University of Maryland at College Park
Dept. of Aerospace Engineering

ENAE 432: Aerospace Control Systems

Final Examination: Design Project

Part II

This project is not a homework, but rather the **approved, formal equivalent of a final examination in ENAE 432**. This is the first of two parts of the project, aimed to get you started thinking about the dynamics of the system and the requirements for your design. More detailed specifications for the final write-up will be contained in the second part of the project, which will follow in about one week.

As this is an examination, **no collaboration of any form is permitted. You must only work individually, and may not discuss your work with anyone else – whether in the class or not**. This means current and former TAs, current and former students, tutors, internet chat groups or message lists, and any other forms of communication you may have been using this semester are *unauthorized and prohibited* sources of information for this project. Similarly, you may not reference any materials beyond those explicitly distributed to you in this semester's class or those available in the suggested texts listed in the syllabus. Any materials beyond this approved set, including any past year course materials or materials downloaded from the internet, are *unauthorized and prohibited* sources of information for this project.

Any evidence of collaboration, discussion, or use of unauthorized sources, for any aspect of this project constitutes academic dishonesty and will be immediately forwarded for judicial review with the attendent potential consequences. Note as well that "apathy or acquiescence in the presence of academic dishonesty is not a neutral act" (quoting from the UMD code of academic integrity). To witness academic dishonesty and remain silent is to be complicit and culpable in that dishonesty.

I remind you again that **this project is your final examination** – while you will be working on it outside of class, it is in no way similar to a homework that you are free to discuss with others. You are in every way subject to the same rules governing an in-class examination, save that you may use computers and course materials to assist your effort.

The only permissible source of discussions about this project is with me personally, and the contents of those discussions may not be shared with others. To the extent that a clarification arises in a private discussion that would be helpful to the class as a whole, an announcement will be posted. However, *as this is your final exam*, I will answer in meetings (or online) only factual questions about the theory discussed in the class, clarifying questions about the subject or general objectives of the project, and practical details about using any provided project code. In particular, I will not give specific guidance on the details of your system modeling, data interpretation, controller design, controller implementation, controller performance, etc. These are entirely up to you, and indeed are the primary basis for your final grade on the project.

Mission Scenario

For the viewing mission, the space telescope is actually controlled about two different axes, allowing the star field in the viewing plane to move in orthogonal directions (tip/tilt). The telescope is assumed completely symmetric, so that each rotation axis has the same (flexible) dynamics, hence the same transfer function $G(s)$ (both symbolically and numerically), and each axis will be controlled independently using the same $H(s)$ that you design based on your $G(s)$. This $G(s)$ will be the same that you compute from the Part 1 equations and data.

An illustration of the star field on the viewing plane of the telescope is shown in the attached figure. The location of the target star is actually specified by two coordinates in this plane, which are used as the desired pointing angle y_d input to your controller. Given the symmetry of the system, the pointing response will be computed about each axis separately to determine the planar motion of the star field as the telescope rotates. Thus, your controller will actually be used twice in simulating the motion, first for the "tip" axis, then again for the "tilt" axis.

There is nothing you need to do in the design process to reflect this "dual" usage – you will still design a single $H(s)$, with a single $y(t)$ and a single (constant) y_d as its inputs, and a single $u(t)$ as its output. However, in simulation this design will be used twice, once for each axis of motion.

Pointing Requirements

The mission will begin with specification of the location (y_d) of the target star, which will be randomly within ± 3 arcminutes away from the initial pointing direction of the telescope. The telescope will also have some small amount of initial rotation rate and may be vibrating slightly. Your controller will become active at this time.

The viewing phase of the mission will start 15 seconds after the the target is specified and your controller is activated. At this point, the telescope will begin to slowly zoom in on the target. The viewing mission will end 120 seconds after the target is specified. The performance of your design is determined by two main factors: the fraction of the (105 second) viewing time the target star (indicated in solid red) is inside the primary light collection zone of the telescope (dark red ring), and the fraction of the viewing time that the target is completely covered by the occlusion disk (solid green circle).

Counters during the mission will track the amount of time each criterion is satisfied. However, the quality of the data collected is determined by the *consecutive* amount of time each criterion is met near the time of maximum zoom. The counters will thus be reset to zero each time a criterion is violated. Your design will be scored as follows: 5 points are awarded simply for keeping the target star in view at all times, even as the telescope zooms in. Up to 10 additional points are awarded proportional to the fraction of the viewing time the star is inside the red ring. Finally, up to an additional 15 points are awarded proportional to the fraction of the viewing time the star is completely under the occlusion disk.

The maximum design score is thus 30 points, and this score will comprise 30% of your grade on this project. The remaining 70% of your grade will come from the quality, accuracy, and completeness of the technical analysis which leads to your controller design.

The single axis "pointing stablility"[1] requirements to satisfy each performance category are: to remain within the red ring, the required single axis accuracy is about 9 arcseconds (1 arcmin = 60 arcsecs), while for the occlusion disk to completely cover the target star, the required single axis accuracy is about 500 mas ("mas" = milliarcseconds). These are slightly conservative requirements, since they assume worst-case error simultaneously in both pointing directions.

Disturbances and Noise

There are two sources of disturbance: environmental effects, primarily from solar pressure, and imperfections in the reaction wheel, primarily due to friction and mass imbalance. This leads to a disturbance model of the form

$$d(t) = d_0 + d_1 \sin(\omega_1 t + \phi_1)$$

where d_0 is typically between 2-6 mN·m (milliNewton-meter), d_1 is typically between 50-150 mN·m, ω_1 is at least 1.5-2 times larger than the natural frequency of the structural mode, and ϕ_1 can be anywhere in the range $[-\pi, \pi]$.

The sensor noise $n(t)$ is essentially random, with an amplitude of between 100-300 mas, but is mostly concentrated at frequencies at or above 3 times the natural frequency of the structural mode.

Simulated Flight Tests

The provided file `simview` can be used to run a simulation of your viewing mission, using a control strategy you specify. For this, the control algorithm which corresponds to your compensator $H(s)$ must be embodied in an m-file called `control.m`, which is stored in the same folder as the `simview` function. The `simview` function will call your `control` function to generate the inputs $u(t)$ to the spacecraft model during the simulation. Your `control` function must be defined so that `u=control(yd,y)` will assign the desired value to `u` given the current values of `yd` and `y` it receives as inputs. The `simview` function will vary the inputs to your `control` function as the simulation evolves, and use the resulting values of `u` that you compute to propagate the dynamics model of the rotating spacecraft.

[1] Astronomers use a different definition of "stability" in this context, quantifying how steady and precise the pointing angle is. In our language, these are really bounds on the magnitude of the steady-state pointing error.

`simview` will call your `control` function at a rate of 10 times per (simulated) second to update the steering commands $u(t)$ to the vehicle, with this input held constant in the model between updates. Thus, there is a 10Hz update rate for the controller calculations.

Usage of the `simview` function is as follows:

```
[t,yd,yma,ua] = simview(G,Km);
```

where `G` is a Matlab transfer function object holding your model of the spacecraft dynamics, and `Km` is a constant holding the value of K_m for the reaction wheel electronics. Note that this constant will also be part of your definition of `G`, but the simulation needs `Km` separately to correctly simulate the physics.

After the function runs, the variable `yma` will hold the angle measurements about each axis recorded during the simulation, at the times contained in the array `t`. There will be two columns in `yma` – one for each of the two rotational axes. The corresponding targets for the each angle during the simulation will be contained in the two element array `yd`. Finally, the variable `ua` will contain the control inputs computed by your `control.m` function during the simulation, again at the times given in the array `t`. Like `yma`, there will be two columns in `ua`, one for each of the two rotational axes.

In addition to the numerical outputs, the results of the simulation will also be displayed as a very basic animation of the resulting star field motion. The performance score for the simulation will also be shown as text output in the command window. Note that the simulator will use random values for the disturbance and noise parameters, within the ranges described above, each time it runs. Depending on your design, this may result in very different scores each run. In the final grading analysis, a single fixed set of parameters for the disturbance and noise characteristics will be used for all student submissions.

Important: Unlike `testdata`, the `simview` function does *not* use your UID and hence *does not know* what the "actual" dynamics of your spacecraft model are. This function simply takes the transfer function $G(s)$ and numerical value for K_m that you provide as input, and uses these as the "truth model" for the simulation. However, when your design is graded, the actual "truth model" that generated your `testdata` results will be used. If your estimates of the transfer function or its numerical parameters are incorrect, the results you see from `simview` may not represent the actual behavior of your spacecraft when I grade its performance using the exact model which corresponds to your UID. You should thus only use the `simview` function after you are confident you have a reasonably accurate model of your system from the `testdata` results, and have developed a suitable control strategy based on this model.

Final Report Preparation and Submission

We are going to use a more systematic, structured presentation format for the projects this year, to make preparing (and grading!) the required documentation easier. Details of this will be posted on ELMS shortly. Reports need not be typeset, but must be neat and arranged according to the specified structure. Up to 10 additional bonus points in the final scoring are awarded for the neatness, clarity, and professionalism of your final write up (making the maximum possible score 110). There is a very high bar for these bonus points; the slightest whiff of BS in your analysis or discussions will be an automatic disqualification.

The formatted report packet must be submitted to me no later than 4pm on 21 May. As usual, a box will be available outside my office to receive these packets. Early submission is encouraged; these can be handed to me directly in class or during office hours, or else placed in my mailbox in the Aerospace main office.

Finally, **the `control.m` file that implements your controller must also be uploaded to ELMS before the submission deadline**. This file is used in the performance scoring; you will automatically lose all (30) potential performance points in the grading if this file is not uploaded to ELMS. Your `control.m` file must accurately implement the control strategy you document in your project report. Otherwise, substantial points will be deducted regardless of how well your submitted code performs on the pointing test.

Figure 1: Example star field view. Target star in red, observing ring in dark red, occlusion disk in green. Ideally, you want to point the telescope so the green disk completely covers the red star during the last 105 seconds of the simulation.