**Tutorial – Self-propulsion in waves**

In this tutorial, a 6-DoF controlled self-propulsion of Iowa BB2 HDM is performed with target speed 10kts and target sail top depth 4m. The simulation is performed using PID controllers for the submarine control surfaces, propeller, and ballast tank. A monochromatic surface wave with wavelength and amplitude equivalent to most likely values for sea-state 5 is also simulated. Please referred to Iowa BB2 Model User’s manual for a more complete description of the solver and general setup.

1. **Case Setting**

1.1. Initial and Target Conditions

Most values for initial and target conditions are maintained to the default values provided in main.m and described in Table 1 of the User’s manual. Only the values of interest for this tutorial are discussed here. Target values for depth and speed are set as:

Usp = -5.14444;

target\_z = -15.3724;

Note that the target speed is negative in accordance with the definition of the coordinate system; also it is expressed in m/s. The target depth includes the distance from the CG to the top of the submarine sail, equivalent to -11.3724 m. These values are also used for the initial condition:

Bx\_0 = [0.0 0.0 target\_z 0.0 0.0 0.0];

Bv\_0 = [Usp 0.0 0.0 0.0 0.0 0.0];

n\_0 = 1.18;

Seastate = 5;

The initial suggested value of 1.18 s-1 for the propeller rotational speed is the deep-water value for 10 knots.

1.2. Controller settings

Most control values in Table 3 of the User’s manual are maintained equal to their default values. Of interest for this tutorial are

m\_switch = 0;

n\_switch = 1;

mass\_switch = 1;

LCG\_switch = 0;

which indicate that the maneuver is a self-propulsion using cruise controller and ballast tanks, but not the trim tanks. Figure 1 shows an expansion of the controller block of the Iowa BB2 Model (Referred to Fig 2 in User’s Manual). The Simulink block consists of seven independent controllers, of which only four (highlighted with a different color in the figure) apply to this case. Each of these blocks uses the generalized position and velocity vectors and the target values to calculate a controlled command to actuators, which are then combined and limited accordingly. Note that the control planes commands are combined twice: first, the appropriate control command (or imposed deflection) is selected to generate final vertical and horizontal commands; those are then combined to generate individual deflections of the sail planes ( and stern planes ( is the lower starboard plane, the upper starboard, the upper port, and lower port) as

|  |  |
| --- | --- |
|  | (1) |

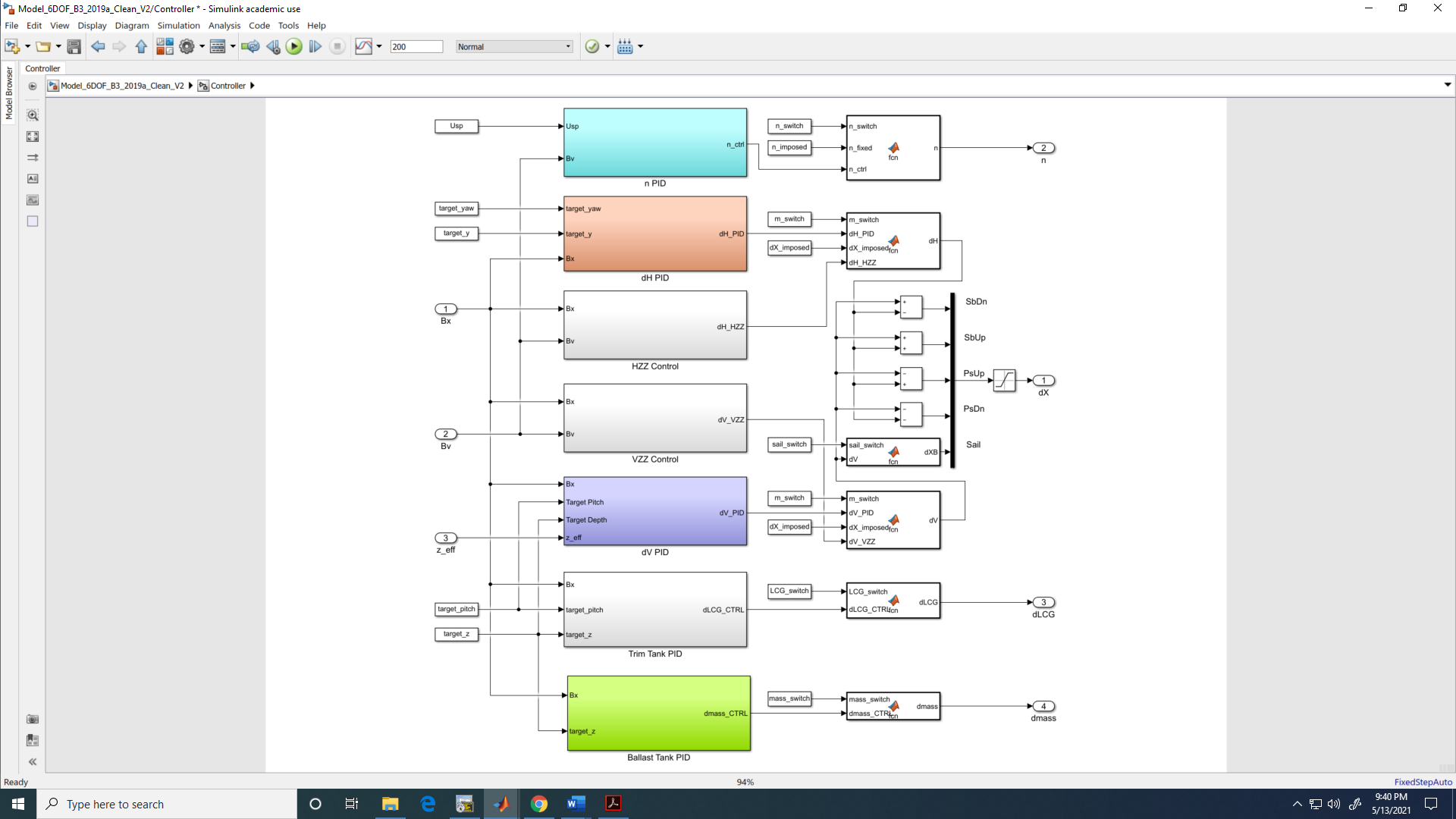


Figure 1. Controller block of BB2 Simulink Model **Model\_6DOF.slx**.

Each controller block corresponds to a PID controller acting on one or more controlled variables. For instance, the horizontal controller actuates on the stern planes to eliminate sway and yaw error:

|  |  |
| --- | --- |
|  | (2) |

The corresponding block, shown in Figure 2, generates the deflection command following Eq. 2, and it also applies rates and deflection limits. All other blocks follow a similar set up and are not discussed in this tutorial.

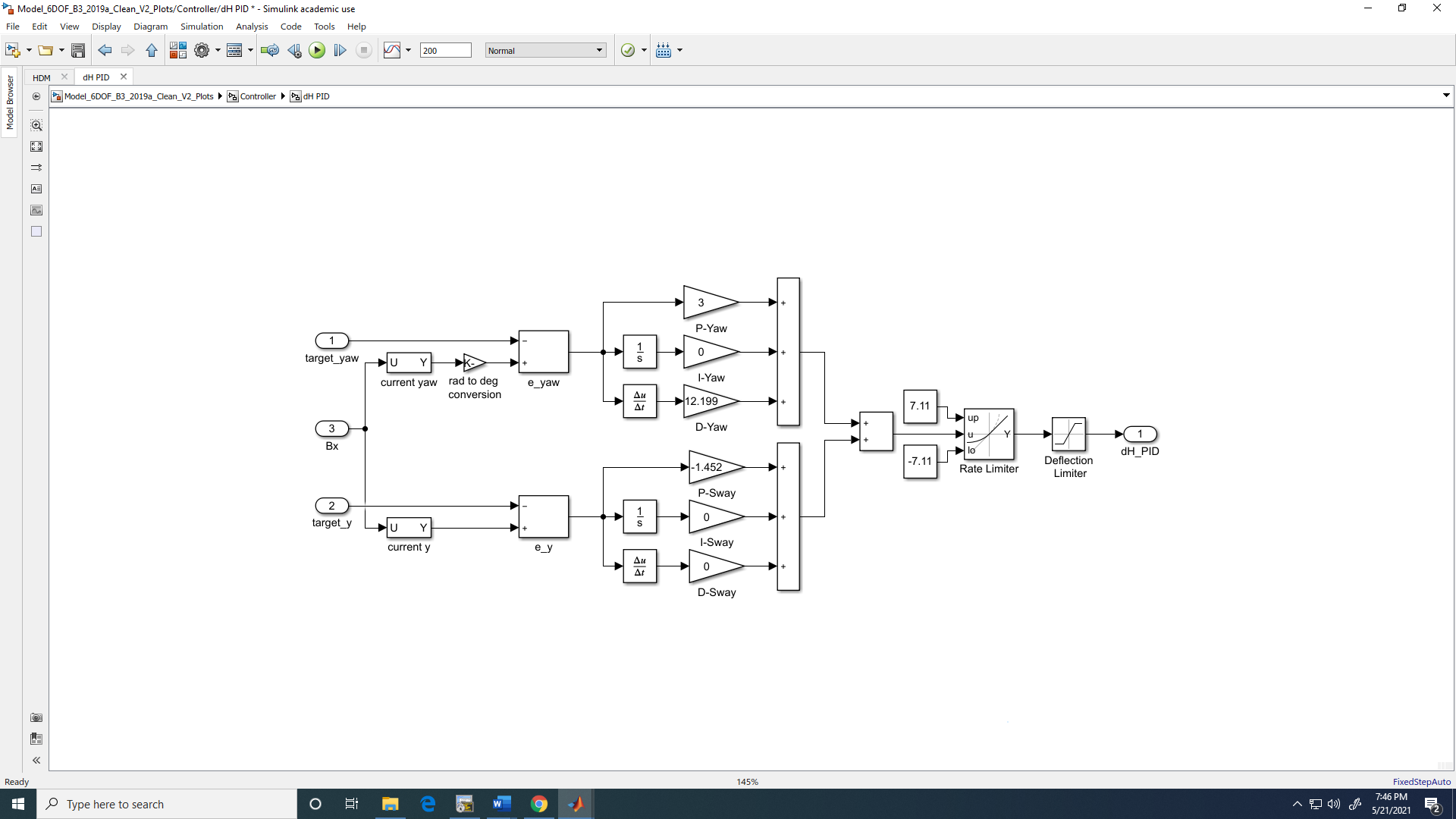


Figure 2. Horizontal PID controller block (orange block in Fig. 1).

1. **Running the model**

For this tutorial a fixed discrete time step size of and a total duration of are recommended. These are set in the simulation model settings prior to start the simulation. All the initial values set in **main.m** need to be loaded before starting the Simulink simulation. This is done by simply executing main as a command in the Matlab Command Window. At this point, the model is ready to run.

While running the simulation, the scope modules placed in the main Simulink module can be accessed to check the evolution of the simulation. Figures 3 and 4 show the position and velocity scopes at the end of the simulation Alternative, after the simulation is completed, the logged motions can be retrieved the variable **out** from the Matlab workspace.

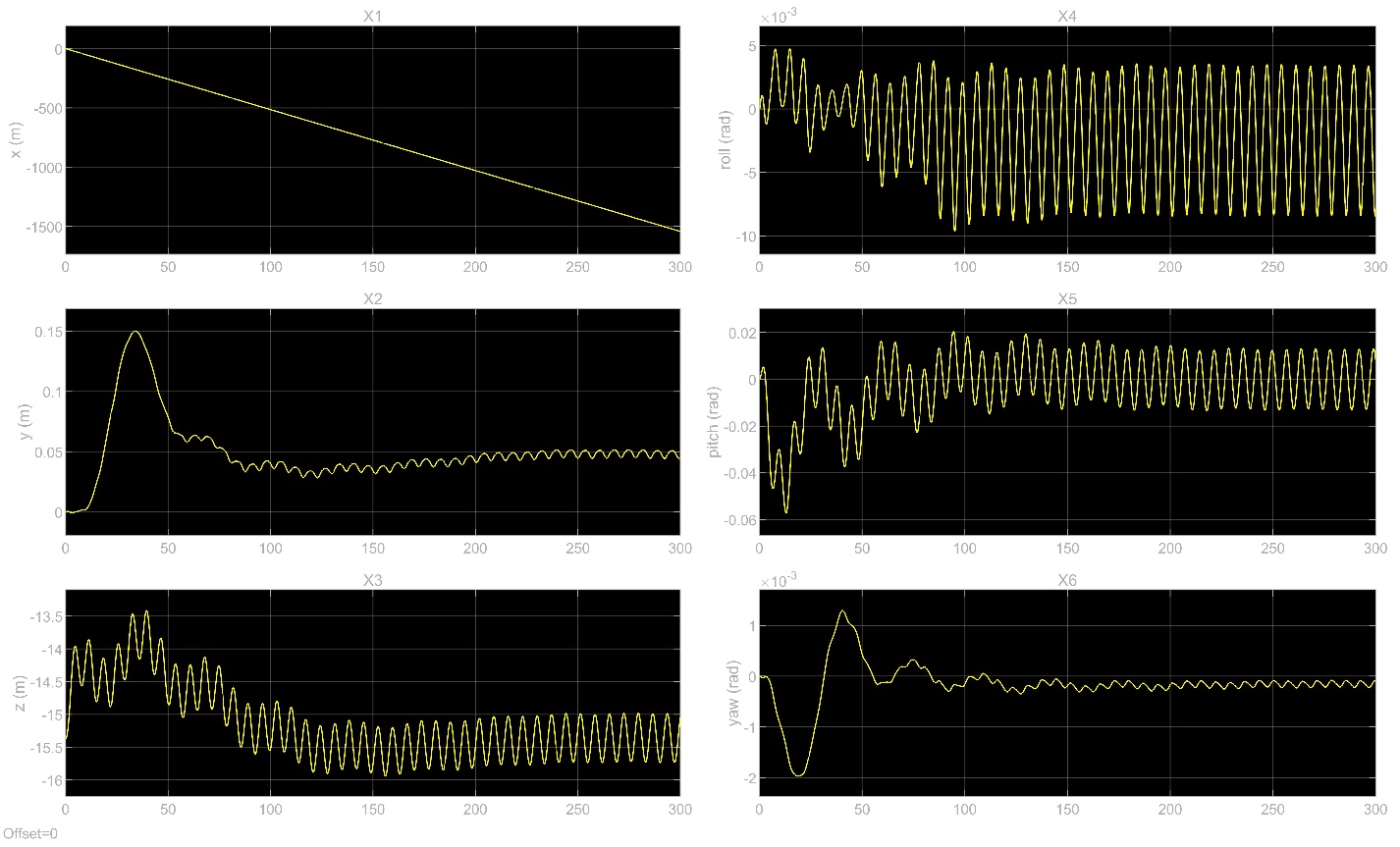


Figure 3. Motions time history from the scope module.

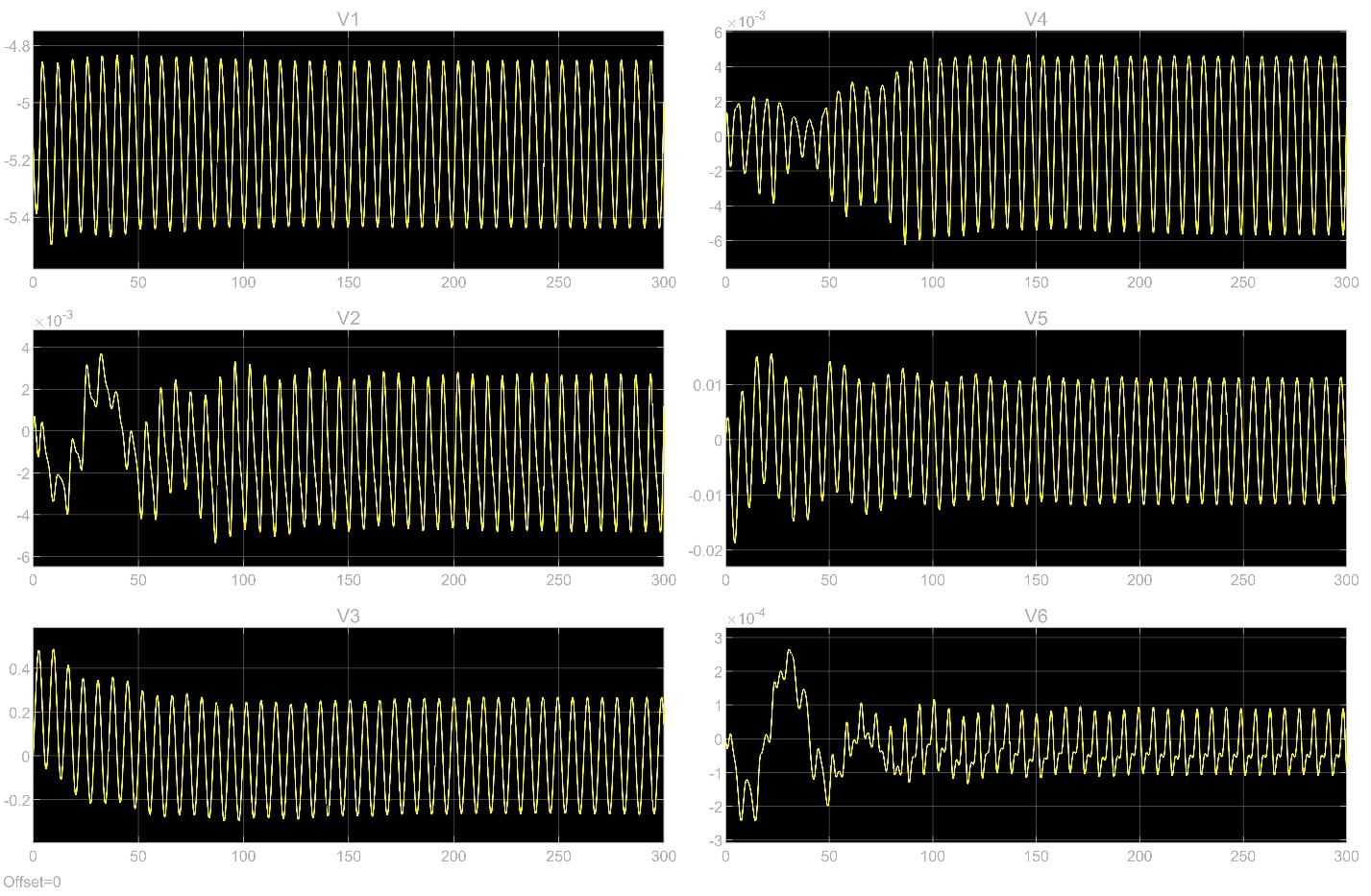


Figure 4. Velocity time history from the scope module.