

UNIVERSITY OF CALIFORNIA, SANTA CRUZ
BOARD OF STUDIES IN COMPUTER ENGINEERING



CMPE-242:
APPLIED FEEDBACK CONTROL

HOMEWORK #8
DUE 09-MAR-2017



Attitude Stabilization of a 1-D satellite: Very often in satellite/space ship design, you wind up putting the big heavy things (engines, etc) on one end, and the sensitive sensors on the other end, and because you don't have to support the weight (microgravity environment), you use a slender flexible truss to move them apart. We model the torsion stages as three inertias connected by two torsional springs. To make matters more difficult, the thrusters that can torque the satellite are connected to the aft stage, not to the front where the sensors are. You do get a star tracker on each stage which gives you very high quality rotational attitude information on the fore and aft stages (but not in the middle).

Collocated Control: The rotational dynamics of the entire satellite system, aft thrusters to aft angle, can be represented by the following transfer function (this is the easy one):

$$G_{AFT}(s) = \frac{\Theta_{AFT}(s)}{U(s)} = \frac{K_1(s^2 + 2\zeta_{z1}\omega_{z1}s + \omega_{z1}^2)(s^2 + 2\zeta_{z2}\omega_{z2}s + \omega_{z2}^2)}{s(s+p)(s^2 + 2\zeta_{p1}\omega_{p1}s + \omega_{p1}^2)(s^2 + 2\zeta_{p2}\omega_{p2}s + \omega_{p2}^2)}$$

Where:

$K_1 = 30.20$	$p = 0.5 \text{ rad/s}$
$\zeta_{z1} = 0.026$	$\omega_{z1} = 10.68 \text{ rad/s}$
$\zeta_{z2} = 0.018$	$\omega_{z2} = 40.83 \text{ rad/s}$
$\zeta_{p1} = 0.020$	$\omega_{p1} = 14.95 \text{ rad/s}$
$\zeta_{p2} = 0.018$	$\omega_{p2} = 41.97 \text{ rad/s}$

Now, having done our previous simple designs, let's explore the design space a bit:

1. Go back to your design $D_1(s)$, from HW #7(a), and re-implement it discretely (with Tustin) using slower and slower sample rates. Remember, $D_1(z)$ will change every time you change the sample rate. How slow can you go before performance breaks down?
2. Simulate your designs using the simulink model *DiscoveryAftDiscrete.mdl* from the website, and overplot the step responses against your original continuous time response. Use either "stair" or "stem" command to plot the digital response.

3. Explain why the performance degrades as the sample time gets slower. Be specific, calculate the actual phase loss.
4. Compute the ZOH equivalent of the transfer function, $G_{AFT}(s)$, and try to design $D_1(z)$ directly in the z-domain. Compare the performance against the $D(z)$'s that you got above in part (1). You should be able to achieve good performance at much lower sample rates.

Non-Collocated Control: When you did the control above, the output sensor and the actuator were in the same place. This is called *collocated* control. You should have observed that the high frequency poles tended to have zeros close by that caused the poles to move in favorable (stabilizing) directions when you cranked up the gain. This is typically the case for collocated control. What happens when the sensor and the actuator are not in the same place? This is non-collocated control, and it gets harder. The rotational dynamics of the entire satellite system, aft thrusters to fore angle, can be represented by the following transfer function (this is the hard one):

$$G_{FORE}(s) = \frac{\Theta_{FORE}(s)}{U(s)} = \frac{K_2}{s(s+p)(s^2 + 2\zeta_{p1}\omega_{p1}s + \omega_{p1}^2)(s^2 + 2\zeta_{p2}\omega_{p2}s + \omega_{p2}^2)}$$

Where:

$K_2 = 5,746,500$	$p = 0.5 \text{ rad/s}$
$\zeta_{p1} = 0.020$	$\omega_{p1} = 14.95 \text{ rad/s}$
$\zeta_{p2} = 0.018$	$\omega_{p2} = 41.97 \text{ rad/s}$

Again, revisiting the controller design that we did in HW #7:

5. Design a new compensator, $D_3(s)$, that uses a lead and notch for the non-collocated system. The notch zeros should be “near” the poles of the first resonant mode of the uncompensated system (i.e.: near ω_{p1}), but do not cancel them. Experiment (in MATLAB) with where to put them, remembering that the exact location of them is unknown. Try to be robust. Also, experiment with where to place the poles of the notch, and see how that affects the system. See if you can match the performance of the system in 7-1(a), i.e.: closed-loop bandwidth of 2π and a phase margin of 55° (you might not be able to make it).
6. Use *DiscoveryFore.mdl* to simulate the system with your new controller.
7. Once you are happy with your design for $D_3(s)$, go ahead and digitize it using Tustin for a range of sample rates. Again, see how slow you can go and still get decent performance. Simulate using *DiscoveryForeDiscrete.mdl*, and show your results (clearly and well reasoned, please).
8. Redesign a new compensator, $D_4(s)$, using the linear phase offset to compensate for the sample delay, and see how much better you do than in (7).
9. Optional: Convert $G_{FORE}(s)$ into $G_{FORE}(z)$ using a ZOH transformation (use MATLAB), and redesign in the z-domain. See what kind of performance you can achieve in terms of bandwidth and phase margin, along with how slow you can sample.

Note: use anything that MATLAB has to offer here, and use this “lab” as an opportunity to stretch yourself in terms of understanding the material. While this is still a fairly canned exercise, it is much closer to the real world than anything else you have done. See how robust your designs are, what happens if you are off on your pole locations (if you really want to be adventurous, do a root locus based on a scaling of the pole locations, and see what happens). Again, the more you put into this, the more you will get out of it. Don’t fail your other classes, but definitely use this opportunity to learn as much as you can.