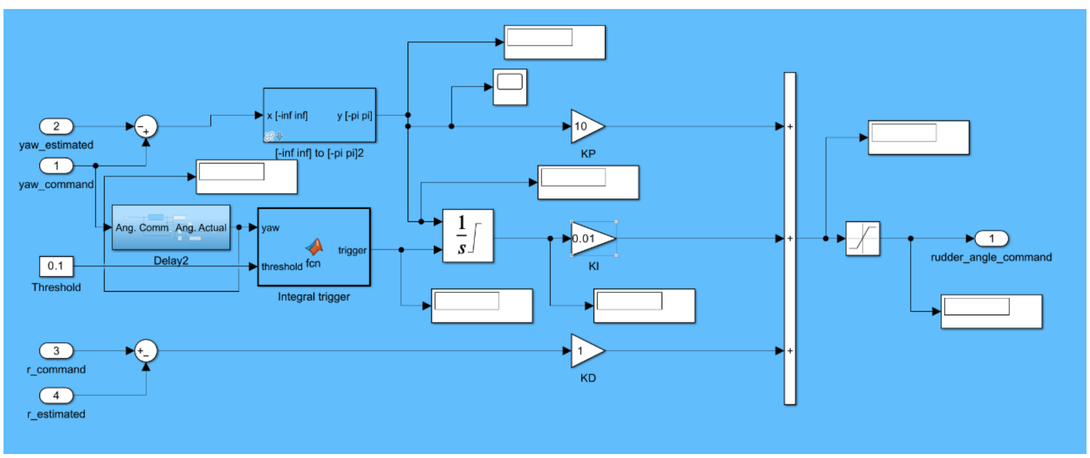
3.4.3 Heading autopilot system (PID control) apply to project

In the figure below, it shows the complete block diagram of heading autopilot. The supply or naval vessels that analysis in this part of report, five main systems and models will be discussed except the observer/wave filter which will be explained in detail in next chapter.

1. **Control system:** control system mainly represents the tracking controller in the block diagram. The function of this controller is to feedback the desired yaw angle ψd come from the reference mode. The desired yaw angle in the Simulink model is yaw_estimated.
2. **Control allocation:** This part will assign the output of control system. It means that the command comes from pilot will act on the rudder and rudder will transfer this instruction to the propeller. Output of control allocation is rudder angle or rudder command δ .
3. **Reference model:** The reference model is used to compute the yaw angle ψd and yaw rate r_d to correct the heading angle. However, the actual value of command angle will be different from the estimated value.
4. **Compass and yaw gyro:** This part is easy to understand, it is the actual yaw angle and yaw rate that use compass and yaw gyro to measure.



The structure of autopilot system is a double closed loop control system, and the Simulink model can be divided into three part, Kp, Ki and Kd. Signal which is detected by compass and yaw gyro needs to compare with the command course signal. If there is error (yaw angle), the PID controller will calculate and produce an estimated rudder angle. Then the error of rudder angle can be calculated by comparing the actual command rudder angle and estimated rudder angle. PID controller will calculate the steps above until the error between command course and actual course is zero.



3.4.4 Angle conversion

The functionality of the angle conversion block is simply to transform angle values which are out of the of -pi to pi into the values in that said range.

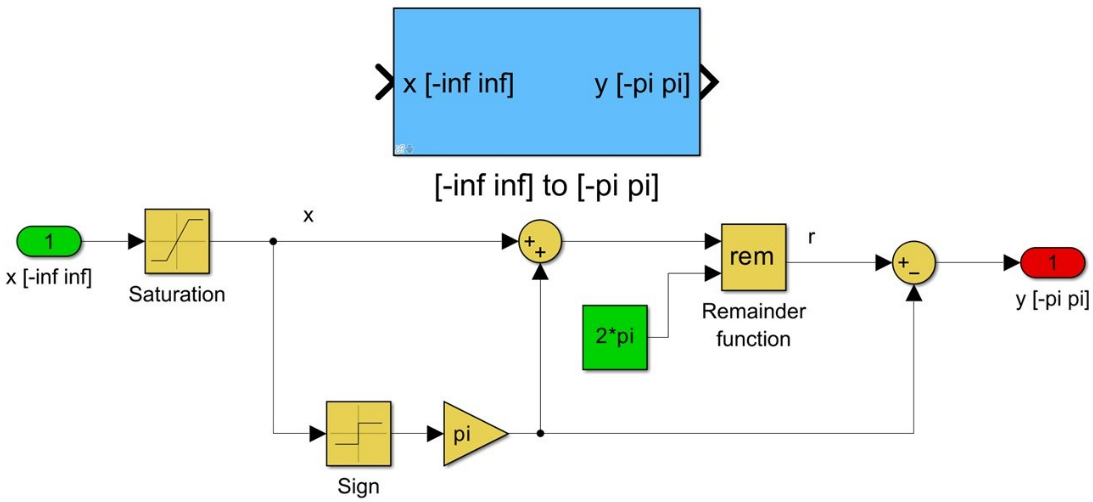


Figure 2. Angle conversion block (blue) and its inner components (below)

3.4.5 Integral trigger

3.4.5.1 Delay block

The role of the delay block is to mimic as closely as possible the real-world behaviour of signal delay from the control bridge of the ship to the actual pod-propeller's motor. Controllers should always be built to account for delay.

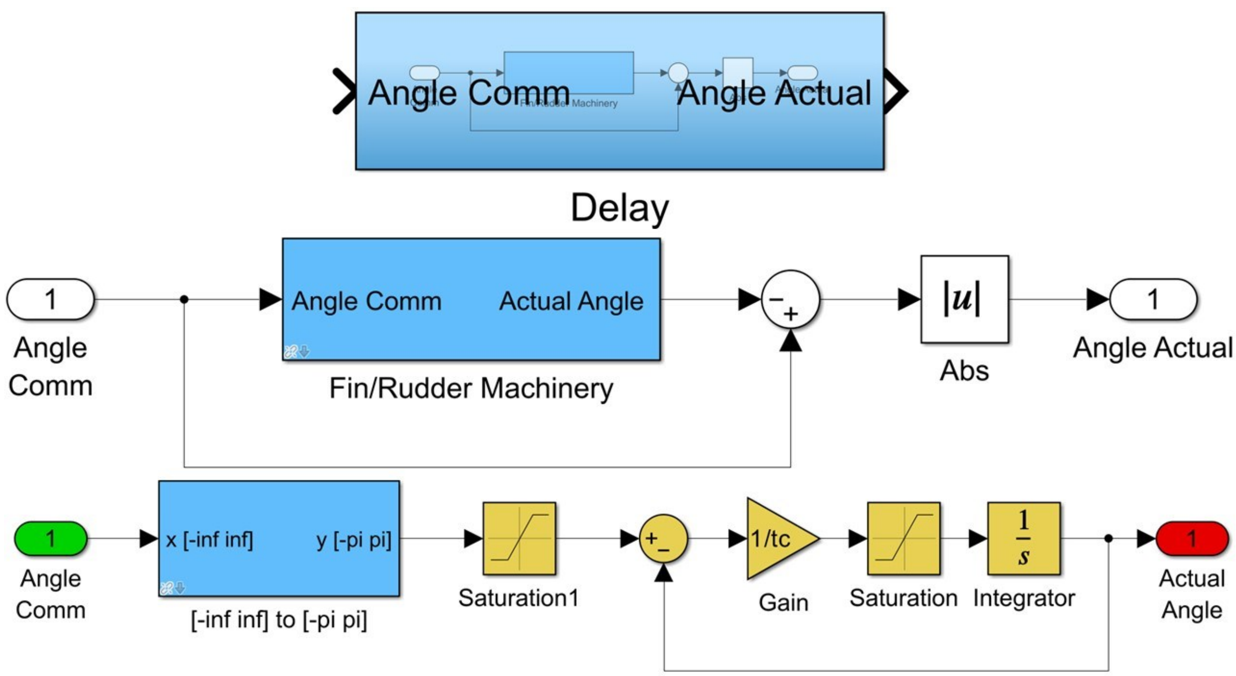


Figure 3. Delay block hierarchy

3.4.5.2 Logic block

When a command is changed, the integral part in a PID controller can accumulate excessive error that will negatively affect the responses of the system after it achieves its desired setpoints. The countermeasure method commonly employed in industries and control research is called anti-windup. Although the method we employed was different, it bore many similarities to anti-windup. We would only trigger the integral part whenever the current position is within an acceptable threshold to the desired setpoint. This action will diminish the adverse effect of the culminative error and take advantage of the integral part.


```
function trigger = fcn(yaw, threshold)

if abs(yaw) < threshold
    trigger = 0;   % Trigger Integral part of PID
else
    trigger = 1;   % Stop Integral part of PID
end

end
```

Integral trigger

3.4.6 Tuning PID gains

To tune the PID gains, we must obtain the transfer function of the ship dynamics. In this case, we firstly attempted to seek the state-space model of the vessel through linearisation. Secondly, we employed the mathematical function to transform the state-space model into the required transfer function. An alternative was to utilise the Control System Toolbox supported by Matlab to acquire the final transfer function from the input state-space model. The steps below carefully explain our process for this task.

Propeller dynamic:

$$\begin{aligned} X_1 &= X_2 = T\cos\alpha \\ Y_1 &= Y_2 = T\sin\alpha \\ N_1 &= X_1 y_P - Y_1 x_P \\ N_2 &= -X_2 y_P - Y_2 x_P \end{aligned}$$

$$\begin{aligned} X &= X_1 + X_2 = 2T\cos\alpha \\ Y &= Y_1 + Y_2 = 2T\sin\alpha \\ N &= N_1 + N_2 = -2Tx_P \sin\alpha \end{aligned}$$

Linearised hydrodynamic effect:

$$\begin{aligned} X_{hyd} &= X_{u|u|} u|u| = K \\ Y_{hyd} &= Y_{|u|v}|u|v + Y_{ur} ur = K_1 v + K_2 r \\ N_{hyd} &= N_{|u|v}|u|v + N_{|u|r}|u|r = K_3 v + K_4 r \end{aligned}$$

Linearised ship dynamic model:

$$
\begin{aligned}
M\dot{v} + Cv &= \tau + \tau_{hyd} \\
\dot{v} &= M^{-1}\left(\tau + \tau_{hyd}\right) \\
\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{r} \end{bmatrix} &= M^{-1}\left(\begin{bmatrix} X \\ Y \\ N \end{bmatrix} + \begin{bmatrix} X_{hyd} \\ Y_{hyd} \\ N_{hyd} \end{bmatrix}\right) \\
\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{r} \end{bmatrix} &= M^{-1}\left(\begin{bmatrix} 2T\cos\alpha \\ 2T\sin\alpha \\ -2Tx_P\sin\alpha \end{bmatrix} + \begin{bmatrix} K \\ K_1 v + K_2 r \\ K_3 v + K_4 r \end{bmatrix}\right)
\end{aligned}
$$

With the equilibrium points of our choice, we can utilize the small angle approximation

$$
\begin{aligned}
\cos\alpha &\approx 1 \\
\sin\alpha &\approx \alpha
\end{aligned}
$$

The linearised ship model becomes

$$
\begin{aligned}
\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{r} \end{bmatrix} &= \begin{bmatrix} M_{11}^{-1} & 0 & 0 \\ 0 & M_{22}^{-1} & M_{23}^{-1} \\ 0 & M_{32}^{-1} & M_{33}^{-1} \end{bmatrix}\left(\begin{bmatrix} 2T \\ 2T\alpha \\ -2Tx_P\alpha \end{bmatrix} + \begin{bmatrix} K \\ K_1 v + K_2 r \\ K_3 v + K_4 r \end{bmatrix}\right) \\
&= \begin{bmatrix} M_{11}^{-1}\left(2T + K\right) \\ M_{22}^{-1}\left(2T\alpha + K_1 v + K_2 r\right) + M_{23}^{-1}\left(2T\alpha + K_1 v + K_2 r\right) \\ M_{32}^{-1}\left(2T\alpha + K_1 v + K_2 r\right) + M_{33}^{-1}\left(2T\alpha + K_1 v + K_2 r\right) \end{bmatrix}
\end{aligned}
$$

For the state space model, we are only concerned with the last two row in the above matrix equation.

The state space model of the ship dynamics is

$$
\begin{aligned}
\dot{q} &= Aq + Bu \\
y &= Cq + Du
\end{aligned}
$$

The state equation can be developed into

$$
\begin{bmatrix} \dot{v} \\ \dot{r} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} M_{22}^{-1}K_1 + M_{23}^{-1}K_3 & M_{22}^{-1}K_2 + M_{23}^{-1}K_4 & 0 \\ M_{32}^{-1}K_1 + M_{33}^{-1}K_3 & M_{32}^{-1}K_2 + M_{33}^{-1}K_4 & 0 \\ 0 & 1 & 0 \end{bmatrix}\begin{bmatrix} v \\ r \\ \psi \end{bmatrix} + \begin{bmatrix} 2TM_{23}^{-1}x_P \\ 2TM_{33}^{-1}x_P \\ 0 \end{bmatrix}\alpha
$$

The output equation can be derived as

$$
y = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} v \\ r \\ \psi \end{bmatrix}
$$

Transfer function of the ship dynamics can be obtained from the transformation of the state space model

$$\frac{\psi(s)}{U(s)} = C\Phi B$$

Then code the coefficient of state space equation in

$$\Phi = (sI - A)^{-1}$$

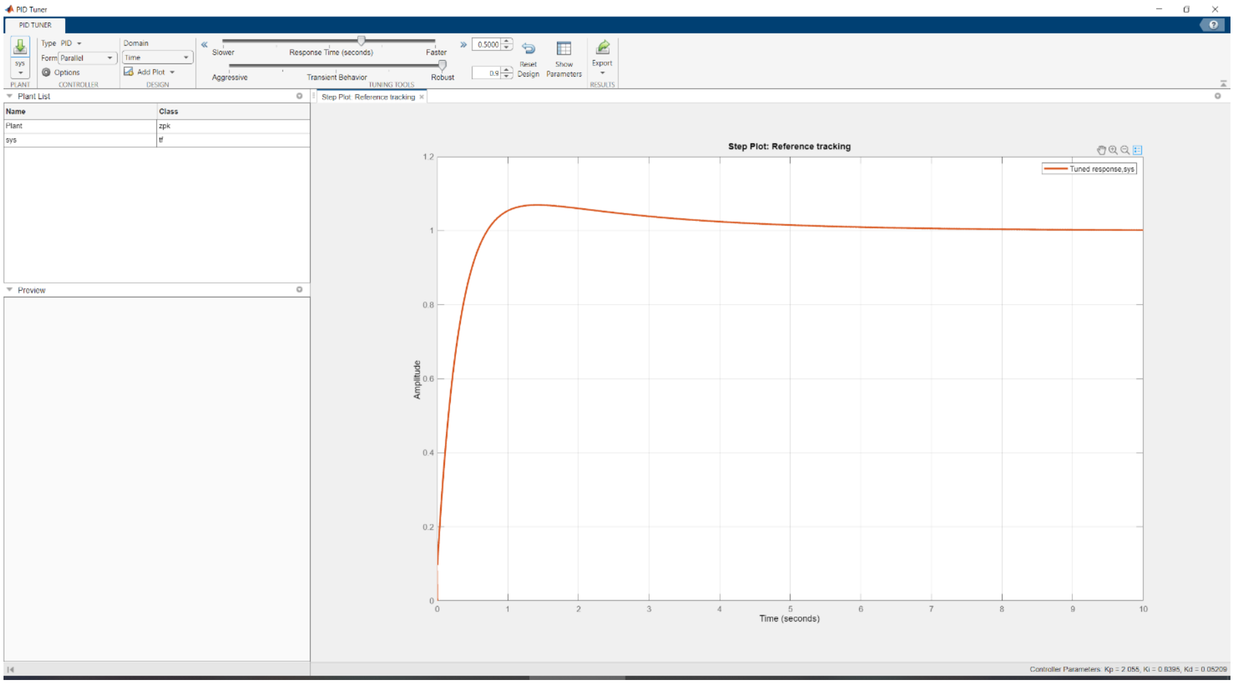
```
load statefeedback.mat
M = [25.8000000000000   0   0;
0   33.8000000000000    1.01150000000000;
0   1.01150000000000    2.76000000000000];
Minv=inv(M);
A = [Minv(2,2)*k1+Minv(2,3)*k3    Minv(2,2)*k2+Minv(2,3)*k4    0 ;
    Minv(3,2)*k1+Minv(3,3)*k3    Minv(3,2)*k2+Minv(3,3)*k4    0 ;
    0    1    0];
B = [2*T*Minv(2,3)*Xp  ;  2*T*Minv(3,3)*Xp ;  0];
C = [0 0 1];
D = [0];
[num,den] = ss2tf(A,B,C,D,1);
printsys(num, den);
```

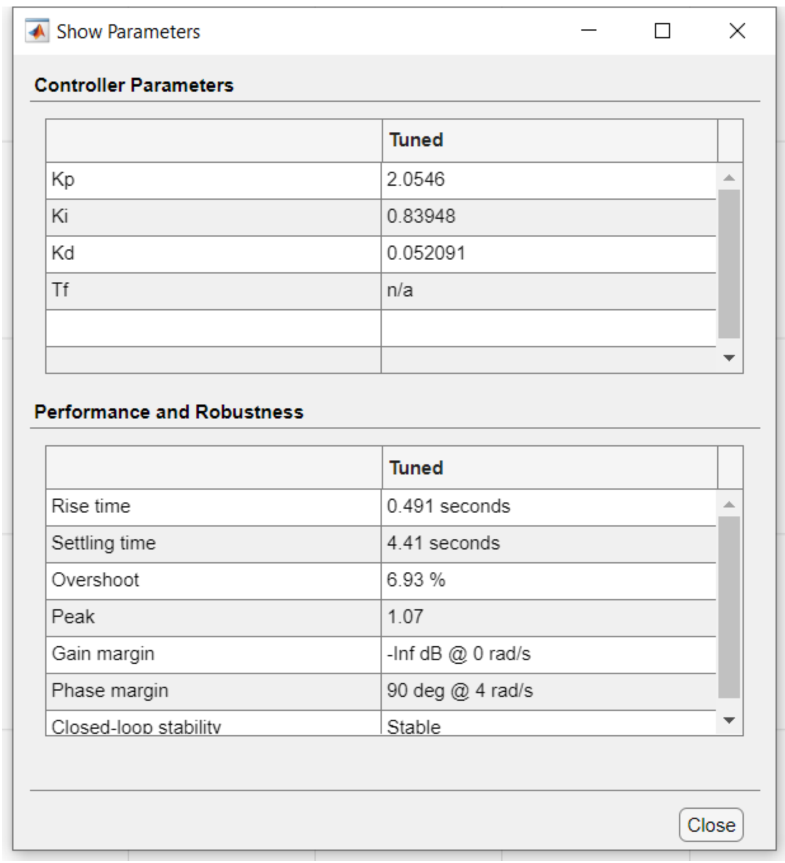
Where, Minv is automatically calculated by MATLAB, and Xp is half vessel length. Run this code, the transfer function will be shown in the command window.

```
Command Window
>> ss2tf1

num/den =

         1410395.9955 s + 4366485855.6119
   -----------------------------------------
   s^3 + 313596.8043 s^2 + 2242858502.4835 s
fx >>
```

Then create a new script to show transfer function and import it in PID tuner APP.

In the PID tuner, type and form need to be changed as PID and parallel. Moreover, the response time and transient behavior should be adjusted gradually to make PID gains reach a reasonable value.





Show Parameters

Controller Parameters

	Tuned
Kp	2.0546
Ki	0.83948
Kd	0.052091
Tf	n/a

Performance and Robustness

	Tuned
Rise time	0.491 seconds
Settling time	4.41 seconds
Overshoot	6.93 %
Peak	1.07
Gain margin	-Inf dB @ 0 rad/s
Phase margin	90 deg @ 4 rad/s
Closed-loop stability	Stable

Close

3.5 Path tracking

The path following algorithm use in this model is a type of "pure pursuit" tracking. The underlying concept behind the algorithm is elaborated in the following sections and figures. Firstly, the user inputs a set of coordinates from which a simple path is built connecting them. The path tracking block assigns the designated path to the vessel. The closest distance from the ship to its path is calculated based on the ship's current location and the coordinate formula of the path. From the projected point on the path that marks the shortest distance to the ship, a lookahead point is realized on the path based on the projected point and the direction vector of the path. The path tracking block calculates the necessary heading angle so that the ship can follow the lookahead point. When the lookahead point is beyond the second waypoint that constructed the path, a new trajectory is devised for the ship to track its next path.

3.5.1 Path generation

This block is designed to be straightforward. Manual input of the desired path is intuitive and possible as illustrated in Figure 4.

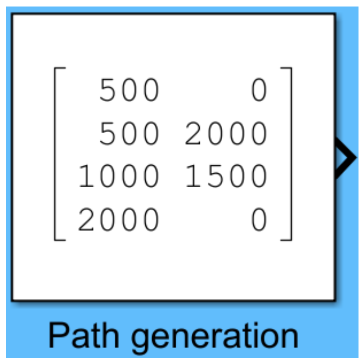


Figure 4. Path generation block

The path is generated from a matrix with multiple rows and two columns. The matrix is the combination of waypoints with their x and y coordinate in the first and second column respectively. Two consecutive waypoints mark the starting and end point of a path, from which the coordinate formula of the path can be devised.

Other than performing the laborious work directly in Simulink, the group has successfully attempted to populate the waypoints through the App or initiate them internally in callbacks when Simulink starts the initialization stage. This way still requires the waypoints to be generated by users or the results from another process. However, this method promotes further the capability of automation for the general path tracking functionality of the overall control system and limits direct manipulation from within the Simulink model.

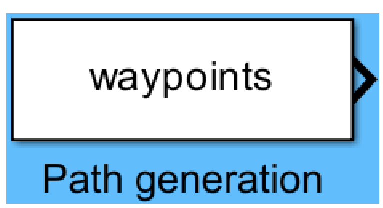


Figure 5. Simulink acquires waypoints variable from Matlab workspace

3.5.2 Assignment of path

Matlab function block in Simulink is employed to perform assignment of path for the vessel. The inputs to are 2 consecutive waypoints from the above path generation block, the distance between the ship and the second waypoint of the current path resulted from the pure pursuit path tracking block below. This block outputs 2 waypoints which form the desired path for the ship to follow and a stop signal when the ship reaches the end of the last path. The logic of this block is simple enough to derive from the overview of the internal Matlab code.

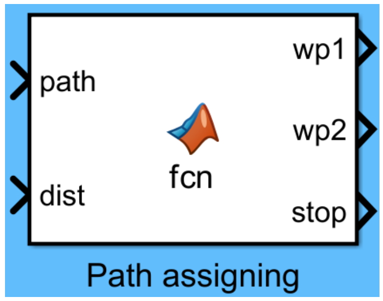
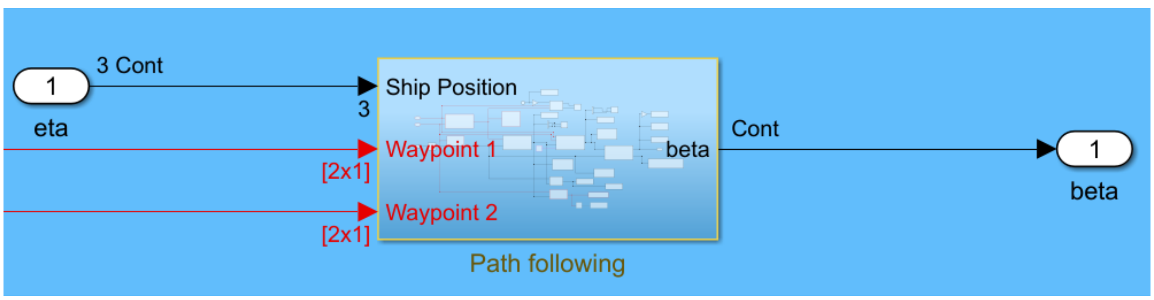


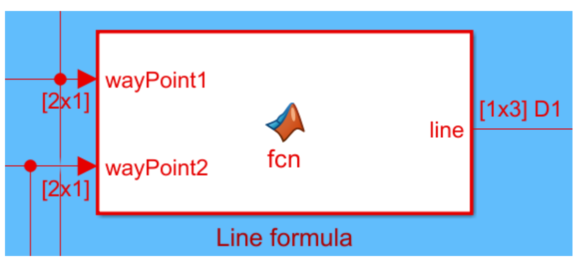
Figure 6. Path assigning block in Simulink

3.5.3 Pure pursuit path tracking

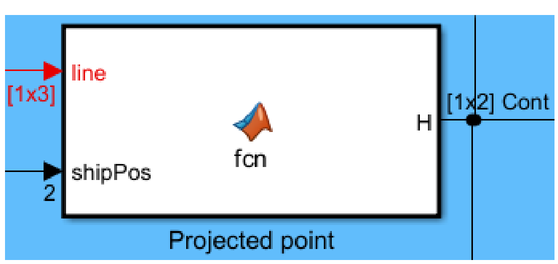
The below block is for the pure-pursuit path-tracking algorithm. Its input arguments include the current vessel's position in the NED frame and two points from the set of waypoints (selected by the Path assigning block above). After processing, it will output the Beta angle, which is the required heading (yaw) of the vessel to follow the desired input waypoints. We will go further inside this block.



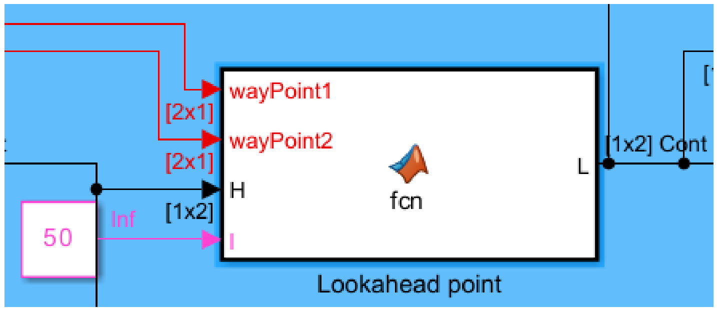
At first, the two points will be fed into a block that outputs the equation for the line passing through them (or the "path" that the vessel should follow).



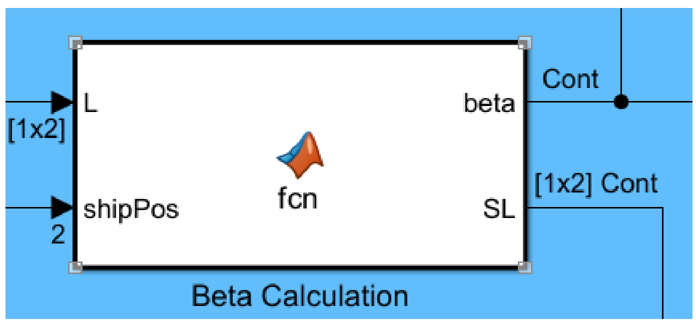
Having the constructed line equation and the current position of the vessel, the projected point from the vessel onto that line is calculated by this below block.



After then, this below block will return the lookahead point for the vessel on that line equation (path). To do so, it calculates the unit vector on the line equation using the two input waypoints, then based on the projected point and the desired lookahead distance (which is five times the vessel's length of 50m in this case) to output the result.



Based on the current position of the vessel and the lookahead point, simple trigonometric calculation is applied to get the beta angle, which is the required heading of the vessel to approach the input path. This angle will then be fed to our controller as the reference yaw angle.



3.6 Observer

In this stage, a Simulink observer model is necessary to add to the project on account of the error and noise of the ship control system. The main purpose of the observer in a control system is to provide estimations of critical state variables from the measured input and output of the system. A significant portion of the observer is similarly replicated from the vessel's dynamic model. Since the observer took the output of the physical model as input parameters, a simulated sinusoidal noise was added to that input signal to mimic actual noises in real-world sensing devices. Moreover, uncertainty coefficients were included in the duplicated hydrodynamic model to address unknown environmental conditions in the real operation of a marine vessel in the ocean. In an attempt to build the observer for the ship model, a most basic Kalman filter was endeavoured. The filter's purpose is to pursue optimal solutions in the system with noise. This can be done by mathematically seeking the weight of the filter, which is calculated by a recursive formula based on the system's initial uncertainty and noise intensity. Through intuition, random numbers were input as the weights to visualise the effect of the filter. However, no proper methods in control theory for securing good weights for the Kalman filter were conducted. So, the Kalman filter will be the ideal candidate to implement in the ship's observer for future work.

3.6.1 Simulink model of Observer

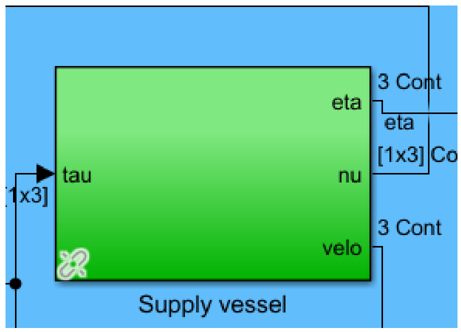
Some formulas that are used in the Observer need to be understood in order to build the model:

$$X(k) = AX(k-1) + BU(k) + w(k)$$

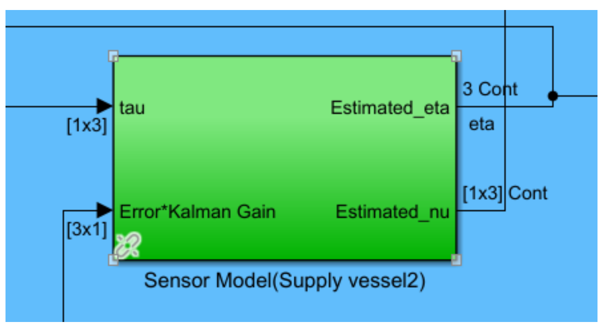
$$Z(k) = HX(k) + v(k)$$

The parameters we need to know are X(k) and Z(k). X(k) is the true value of the current system state and Z(k) is the measured value of the current system state.

In the ship model,



This is the true value of X(k) after several iterations.



This is the estimated Z(k) value that is measured by the sensor. This model is multiplied by 0.8, because there is not a sensor in this ship control model, so a mathematical model is built as a sensor.

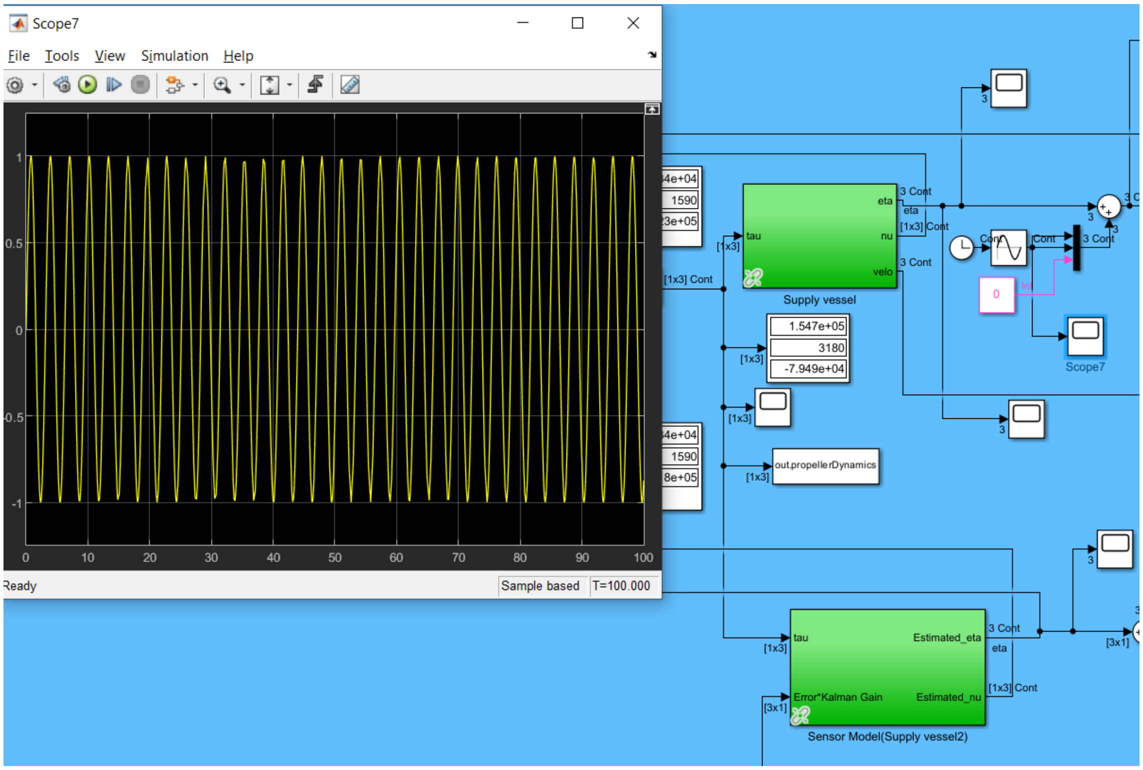
Only the mass and Izz have to be multiplied by 0.8, coefficient cannot be implemented in this step.

$$X(k|k-1) = AX(k-1|k-1) + BU(k) + w(k)$$

$$X(k|k) = X(k|k-1) + K(k)[Z(k) - HX(k|k-1)]$$

Then these two formulas are a part of the Observer equation, the first equation means the last true value add the process noise, like model or system disturbance. The second equation is to explain the final true value. K(k) in the second equation is the gain, which means if the system should believe the measure value or the true value.

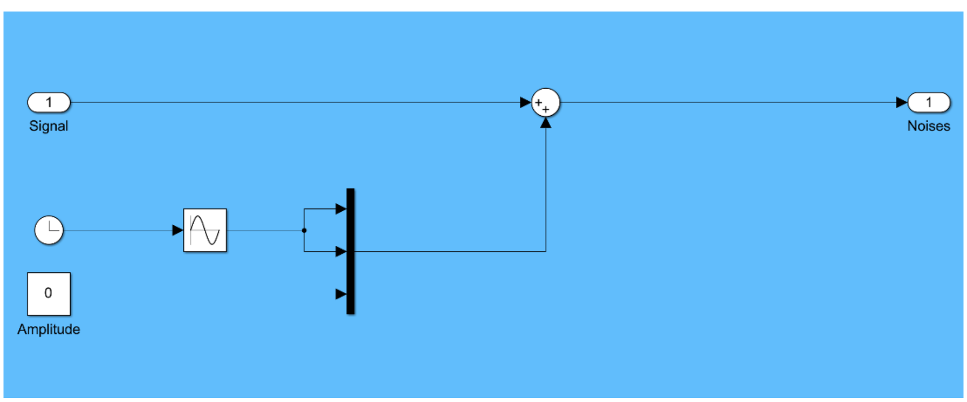
W(k) is a process noise in the system which is explained before. It is simulated as a sine wave noise.



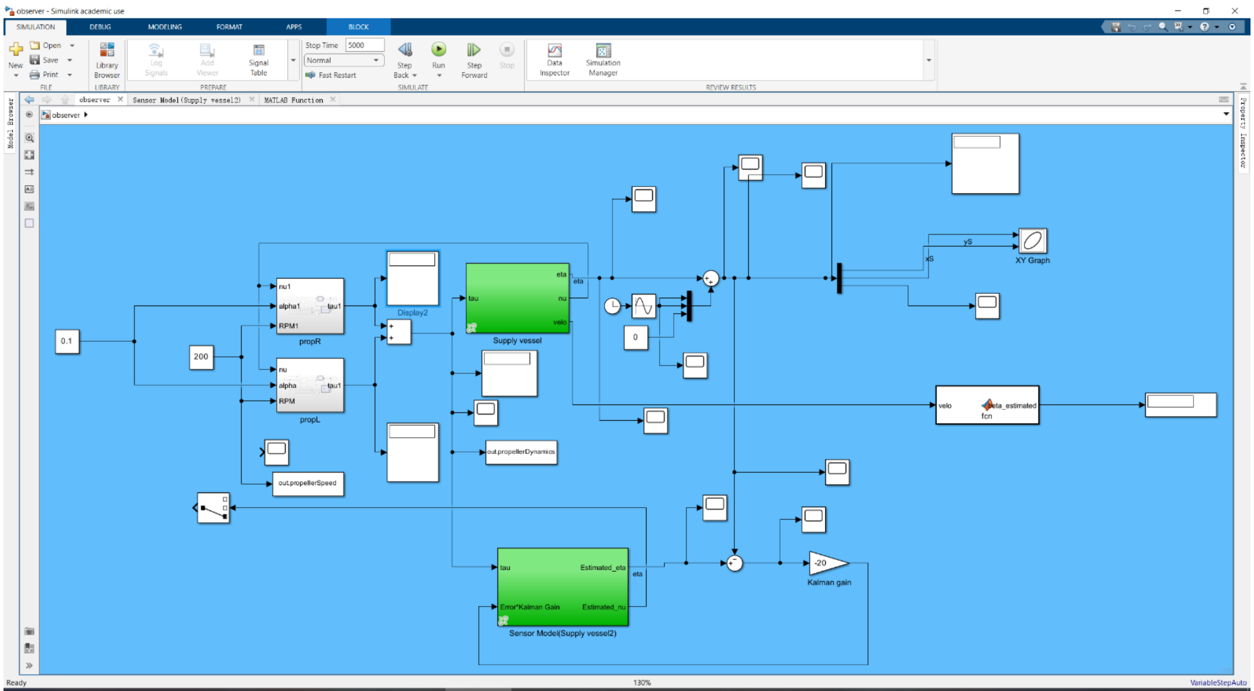
Therefore, a frame of Simulink model can be built with these formulas.

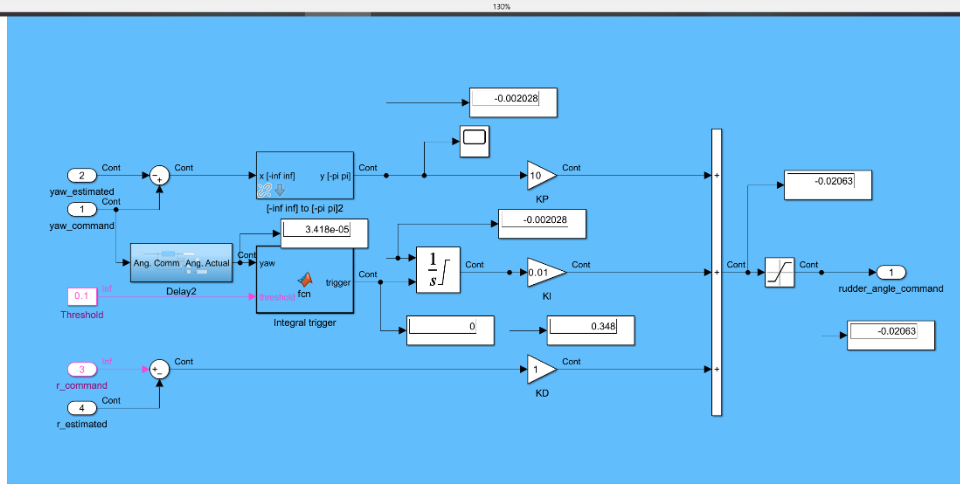
3.7 Noises and Disturbances

In real systems there will always be noise associated with sensor readings and signal transmission. These noise needs to be filtered out before further processing. The below figure simulates a noise signal introduced into the system at the Ship Dynamics block.



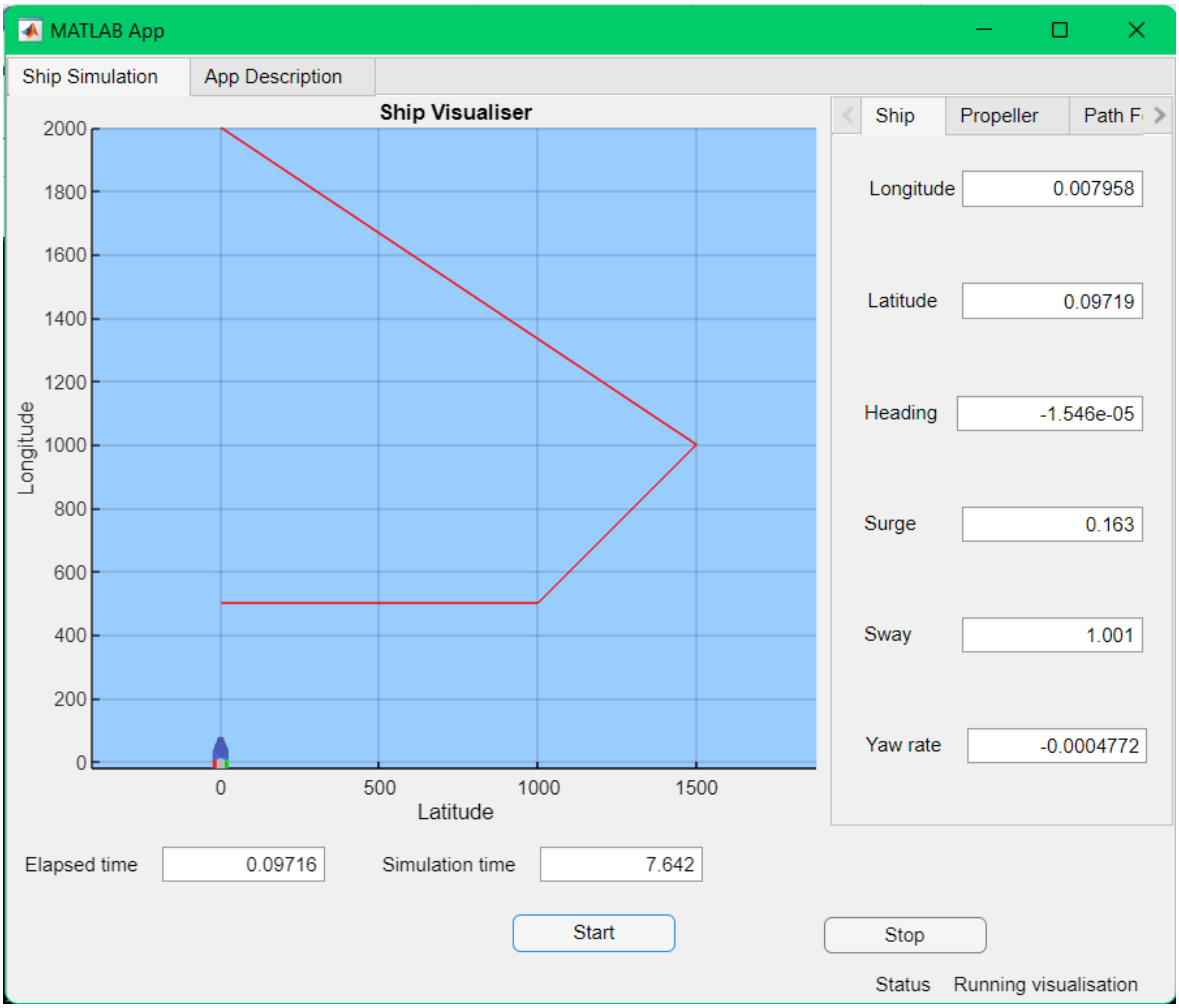
Wind, waves and currents are external forces that act on the ship and alter its state. The PID controller will adjust the propellor speed and angle based on sensor feedback. The sensors on the ship will measure the current trajectory and the autopilot can use that information to adjust the power and angle of the ropellers to stay on course.



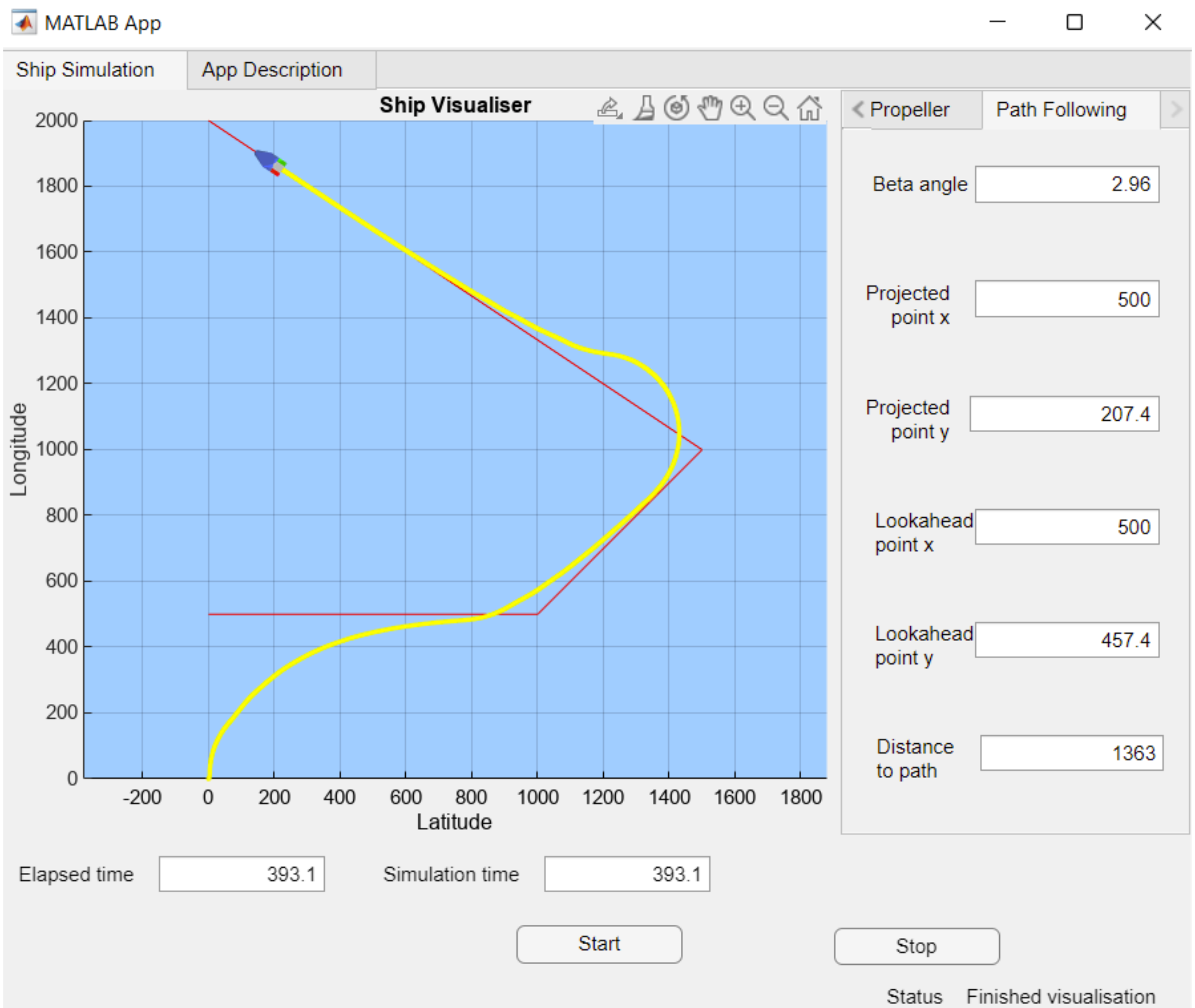


3.8 Graphical user interface

To make the simulation results more intuitive to understand, a visualizer was developed that animated the simulation results. This interface shows the ship's position, heading, and propeller angle change over time, as well as precise statistics on the sidebar. The vessel's course plan is overlayed so that the precision of path following may be visually understood. The interface is activated by running the *app1.mlapp* file.



MATLAB App
Ship Simulation
App Description
Ship Visualiser
2000
1800
1600
1400
1200
1000
800
600
400
200
0
Longitude
0
500
1000
1500
Latitude
Ship
Propeller
Path F
Longitude
0.007958
Latitude
0.09719
Heading
-1.546e-05
Surge
0.163
Sway
1.001
Yaw rate
-0.0004772
Elapsed time
0.09716
Simulation time
7.642
Start
Stop
Status
Running visualisation



4 References

Pivano, L., Smogeli, O., Johansen, T, A., Fossen, T, I. (2007). *Experimental Validation of a Marine Propeller Thrust Estimation Scheme.* *https://www.mic-journal.no/ABS/MIC-2007-4-2.asp/*

Fossen, T, I. (2011). Handbook of Marine craft Hydrodynamics and Motion control. *https://onlinelibrary.wiley.com/doi/book/10.1002/9781119994138*

The Nautical Institute. (n.d). *Dynamic Positioning.* https://www.nautinst.org/resource-library/technical-library/dynamic-positioning.html#:~:text=Dynamic%20Positioning%20(DP)%20is%20a,her%20own%20propellers%20and%20thrusters.

T. I. Fossen and T. Perez (2004). Marine Systems Simulator (MSS). URL: https://github.com/cybergalactic/MSS