George Mason University

Compensator Design and Evaluation For Ship Heading Angle

Report from laboratory experiment 5 conducted on 23 February 2016
As part of ECE 429 Control Systems Lab
Course Instructor: Dr. Daniel M. Lofaro

Michael Kepler
G00804828

01 March 2016

The Volgenau School of Engineering

Objective

The objective of this lab is to design and analyze a system to control the ship heading angle of a ship based on time-domain performance objectives.

Experiment

Task 1:

The system plant is defined as:

$$G_p(s) = \frac{3.7424 * 10^{-3}(s + 5.3879 * 10^{-2})}{s(s + 8.4688 * 10^{-3})(s + 1.2870 * 10^{-1})} \tag{1}$$

The first task requires a compensator to be designed so that the system meets the following closed-loop specifications:

$$Maximum\ Percent\ Overshoot: (PO) = 25\ \%$$

*for a step input

$$Settling\ time: \quad (t_s) = 125\ s$$

*to within 2% of final value for a step input

The first step was to find the desired closed-loop poles to satisfy percent overshoot and settling time, which will be of the form:

$$s_d = -\zeta\omega_n \pm j\omega_n\sqrt{1-\zeta^2} \tag{2}$$

Where ζ was derived from the PO specification:

$$PO(\%) = 100e^{-\frac{\zeta\pi}{\sqrt{1-\zeta^2}}}$$

$$\zeta = \frac{|\ln\frac{PO}{100}|}{\sqrt{\pi^2 + \ln^2\frac{PO}{100}}} \tag{3}$$

Incorporating a safety factor (PO =10%) for maximum PO resulted in the following ζ:

$$\zeta = 0.5912$$

ω_n was derived from the following equation:

$$t_{s(2\%)} = \frac{4}{\zeta\omega_n}$$

$$\omega_n = \frac{4}{\zeta t_s} \tag{4}$$

Incorporating a safety factor (ts = 115 s) for settling time resulted in the following ζ:

$$\omega_n = 0.0588$$

Thus, plugging in ζ and ω_n into eqn. (2) the desired closed-loop poles were obtained

$$s_d = -0.0348 \pm j0.0475$$

These calculations were done with the following MATLAB code:

```
%%
%Determining Desired Closed Loop Poles
%Specifications
po = 10;  %(Percent)
ts = 115;  %(Seconds)For 2% of final value

%Calculation of closed loop poles
zeta = sqrt((log(po/100)).^2/(pi.^2 + (log(po/100).^2)))
wn = 4/(zeta*ts)
sdp=-zeta*wn+j*wn*sqrt(1-zeta^2)
```

The phase of the compensated open loop system should be -180 degrees. The system G_p evaluated at the roots s_d resulted in:

```
%Define transfer function
s = tf('s');
k = 3.7424e-3;
gp = k*(s+5.3879e-2)/(s*(s+8.4688e-3)*(s+1.2870e-1));
```

$$G_p(s_d) = -0.3623 + 0.0498i$$

Obtaining the phase of the frequency response:

```
phiRad = angle(freqResp);
phiDeg = phiRad*180/pi;
```

$$\begin{aligned}\phi &= 156.0262 \\ &= 156.0262 - 360 \\ &= -203.9738\ deg.\end{aligned}$$

Thus the necessary phase compensation is the difference between the phase and -180 degrees:

$$\begin{aligned}\phi_c &= -180 - \phi \\ &= -180 - (-203.9738) \\ &= 23.9738\ deg.\end{aligned}$$

If:

$$\boldsymbol{\phi_c > 0 \rightarrow Lead\ Compensator}$$
$$\phi_c < 0 \rightarrow Lag\ Compensator$$

The lead compensator should contribute the amount of phase calculated in the previous step at the desired closed loop pole location.

$$G_{c\ Lead} = k\frac{s - z_{cd}}{s - p_{cd}} \tag{5}$$

One possible location is to cancel the pole closest to zero of G_p.

$$z_c = -8.4688 * 10^{-3}$$

```
z = -8.4688e-3;
```

The new zero adds phase to the system:

$$\phi_{zc} = \angle(s_d - z_c) \tag{6}$$
$$\phi_{zc} = \angle\big((-0.0348 \pm j0.0475) - (-0.0084688)\big)$$
$$\phi_{zc} = 119.0076\ deg.$$

```
phizc = angle(sdp - z)*180/pi;
```

The phase needed by the pole to compensate the new phase is:

$$\phi_{pc} = \phi_{zc} - \phi_c \tag{7}$$
$$\phi_{pc} = \phi_{zc} - \phi_c$$
$$\phi_{pc} = 119.0076 - 23.9738$$
$$\phi_{pc} = 95.0337\ deg.$$

```
phipc = phizc - phiComp;
```

The new pole location was calculated by:

$$p_c = real\{sd\} - \frac{imag\{sd\}}{\tan\phi_{pc}} \tag{8}$$
$$p_c = -0.0356 - \frac{0.0749}{\tan 95.0337}$$
$$p_c = -0.0306$$

```
p= real(sdp)-imag(sdp)/tand(phipc)
```

Knowing that the magnitude of the open loop system at the desired closed-loop poles on the root locus must equal 1, the gain K can be calculated:

$$|G_{cLead}(s_d)|\,\left|G_p(s_d)\right| = 1 \tag{9}$$

$$1 = \left|K\frac{(s+0.0084688)}{(s+0.0306)}\right| * \left|\frac{3.7424*10^{-3}(s+5.3879*10^{-2})}{s(s+8.4688*10^{-3})(s+1.2870*10^{-1})}\right|$$

$$for\ s = s_d = -0.0356 \pm j0.0749$$

$$1 = K * 0.6490$$

$$K = 1.5408$$

```
kc = 1
gclead = kc*(s-z)/(s-p);
g = gclead*gp
mag = abs(evalfr(g,sdp))
```

Resulting in the following the lead compensator:

$$G_{cLead} = \frac{1.5408(s+0.0084688)}{(s+0.0306)}$$

```
kc = 1.5408
gclead = kc*(s-z)/(s-p);
```

Yielding the final open-loop system:

$$G(s) = G_{cLead}(s) * G_p(s)$$

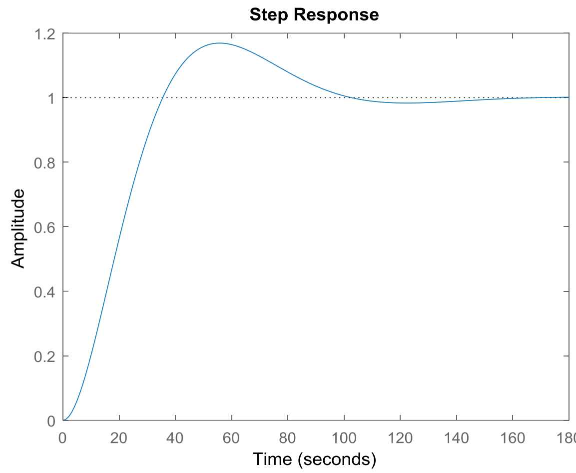
$$G(s) = \frac{1.5408(s+0.0084688)}{(s+0.0306)} * \frac{3.7424*10^{-3}(s+5.3879*10^{-2})}{s(s+8.4688*10^{-3})(s+1.2870*10^{-1})}$$

```
g_ol = gclead*gp
```

Checking the closed-loop system response to a step input:

```
g_cl = feedback(g_ol,1);
stepinfo(g_cl)
step(g_cl
```

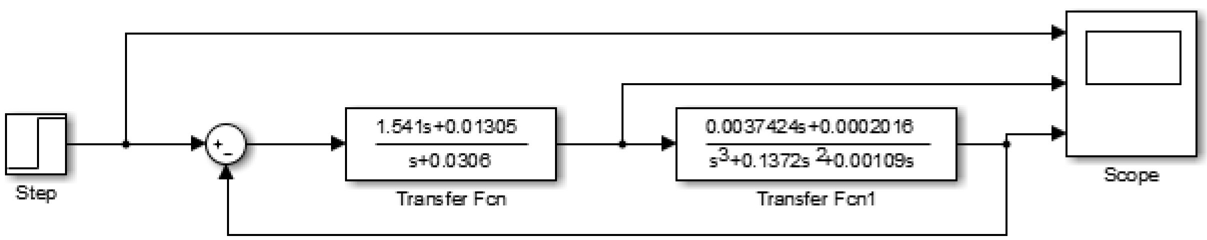
```
ans =

       RiseTime: 24.3277
   SettlingTime: 94.6537
    SettlingMin: 0.9214
    SettlingMax: 1.1692
      Overshoot: 16.9182
     Undershoot: 0
           Peak: 1.1692
       PeakTime: 55.4250
```



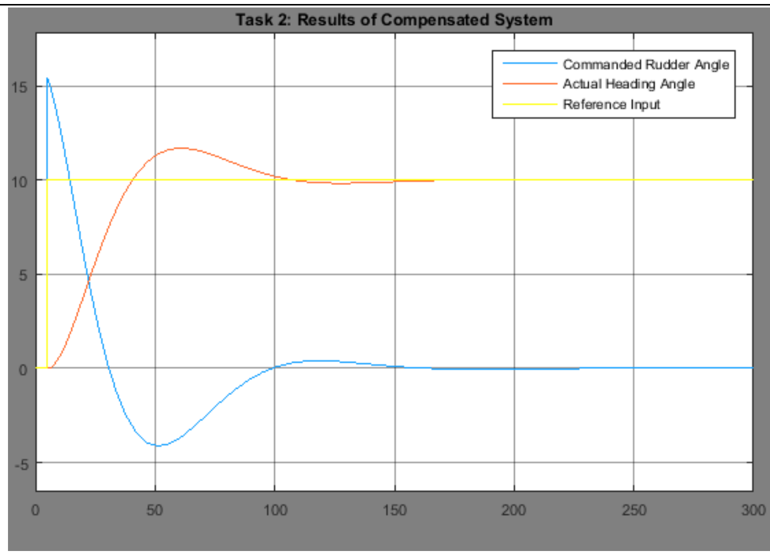
The compensator enables the system to meet the specifications for settling time and percent overshoot.

Task 2:

Task 2 asks the student to create SIMULNK model for the total closed loop system:



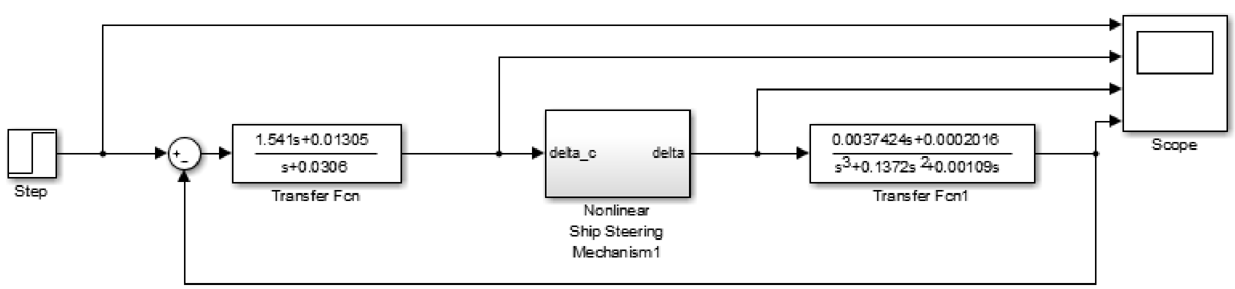
The step input began at t=5s and had an amplitude of 10. The simulation ran for 300 seconds and the numerical integration method was ode23s.

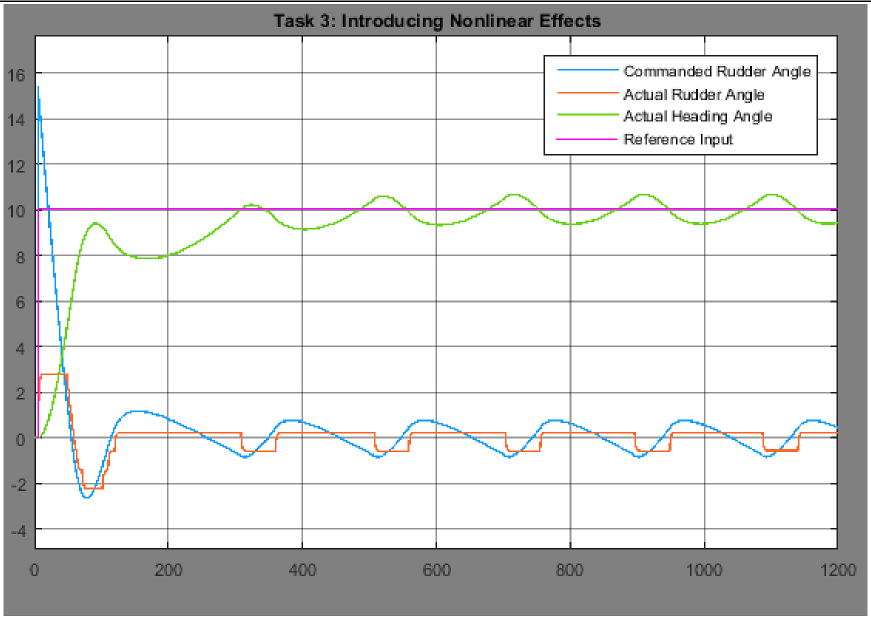


The yellow line represents the reference input, which was set to 10 degrees, the blue line represents the rudder angle, or the error signal, and lastly the orange line represents the heading of the ship.

Task 3:

Task 2 modeled an ideal linear system. In reality the 'actual rudder angle' is a nonlinear transformation of the 'commanded rudder angle.' Task 3 models this behavior by incorporating a nonlinear ship steering mechanism which limits the 'actual rudder angle' to $\pm$ 3 degrees.





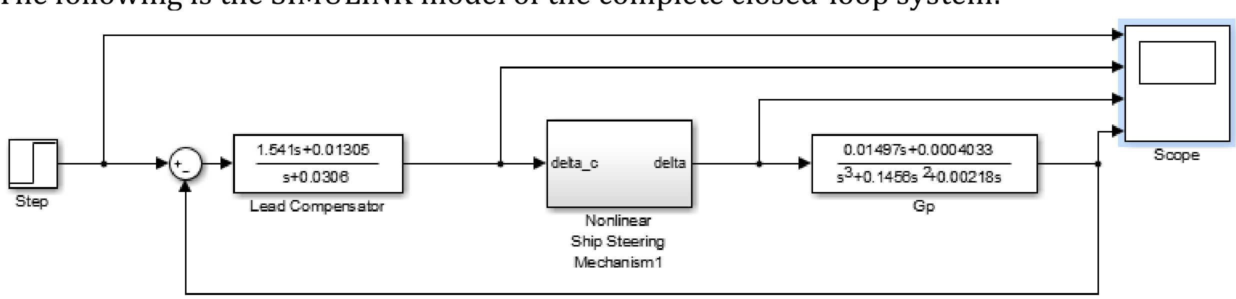
In comparing the output from task 2 and task 3 it can be seen that the steering mechanism creates a limitation on how much the angular range of the rudder (approximately between $\pm$ 3 degrees). This limitation forces the output of the system to be marginally stable, oscillating between ~9 and ~11 degrees as opposed to the system without the steering mechanism which was able to achieve stability at 10 degrees.

Task 4:

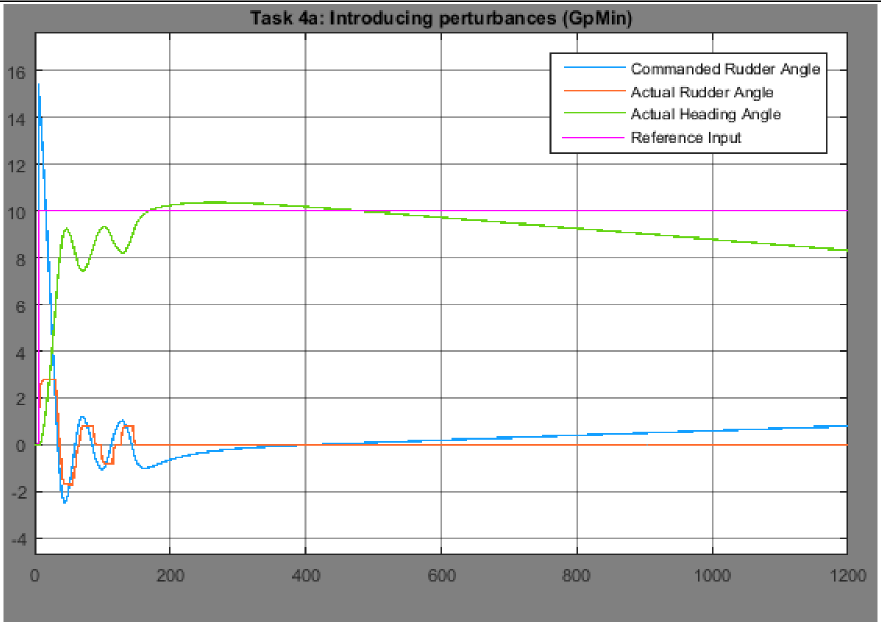
Task 4 is geared towards testing the robustness of the compensator as perturbations are introduced into the system. Thus the first new plant is:

$$G_{pmin}(s) = \frac{1.4970 * 10^{-3}(s + 2.6940 * 10^{2})}{s(s + 1.6938 * 10^{-2})(s + 1.2870 * 10^{-1})} \tag{10}$$

The following is the SIMULINK model of the complete closed-loop system:

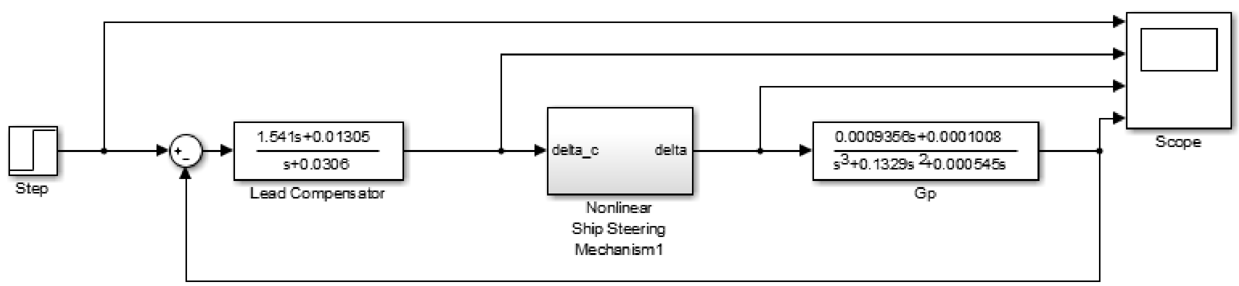


Which results in the following output:

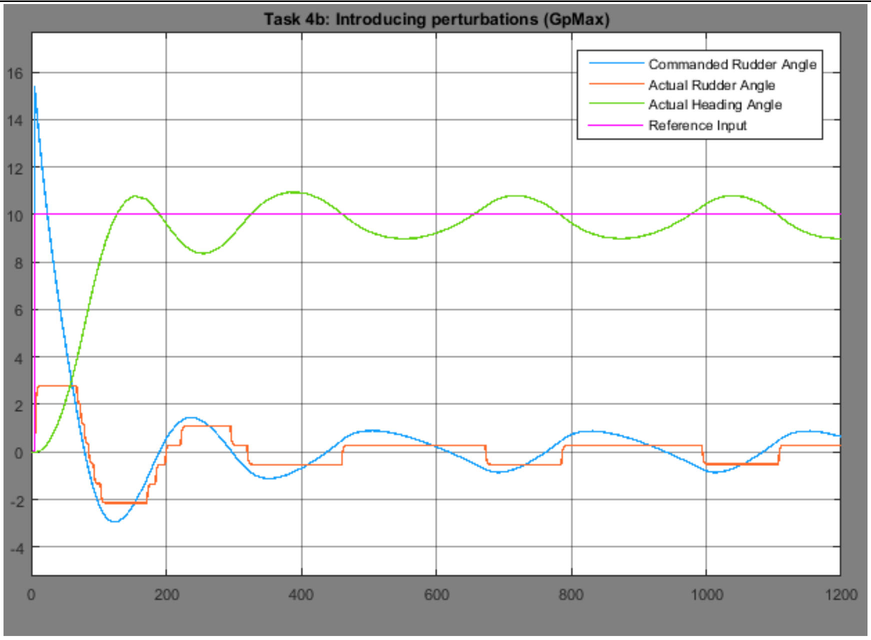


The second plant to test is:

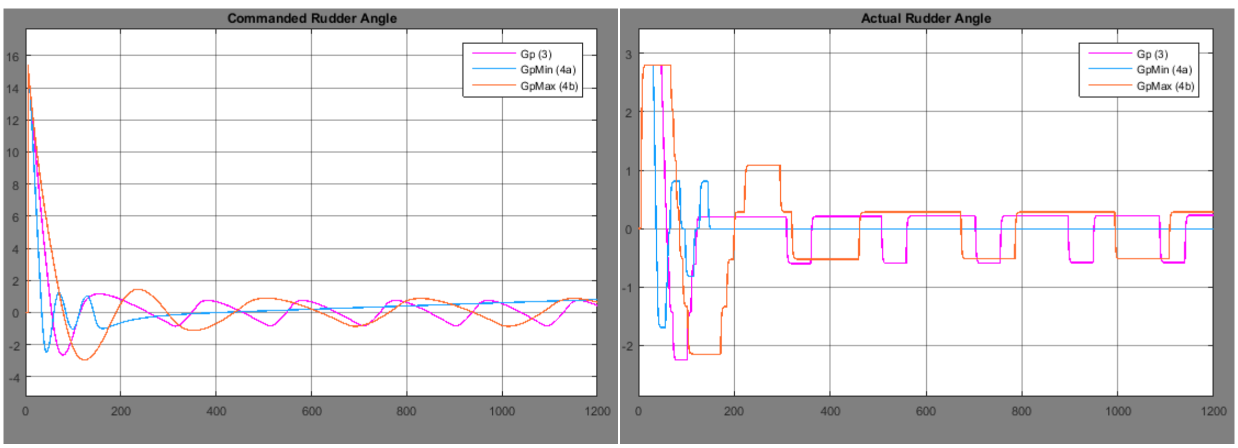
$$G_{pmax}(s) = \frac{9.356 * 10^{-4}(s + 1.0776 * 10^{-1})}{s(s + 4.2344 * 10^{-3})(s + 1.2870 * 10^{-1})} \tag{11}$$

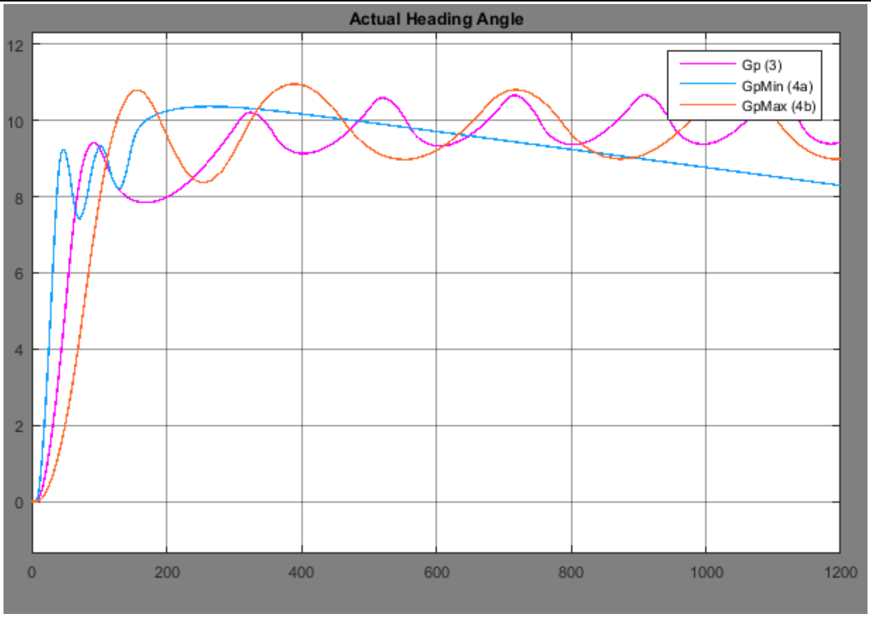


Which results in the following output:



This next section includes graphs that compare the commanded rudder angle, the actual rudder angle, and the actual heading angle.





Conclusion

As can can be seen from the graphs, the actual heading angle is controlled, but not controlled successfully. Ideally the ship heading should settle at 10 degrees, rather than have oscillations around 10 degrees. The nonlinear distortion places the system into a state of marginal stability. The frequency of the response GpMin is much lower than GpMax as can be illustrated by the 5 time periods present in the graph for GpMax in contrast to not even a full oscillation for the response of GpMin. The systems do not have a settling time since their values do not stay within a 2% steady-state range.

Even though the locations of the closed loop poles were designed to meet the specifications of the system, they were not realized after the nonlinear distortion was introduced. Implementing the SIMULINK model is simple and visually intuitive, but writing the code in MATLAB is better suited for custom manipulations and obtaining precise information about a system with functions such as 'stepinfo()'