Aero 759
HW9
Liam Hood

My network consists of 3 inputs; desired heading angle, current heading, and current turn rate. There is one hidden neuron with ‘tansig’ activation functions that then lead to a single linear output neuron producing the commanded rudder angle. Training was done using the Levenberg-Marquardt algorithm using Mean Squared error as the performance criteria. I used ‘feedforwardnet’ in matlab to take advantage of the validation runs to make sure that I don’t overtrain. I selected 75% of the data as training data and reserved the other 25% for validation. Epochs and performance goal were both set to be ambitious enough to never be achieved without first making validation results start to go in the wrong direction. The interesting thing about using this set up is that my training frequently failed early with relatively high performance, on the order of 1 or 0.10. If the performance reached 10^-3 it almost always eventually reached 10^-9 to 10^-11. The best training run that I achieved got a performance value of 3.2e-11 when training. RMS calculated by manually taking the square root of the mean of the squared error showed that the error in rudder angle when testing was 2.24e-6. The training RMS error in state was 9.77e-5. Testing can’t have RMS in rudder angle since there were not targets but the RMS in state was 8.62e-5 for the standard controller and 2.00e-5 for the one step ahead controller. Perhaps more importantly the largest the error got for the full controller in x-position was 0.126 milimeters, y-position was 0.222milimeters, and heading was 4.43e-4 degrees.

The network needs to decide on the rudder angle in order to get to a desired heading angle at the next time step. I quickly decided that the position data was irrelevant since the goal was only to match heading angle, regardless of where the boat was. The first obvious thing that was needed was the desired heading angle since that is the goal. The next was the current heading because the controller needs to know how much of a change is needed. The NN controller was effectively figuring out the turning rate that was needed and to figure out the rudder angle. Since the turning rate was a function of the current turning rate and the rudder angle I knew that I would need to include the current turning rate as an input.

With the three inputs selected I set about creating some training data. I did this by modeling the boat driving around in paths that weren’t really circular but were meant to include a full 360 of heading angle variation. This was done by adding a bias to a sinusoidally varying rudder angle. The training data consisted of two 25 second runs of this with paths circling in opposite directions. The goal was for the heading angle and turn rate to have greater variety and cover a wider range than the test path of the tightening turn.

If this network was controlling a real boat then we would have some issues since the in the testing data the rudder movements are not very smooth, jumping from 114 degrees one direction to 88 the other direction. If effects on velocity of turns were modelled then we see some issues with the control matching the desired values.

There was no need for the controller to be ready for a wider range of applications so the velocity, rudder lag, and time step were not included as inputs. This let the current values effectively be memorized by the network. If this was going to applied to a real problem I would have included those as inputs and added additional training paths with the varying values for all of those constants.

Network Diagram

Training Results

Training finished: Reached minimum gradient

Training Progress

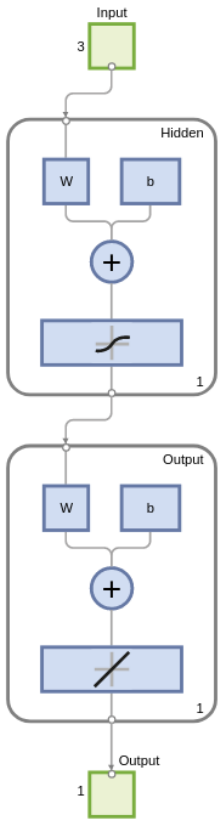
Unit	Initial Value	Stopped Value	Target Value
Epoch	0	9698	100000
Elapsed Time	-	00:00:07	-
Performance	0.622	5.01e-12	1e-13
Gradient	0.863	1.35e-09	1e-07
Mu	0.001	1e-11	1e+10
Validation Checks	0	1	6

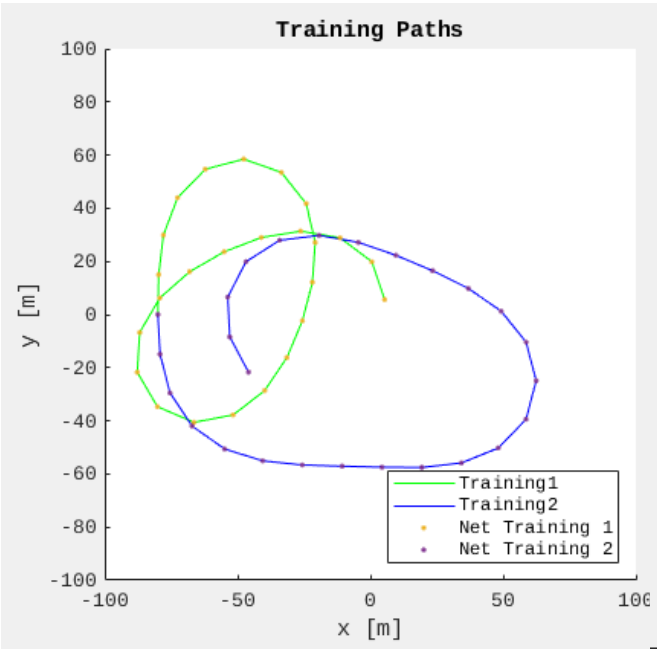
Training Algorithms

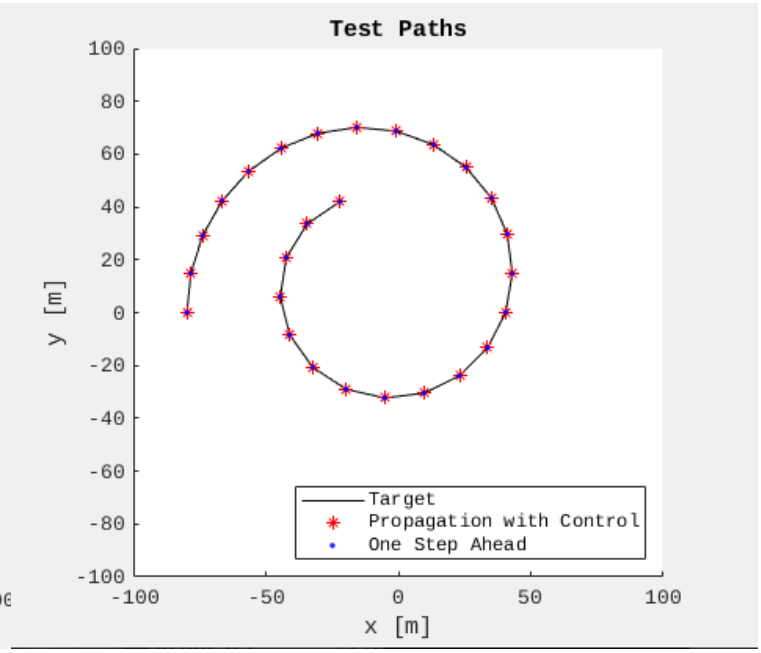
Data Division: Random dividerand
Training: Levenberg-Marquardt trainlm
Performance: Mean Squared Error mse
Calculations: MEX

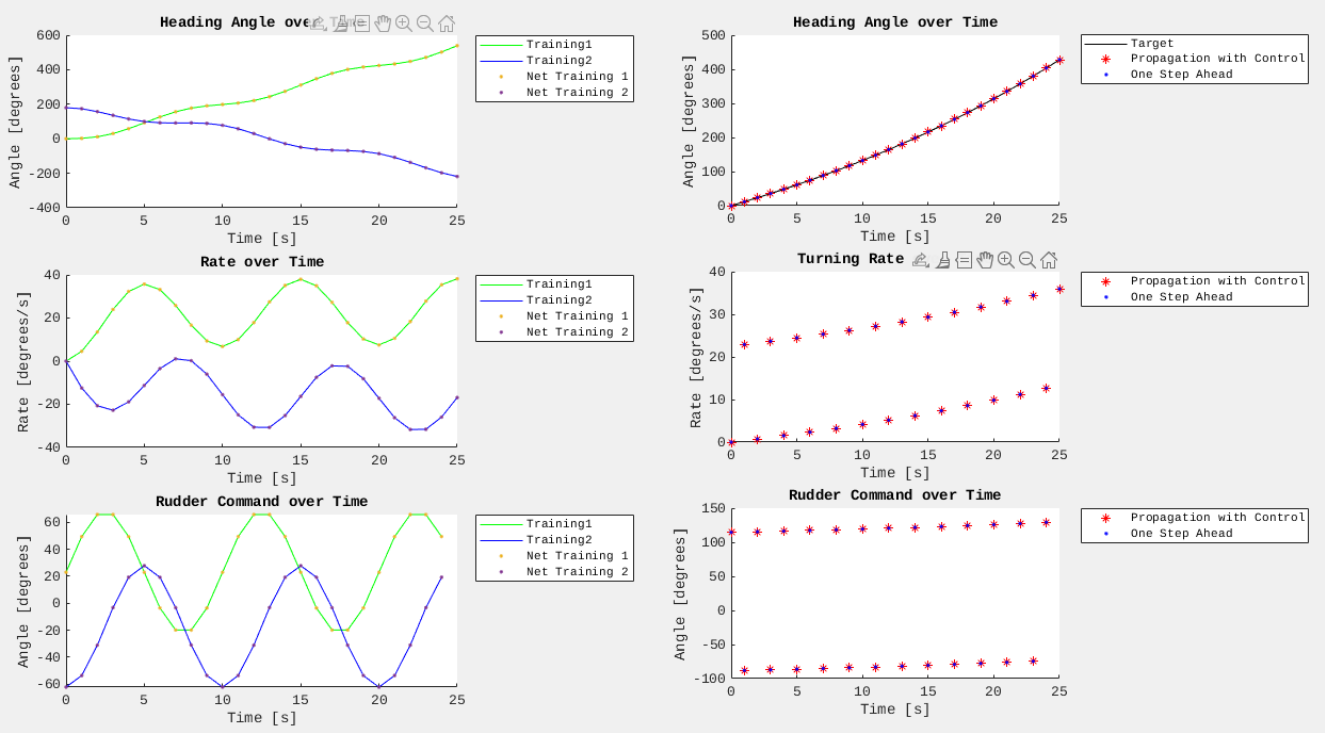
Training Plots

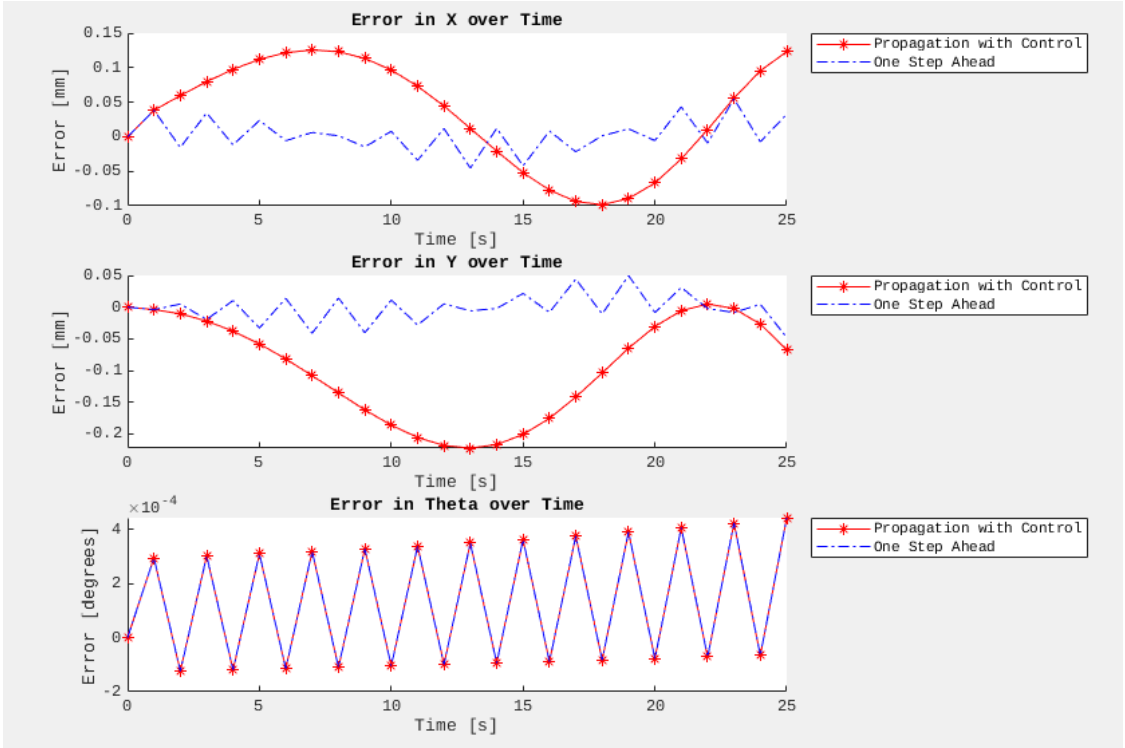
Performance | Training State
Error Histogram | Regression








Heading Angle ove
Training1
Training2
Net Training 1
Net Training 2
Angle [degrees]
Time [s]
Heading Angle over Time
Target
Propagation with Control
One Step Ahead
Angle [degrees]
Time [s]
Rate over Time
Training1
Training2
Net Training 1
Net Training 2
Rate [degrees/s]
Time [s]
Turning Rate
Propagation with Control
One Step Ahead
Rate [degrees/s]
Time [s]
Rudder Command over Time
Training1
Training2
Net Training 1
Net Training 2
Angle [degrees]
Time [s]
Rudder Command over Time
Propagation with Control
One Step Ahead
Angle [degrees]
Time [s]


Error in X over Time
Propagation with Control
One Step Ahead
Error [mm]
Time [s]
Error in Y over Time
Propagation with Control
One Step Ahead
Error [mm]
Time [s]
Error in Theta over Time
×10^-4
Propagation with Control
One Step Ahead
Error [degrees]
Time [s]