DP1: Control Moment Gyroscope Control of a Spacecraft

Patrick D. Thornton and Varsha Krishnakumar
University of Illinois at Urbana-Champaign

The objective of this exploration is to implement and verify a feedback controller that uses a single-gimbal control moment gyroscope (CMG) to reorient and control the direction of a spacecraft platform in a gravitational field. The controller and simulation are implemented, designed, and tested in a Python Jupyter Notebook environment.

I. Nomenclature

CMG	=	Control Moment Gyroscope
ODE	=	Ordinary Differential Equation
f	=	Equations of motion as a function
q_1	=	Angle (*rad*) of platform
v_1	=	Angular Velocity (*rad/s*) of platform
q_2	=	Angle (*rad*) of gimbal
v_2	=	Angular velocity (*rad/s*) of gimbal
τ	=	Torque ($N \cdot m$) applied by the platform on the gimbal
u	=	Inputs for the system as a vector
x	=	State Vector

II. Introduction

Structure:

- Definition of a CMG and uses in spacecraft (attitude control, GNC)
- (General) Benefits and Disadvantages
- Single-Gimbal CMG definition (for this project)

The platform test angle for the intended design of the Single-Gimbal is noted as $q_{1_{des}} = 45°$. To design such a controller, a non-linear state space model of the system is initialized. The relevant testing and configuration of this model is performed in a python environment.*

III. Theory

A. Equations of Motion for CMG System

The CMG system is modeled based on a model in state-space form*. The motion of this particular CMG system is governed by the set of Ordinary Differential Equations below

$$\ddot{q}_1 = \frac{0.01\sin(2q_2)\dot{q}_1\dot{q}_2 + 20\cos(q_2)\dot{q}_2 - 39.24\sin(q_1)}{9.002 + 0.02\cos^2(q_2)} \tag{1a}$$

$$\ddot{q}_2 = -909.09\cos(q_2)\dot{q}_1 + 90.909\tau \tag{1b}$$

In order to design a proper controller, the system must be modeled in state-space form. The equations of motion must first be linearized in order to approximate the system with the desired state-space model. To do so, the system is first rewritten as a set of first-order ordinary differential equations.

*https://github.com/varshakrishnakumar/AE-353-Design-Project-1
*https://tbretl.github.io/ae353-sp22/projectsthe-system

$$\dot{x} = f = \begin{bmatrix} v_1 \\ \frac{0.01\sin(2q_2)v_1v_2+20\cos(q_2)v_2-39.24\sin(q_1)}{9.002+0.02\cos^2(q_2)} \\ v_2 \\ -909.09\cos(q_2)v_1 + 90.909\tau \end{bmatrix} \tag{2}$$

The equilibrium point for which to linearize the system about is the set of variables for f that cause $\dot{x} = 0$, and thus x remains constant.

1) $v_{1e} = v_{2e} = 0$ as is apparent from the first and third elements of $\dot{x}$.
2) As a result, $\tau_e = 0$ then becomes a requirement as well.
3) From the second element, it is important to note that the choice of q_{1e}, i.e., the desired angle at which to hold the platform at must be an integer multiple of π. In plain terms, this means that the spacecraft can only maintain stability in a completely vertical orientation. For now, a choice of $q_{1e} = \pi$ will be made.
4) q_{2e} may have any value. A reasonable choice is $q_{2e} = 0$.

The state and input can now be defined based on this choice of equilibrium point:

$$\mathrm{x} = \begin{bmatrix} q_1 - q_{1e} \\ v_1 - v_{1e} \\ q_2 - q_{2e} \\ v_2 - v_{2e} \end{bmatrix}$$

$$\mathrm{u} = \begin{bmatrix} \tau - \tau_e \end{bmatrix}$$

The matrices A and B can be consequently found by calculating the Jacobian of f with respect to the state vector (x) and the input vector (u) respectively. These are to be evaluated at the equilibrium points $q_{1e}, q_{2e}, v_{1e}, v_{2}e$.

$$A = \frac{\partial f}{\partial x} = \frac{\partial f}{\partial \begin{bmatrix} q_1 \\ v_1 \\ q_2 \\ v_2 \end{bmatrix}} \tag{4a}$$

$$B = \frac{\partial f}{\partial u} = \frac{\partial f}{\partial \begin{bmatrix} \tau \end{bmatrix}} \tag{4b}$$

The results of the A and B matrices, which was found through python implementation, are shown below.

$$\mathrm{A} = \begin{bmatrix} 0. & 1. & 0. & 0. \\ -4.34936821 & 0. & 0. & 2.21680337 \\ 0. & 0. & 0. & 1. \\ 0. & -909.09090909 & 0. & 0. \end{bmatrix}$$

$$\mathrm{B} = \begin{bmatrix} 0. \\ 0. \\ 0. \\ 90.90909091 \end{bmatrix}$$

B. Controller Analysis

The input u, is used as a function in a feedback loop for a closed-loop system. It is reliant on x, as well as constant matrix that will be introduced as K.

$$u = -Kx \tag{6}$$

Since u is now defined appropriately, the closed-loop system becomes:

$$\dot{x} = (A - BK)x \tag{7}$$

Where F can be defined as $F = A - BK$, yielding

$$\dot{x} = Fx \tag{8}$$

With the initial condition

$$x(t_0) = x_0, \tag{9}$$

the solution to the prior equation can be written as

$$x(t) = e^{Ft}x_0 \tag{10}$$

The linear system must be asymptotically stable in order for the single-gimbal CMG system to have smooth behavior as desired. For this to be achieved, $x(t)$ must asymptotically approach 0. Accordingly, all the eigenvalues of F must be negative real numbers. The procedure for determine a valid K matrix currently involves randomizing elements of K until the eigenvalues of F are all negative real values. It can further be noted that with lower values of K, there is a higher plausibility for constraining the desired torque applied on the gimbal by the platform. Furthermore, this can result in more efficient motion of the platform. Correspondingly, the choice of K for this process is shown below:

$$K = \begin{bmatrix} 2. & -2. & 2. & 0.18 \end{bmatrix} \tag{11}$$

It is apparent that not all valid K matrices yield the desired CMG behavior, so this process will be constrained further. The next appropriate step is to efficiently constrain the time frame for the system achieving a desirable platform angle. With this, the controller will be featured as computationally and time efficient. To achieve this, an analysis on its response, $x(t)$ must be performed. Here, the matrix F, as found previously, must be diagonalized so that evaluating the matrix exponential, e^{Ft}, is simplified.

The relation used for this process if as follows:

$$x(t) = Ve^{St}V^{-1}x(0) \tag{12}$$

Here, V is a matrix comprised of the eigenvectors of F, and S is a diagonal matrix with F's eigenvalues. The corresponding results of the analysis on the controller's response is shown in *Figure 1*.

IV. Experimental Methods

Jupyter Notebook was utilized as code interface for simulating and testing the CMG model. The instructions for compilation of the project code is detailed in the GitHub repository associated with this report *.

Provided a matrix of controller gains, **K**, that in theory results in a controller that produces an asymptotically stable CMG system, **K** is implemented in the closed-loop system and simulation trials can then be conducted.

To test the stability of the controller in achieving and maintaining the desired platform angle, the following parameters are set:

1) q_{1e} is set to the desired platform angle of π rad. This is a completely vertical position where the platform's load mass is held above the platform.
2) All other equilibrium variables are set to a value of 0 as discussed prior.
3) The initial platform angle q_{1i} at time $t = 0$ is set to a value of $\pi - 0.2$ rad.
4) The initial platform velocity v_{1i} is set to a value of $-0.2\frac{rad}{s}$.
5) The other initial variables q_{2i} and v_{2i} are set to a value of 0, so that the rotor wheel is initially flush with the platform.
6) The gains matrix is set to $K = \begin{bmatrix} 2. & -2. & 2. & 0.18 \end{bmatrix}$

*https://github.com/varshakrishnakumar/AE-353-Design-Project-1

Under these conditions, the controller code is executed, and a visual demonstration and plots are output to allow for analysis of the result.

To test the limits of initial conditions for which the controller can achieve stability, further trials were run with varying values for q_{1i} and v_{1i}, $[q_{1i}, v_{1i}]$:

1) Trial 1: [0.5 *rad*, 0 *rad/s*]
2) Trial 2: [$\pi - 0.1$ *rad*, 0.1 *rad/s*]
3) Trial 3: [$\pi - 0.3$ *rad*, 0.2 *rad/s*]
4) Trial 4: [$\pi - 0.5$ *rad*, 0.3 *rad/s*]
5) Trial 5: [$\pi - 0.7$ *rad*, 0.4 *rad/s*]

V. Results

The results of the single-CMG simulation are shown in *Figure 1*.

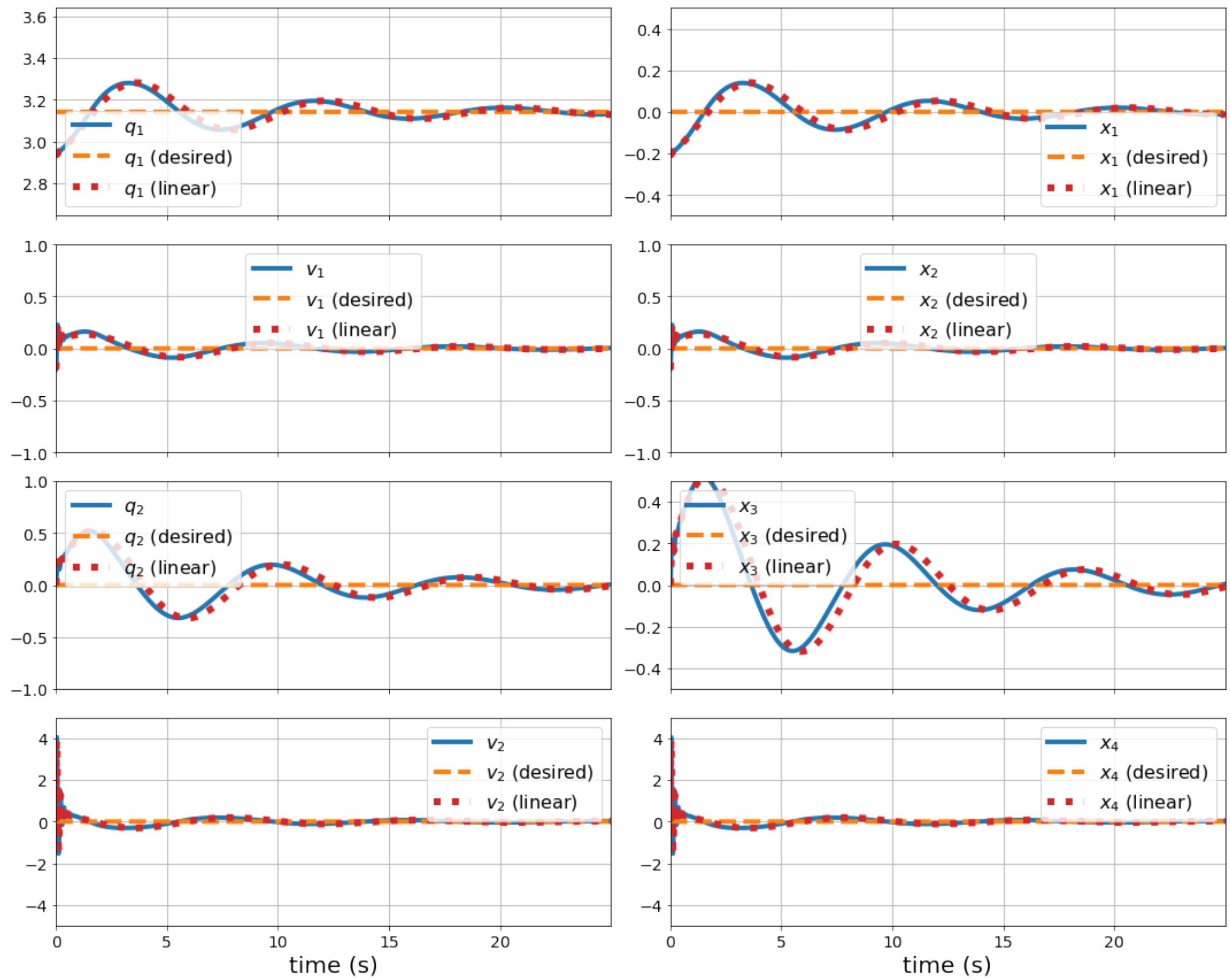

Fig. 1 Results of Simulation

The most critical outcomes from this simulation and corresponding testing are:

1) For the initial condition of $q_{1_i} = \pi - 0.2$ and an equilibrium condition of $q_{1_e} = \pi$, the platform angle converges to the desired angle of π within 20 seconds. This indicates the stability of the controller, as well as the efficiency of the model for desired angles close to the equilibrium condition.
2) Similarly, the desired angular velocity of the platform, $v_1 = 0$, is achieved within 13 seconds.
3) For the initial condition of $q_{2_i} = 0$ and an equilibrium condition of $q_{2_e} = 0$, the gimbal angle converges to the desired angle of 0 within 20 seconds.
4) Similarly, the desired angular velocity of the gimbal, $v_2 = 0$, is achieved within 8 seconds.

The results for the other five conducted trials of varying initial platform and angular velocity are detailed in the table below.

As seen from the results, the controller is efficient is stabilizing for initial conditions that are close to the equilibrium or desired conditions. For values further away from the equilibrium conditions, the controller is slow to stabilize, or does not achieve stability at all. This could be a result of the **K** matrix consisting of smaller values.

Overall, the system has been successful at achieving the objective of rotating the platform to the desired platform angle, given that the initial conditions are within 1.0 *rad* of the equilibrium condition, $q_{1e} = \pi$.

Initial Platform Angle	Initial Platform Velocity	Final Platform Angle	Final Platform Velocity	Stability Achieved?
$q_{1_i} = 0.5$	$v_{1_i} = 0.1$	q_1 approaches infinity	v_1 oscillates from -200 to 200 *rad/s*	no
$q_{1_i} = \pi - 0.1$	$v_{1_i} = 0.2$	$q_1 = \pi$	$v_1 = 0$	yes
$q_{1_i} = \pi - 0.3$	$v_{1_i} = 0.3$	$q_1 = \pi$	$v_1 = 0$	yes
$q_{1_i} = \pi - 0.5$	$v_{1_i} = 0.4$	$q_1 = \pi$	$v_1 = 0$	yes
$q_{1_i} = \pi - 0.7$	$v_{1_i} = 0.5$	$q_1 = \pi$	$v_1 = 0$	yes

Table 1 Simulation Results for Varying Initial Conditions

VI. Conclusion

Summarize key conclusions and identify ways that others could improve or build upon your work

Acknowledgments

- Prof. Timothy Bretl & Mr. Jacob Kraft

References

Appendix

Day	**Task**	**Person or People**
01/28/2022	Initiation and set-up of report document	Patrick Thornton
02/01/2022	Basic structure for Abstract and Introduction sections	Varsha Krishnakumar
02/02/2022	Creation of GitHub repository with basic outline of model generating code in python environment	Varsha Krishnakumar
02/03/2022	Description of Equation of Motion for the CMG system	Varsha Krishnakumar
02/03/2022	Description of method for finding equilibrium points	Patrick Thornton
02/04/2022	Exposition of theory and model with python code for the controller: worked together on analyzing faults in the system's motion and possible reasons for inefficient working	Patrick Thornton & Varsha Krishnakumar
02/09/2022	Analyzed, debugged, formatted, and annotated existing code.	Patrick Thornton
02/09/2022	Completed working code, implemented plots to visualize state of the system to determine if stability is achieved.	Patrick Thornton
02/09/2022	Adapted the in-class interactive eigenvalue plot notebook for use with the CMG system, devised an improved system for determining the K matrix, and conducted testing for various K matrices.	Patrick Thornton
02/10/2022	Conducted testing of controller for various initial conditions	Patrick Thornton & Varsha Krishnakumar
02/11/2022	Finalized *Experimental Methods* and *Results* sections.	Patrick Thornton & Varsha Krishnakumar
02/11/2022	Improved report formatting and made adjustments based on comments	Patrick Thornton & Varsha Krishnakumar