Controller design for a Control Moment Gyroscope

Rohen A. Agarwal
Junior (2021), Aerospace Engineering, University of Illinois at Urbana-Champaign

This is the project report for the first AE 353 project. In this report, the journey from having a non linear system governed by differential equations, to designing and implementing a controller that linearizes the performs attitude control maneuvers on a Control Moment Gyroscope is detailed.

I. Nomenclature

A = Matrix of size n_x x n_x
B = Matrix of size n_x x n_u
C = Matrix of size n_y x n_x
D = Matrix of size n_y x n_u
K = Matrix of size n_u x n_x
F = Matrix of size n_x x n_x
q_1 = Angle of platform
q_2 = Angle of gimbal
v_1 = Angular velocity of platform
v_2 = Angular velocity of gimbal
v_3 = Angular velocity of rotor
$\dot{v_1}$ = Angular acceleration of platform
$\dot{v_2}$ = Angular acceleration of gimbal
$\dot{v_3}$ = Angular acceleration of rotor
τ_2 = Torque on gimbal by platform
τ_3 = Torque on rotor by gimbal
x = State
$\dot{x}$ = Time derivative of state
u = Input
y = Output

II. Introduction

THIS document follows a series of steps needed to be taken from understanding the problem, to successfully designing an appropriate controller. For this project my aim was to re-orient a platform on which a CMG (control moment gyroscope) is mounted, to a certain angle chosen by me such that once that angle is achieved the platform and along with the Control moment gyroscope is a syntactically stable. To do this I must first design a controller, implement it, test and see if my controller works for my purpose. If it doesn't work for my purpose I must modify my controller to achieve the desired angle and remain there in equilibrium.

III. System

The first step is to define the system. The system has three parts. The first, is a platform (the dark blue checked platform). This is a Newtonian frame on which we will perform all controls (Bretl, 2021). The second and third parts are the gimbal (blue) and the rotor(orange)(Bretl, 2021). The gimbal and the rotor are mounted atop the platform. The gimbal and a rotor together constitute the control moment gyroscope.

The general idea of the system is that if the rotor is spun at a high rate, an "input torque" will be applied to the gimbal, which in turn will apply an "output torque" to the platform (Bretl, 2021). This output torque applied to the platform causes the platform to reorient and, hopefully move toward the direction/angle we desire. The output torque and the input torque are related through the conservation of angular momentum. The question that arises is, why are

use a control moment gyroscope. There are other ways of attaining a desired attitude, for example reaction wheels or momentum wheels. Reaction wheels work by spinning up or down and forcing the platform/body to move in a particular direction (Evans , 1967). Momentum wheels work by spinning at a very high rate in one direction, attaining stability and resistance to disturbance (Evans , 1967). Again, conservation of angular momentum is applied to the platform and since the angular momentum of the wheel is conserved, the angular momentum of the platform is also conserved. The third way, which we are discussing in this case is the control moment gyroscope. The benefit of the control moment gyroscope is that it is a combination of a reaction wheel and the momentum wheel. The rotor spins at a very high rate just like the reaction wheel, while the gimbal changes its orientation making it possible to do maneuvers just like the momentum wheel. The drawback of using a reaction wheel is that we only have control about the pitch axis (Evans , 1967). However that is bypassed in the control moment gyroscope because of the addition of the gimbal. However the drawback of using a controlled mum in general scope is that the controller is much more involved in complex than that of a reaction with. Therefore while it might be easier to develop a controller for the reaction wheel, efficiency is reduced whereas while the efficiency is higher for a controlled moment gyroscope controller, it is much harder to develop. For a purpose we care about inefficient process and as we use a controlled moment gyroscope. Some examples of where these three types of attitude control systems are used are the ISS(CMG), Hubble(momentum wheels), Kepler(reaction wheels) (Evans , 1967).

IV. Equations of motion

For the given system there will be a few equations of motion that need to be satisfied in order for the system to work perfectly. These equations of motion are differential equations involving the angle, angular velocity of the platform, gimbal, and the rotor as well as the torque applied by the platform with the gimbal and the gimbal to the rotor. The units of angles are radians (rad), the units for angular velocities are radians / second or (rad/s), and that for torque is newton metre or $(N.m)$.
The general notation we are using is as follows:
1) q_1 and v_1 are the angle and angular velocity of the platform
2) q_2 and v_2 are the angle and angular velocity of the gimbal
3) v_3 is the angular velocity of the rotor
4) τ_2 is the torque applied by the platform to the gimbal
5) τ_3 is the torque applied by the gimbal to the rotor
Figure 1 as shown below shows all the equations of motion that govern our system.

$$\dot{v}_1 = -\left(\frac{5(200\tau_3 \sin(q_2) + \sin(2q_2)v_1v_2 + 2\cos(q_2)v_2v_3)}{10\sin^2(q_2) - 511} \right)$$

$$\dot{v}_2 = \frac{10(100\tau_2 - \cos(q_2)v_1v_3)}{11}$$

$$\dot{v}_3 = -\left(\frac{51100\tau_3 + 5\sin(2q_2)v_2v_3 + 511\cos(q_2)v_1v_2}{10\sin^2(q_2) - 511} \right)$$

Fig. 1 Governing equations of motion

We can note that neither of the governing equations depend on either $q3$ or tau_3, and therefore it would seem that the system is independent of these parameters, making it possible for us to set them randomly.
Now that we have defined our system, it is time to linearise it.

V. Designing

A. Linearising system

It is useful to linearise our system into the state space from because that will give us information about the state and the input of our system. By general form of the state space model is shown below.

$$\dot{x} = Ax + Bu$$
$$y = Cx + Du$$

Fig. 2 State Space model

Here x is the state, u is the input, and y is the output. A, B, C, and D are matrices that help put the state and input in the state space model. In general if the state x is an n_x x 1 matrix, input u is an n_u x 1 matrix, and output y is an n_y x 1 matrix, then:

1) Matrix A is of size n_x x n_x
2) Matrix B is of size n_x x n_u
3) Matrix C is of size n_y x n_x
4) Matrix D is of size n_y x n_u

B. Forming matrix F

I have used Python to linearise the system. My general approach is as follows. I begin by defining my equations of motion shown in fig.1 by using the "sympy" module.

Defining state space model

```
# Creating required sympy symbols
# From EOMs
q1, q2, v1, v2, v3, tau2, tau3 = sym.symbols('q1, q2, v1, v2, v3, tau2, tau3')
```

```
# Defining EOMs given using the sympy symbols
v1_dot = -((5*(200*tau3*sym.sin(q2) + sym.sin(2*q2)*v1*v2 + 2*sym.cos(q2)*v2*v3))/(10*((sym.sin(q2))**2)-511))
v2_dot = 10*((100*tau2 - sym.cos(q2)*v1*v3)/11)
v3_dot = -((51100*tau3 + 5*sym.sin(2*q2)*v2*v3 + 511*sym.cos(q2)*v1*v2)/(10*((sym.sin(q2))**2) - 511))
```

Fig. 3 Sympy EOMS

I then converted this into a matrix F (fig.4) that involves v_1, $\dot{v_1}$, v_2, $\dot{v_2}$, and $\dot{v_3}$. It is clearly visible that this matrix is a 5 x 1 matrix. Therefore, $n_x = 5$

$$\begin{bmatrix} v_1 \\ -\dfrac{1000\tau_3 \sin(q_2) + 5v_1v_2 \sin(2q_2) + 10v_2v_3 \cos(q_2)}{10\sin^2(q_2) - 511} \\ v_2 \\ \dfrac{1000\tau_2}{11} - \dfrac{10v_1v_3 \cos(q_2)}{11} \\ -\dfrac{51100\tau_3 + 511v_1v_2 \cos(q_2) + 5v_2v_3 \sin(2q_2)}{10\sin^2(q_2) - 511} \end{bmatrix}$$

Fig. 4 Matrix F

VI. Implementing

A. Choosing Equilibrium point

The next step is to choose an equilibrium point. I have randomly chosen that I want my platform to point at $3\pi/4$ radians from its initial position. Since q_1 is the angle associated with the platform, q_{1_e} is the angle associated with the platform at equilibrium. Therefore I equate q_{1_e} to $3\pi/4$ radians. I also chose v_{3_e} to be 27 at random. This corresponds to the angular velocity of the rotor at equilibrium. For the rest of the equilibrium values I solved the governing differential equations and found suitable values that result in the matrix of being filled with zeros. My final choice of values are q_{2_e} = 0., v_{1_e} = 0., v_{2_e} = 0., tau_{2_e} = 0., tau_{3_e} = 0.

B. Finding A and B

My hypothesis was that at this equilibrium point, the matrix F will be an array of zeros. Indeed when I plug these values into the system/matrix F, I get the results as a matrix full of zeros. Therefore this is a valid equilibrium point. i then needed to find matrices A and B. To find these I found the Jacobians for A and B, and evaluated a lambda function at the equilibrium point.

My matrix for A:
$$\begin{bmatrix} 0. & 0. & 1. & 0. & 0. \\ 0. & 0. & 0. & 0.52837573 & 0. \\ 0. & 0. & 0. & 1. & 0. \\ 0. & 0. & -24.54545455 & 0. & -0. \\ 0. & 0. & 0. & 0. & 0. \end{bmatrix}$$

My matrix for B:
$$\begin{bmatrix} 0. & 0. \\ 0. & 0. \\ 0. & 0. \\ 90.90909091 & 0. \\ 0. & 100. \end{bmatrix}$$

Now I have everything I need to linearise my system. After linearising my system I must find suitable gains, also known as the "K matrix", that will make my system asymptotically stable. An asymptotically stable system is one that, as time passes the system will reach its desired angle/angular velocity and remained there for the entirety of the test. An indication that my controller will work so as to make my system asymptotically stable is that the eigenvalues of my gains have negative real parts and preferably zero imaginary parts.
I first formed my K matrix or appropriate size with random values. I then used the equation:

$$F = A - B@K$$

and subsequently,

$$eig = linalg.eigvals(F)$$

to compute the eigenvalues of F to find a suitable array/list of K matrix values that satisfy my conditions for the real and imaginary parts. One such array is:

$$\text{K} = \begin{bmatrix} 0.22739229019075757 & 0.8955567853864849 & 0.4178316692639851 & 0.6366058743800221 & 0.029898455546265912 \\ 0.90871561535014 & 0.39018343744587736 & 0.1922966489199195 & 0.4885937105342908 & 0.7139200609411928 \end{bmatrix}$$

with eigen values: $\begin{bmatrix} -77.62064257 + 0.j & -49.72556024 + 0.j & -1.73563253 + 0.j & -0.18343207 + 0.j & -0. + 0.j \end{bmatrix}$

C. Implementing controller

I have made changes to *CMGDemo.ipynb* code according to my designed controller. As discussed earlier, Our main goal is to get a resultant output torque that will move the platform to our desired angle, therefore the output of our code is τ_2 and τ_3, the torque applied by the platform to the gimbal and the gimbal to the rotor respectively.

VII. Results

My controller run for 10 seconds. During this time, I took values of the angle (outer angle), the angular velocity (outer angular velocity) of the platform and the dark applied to the gimbal and rotor as time varies and plotted it. I would know that my controller was successful if after 10 seconds I have reached my desired angle of the platform and stayed there, therefore confirming that my system is asymptotically stable. Along with this I also want the angular velocity of my platform after reaching equilibrium be and remain zero.

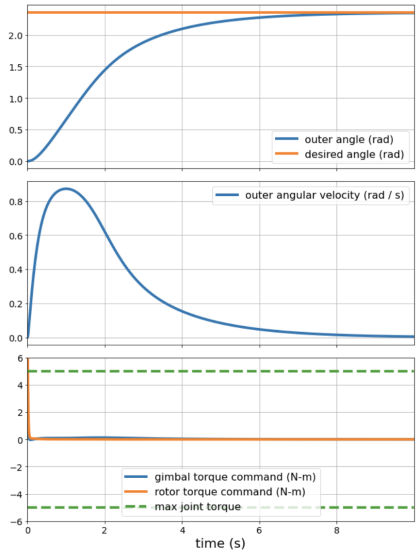

Fig. 5 Result plots

As expected, the outer angle approaches the desired outer angle and stays there, while the outer angular velocity increases, decreases and then goes to 0, meaning that the platform has reached equilibrium and is stable. In the third plot, we see that there is minor spike where the rotor torque exceeds the max joint torque but then soon comes back and stabilizes at zero.

VIII. Conclusion

This was a very interesting project for me because I got to learn, as well as implement my understanding of controllers to a real life application. In the process of figuring out how I should go about designing my controller, I realised that there are multiple ways to go about this and and reach my desired output. While this is a smaller implementation, this is a depiction of what is used in the ISS, something that really fascinates me. My journey has not ended here. I intend to go ahead with this quest of designing controllers that are asymptotically stable to try to design a controller than can also change the pitch of the platform.

Acknowledgments

Thanks to Professor Bretl for the pre-derived EOMs, CMGDemo.ipynb, along with all the reference material/notes on the AE353 2021 website, a lot of which helped me form my model. Thanks to Professor Bretl for his explanations/hints to go on when I struggled with finding appropriate gains. Thanks to Anshuk (anshukc2) for providing inspiration and direction for me through his code when I was stuck. I thank all my peers for asking doubts and clarifying concepts that I too faced difficulty in.

References

[1] Bretl, T., https://tbretl.github.io/ae353-sp21/projectsdesign-project-1

[2] Evans, H., "Reaction Wheels vs. Momentum wheels," Space Exploration Stack Exchange Available: https://space.stackexchange.com/questions/25658/reaction-wheels-vs-momentum-wheels: :text=A