Name: Usman Usman

ID: 40174744

ELEC-372 Project

Due date: November 19th, 2023.

# Abstract

This project entails designing a Satellite pitch control system in a geostationary orbit using MATLAB control pkg to select the right micro-controller that meets our design specification. I was provided some specifications for the design of the pitch control system, see below.

* DS1: The closed–loop system must be stable.
* DS2: The settling time of response to step inputs must be less than 300s.
* DS3: The percentage of overshoot of response to step inputs must be less than 25%.
* DS4: The pitch angle accuracy must be at least 0.05 deg; that is the effect of disturbance on the output in steady state should be less than 0.05 deg.
* DS5: As with other satellite subsystems, the ADCS is subject to a power budget. Hence it is desirable to minimize the power consumption of the ADCS (i.e., the sum of power consumptions of the actuators, sensors, and processing electronics).

Therefore, I was provided a selection of three controllers to choose from that satisfies the design specifications listed below.

* =
* =
* =

Using analytical calculations and computer simulations such as MATLAB, I selected the right controller that meets all the Design specifications.

# Introduction

A Satellite exhibiting circular geostationary orbit entails a circular orbit at an altitude of approximately 36,000 km while the period of the earths rotation and the satellite have been equal [1]. Therefore, the angular velocity of the satellite in its orbit is 7.28 e-5 rad/s [1].

In this project, I sought to design a control system aimed to stabilize the pitch angle using the linearized model and control system shown below.

A diagram of a block diagram

Description automatically generated

Fig 1: Pitch control system

# Methodology

## Graphs

Using MATLAB control system toolbox, I produced four graphs for each controller.

A graph with a line

Description automatically generated

Fig1: The output θ(*t*) in response to a step reference input of 5 deg = 5*π/*180 rad for k\_1 (s)

A graph of a curve

Description automatically generated

Fig2: The output θ(*t*) in response to a step reference input of 5 deg = 5*π/*180 rad for

A graph with a line

Description automatically generated

Fig3: The output θ(*t*) in response to a step reference input of 5 deg = 5*π/*180 rad for

A graph with a blue line

Description automatically generated

Fig4: The control torque *Tc*(*t*) in response to a step reference input of 5 deg = 5*π/*180 rad for

A graph with a line

Description automatically generated

Fig5: The control torque Tc(t) in response to a step reference input of 5 deg = 5π/180 rad for

A graph with numbers and lines

Description automatically generated

Fig6: The control torque Tc(t) in response to a step reference input of 5 deg = 5π/180 rad for

A graph with a line

Description automatically generated

Fig7: The output θ(t) in response to disturbance Td(*t*) for

A graph of a function

Description automatically generated

Fig8: The output θ(t) in response to disturbance Td(*t*) for

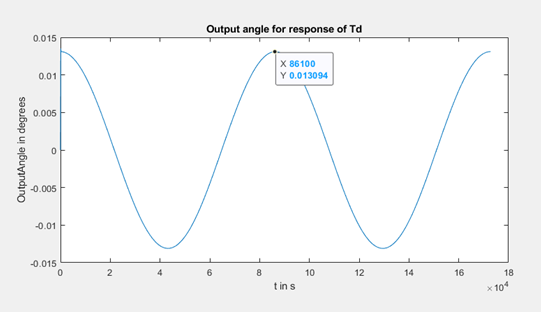


Fig9: The output θ(t) in response to disturbance Td(*t*) for

## DS1: The closed–loop system must be stable.

* For =

I = 400N.m.s^2

=

Is3 + 0.1s +0.01

Using the Routh’s table to determine stability:

|  |  |  |  |
| --- | --- | --- | --- |
| S3 | I | 0.1 | 0 |
| S2 | E | 0.01 | 0 |
| S1 | (0.1E-0.01I)/E | 0 | 0 |
| S0 | 0.01 |  |  |

(4E-10)/E =-

Therefore*, it’s instable* due *to the changes in sign.*

* For =

=

*Using the Routh’s table to determine stability:*

|  |  |  |  |
| --- | --- | --- | --- |
| S3 | 80I | 240 | 0 |
| S2 | 20I | 30 | 0 |
| S1 | 120 | 0 | 0 |
| S0 | 30 |  |  |

From the rout’s table we can determine that the closed system is stable because there is no sign change from one row to another.

* For =

G(s) =

=

Using the Routh’s table to determine stability:

|  |  |  |  |
| --- | --- | --- | --- |
| S3 | 40I | 1400 | 0 |
| S2 | 80I | 35 | 0 |
| S1 | 2765/2 | 0 | 0 |
| S0 | 35 |  |  |

From the rout’s table we can determine that the closed system is stable because there is no sign change from one row to another.

From using the rout table analysis, I determined that controllers 2 and 3 will provide a stable closed system loop therefore these controllers will go through the next step of testing.

## DS2: The settling time of response to step inputs must be less than 300s.

* For =

I = 400N.m.s^2

G(s) =

Td(s) =

Using Matlab codes:

K = tf([0.1 0.01],[1 0]);

I = 400;

G = tf(1,[I 0 0]);

W0= 7.28\*10^(-5);

SqW0= W0^(2);

Tds= 10^(-4)\*tf([1, 0],[1, 0, SqW0]);

SqW0= W0^(2);

Taddk = K+Tds;

srs = series (Taddk,G);

trf = feedback(srs,1);

poles = pole(trf)

:

A black and white image of numbers and lines

Description automatically generated

Poles:

1.325326104956351e-02 + 2.787376587485687e-02i

1.325326104956351e-02 - 2.787376587485687e-02i

-2.650652157959694e-02 + 0i

-2.597650259436380e-10 + 7.243870763312508e-05i

-2.597650259436380e-10 - 7.243870763312508e-05i

Since 2.650652157959694e-02 > 10 \*2.597650259436380e-10, therefore s2 and s3 are the dominant poles.

Setting time = 4/ζwn = 1.539e10 seconds

The setting time is greater than 300 seconds therefore it fails this design specification.

* For =

G(s) =

Td(s) =

Using Matlab codes:

K = tf([240 3],[80 20]);

I = 400;

G = tf(1,[I 0 0]);

W0= 7.28\*10^(-5);

SqW0= W0^(2);

Tds= 10^(-4)\*tf([1, 0],[1, 0, SqW0]);

SqW0= W0^(2);

Taddk = K+Tds;

srs = series (Taddk,G);

trf = feedback(srs,1)

poles = pole(trf);

:

A math equations with numbers and a line

Description automatically generated with medium confidence

I = 400N.m.s2

Poles:

-0.2175 + 0i

-0.0159 + 0.0125i

-0.0159 - 0.0125i

-0.0007 + 0i

-0.0000 + 0i

Since 0.2175 > 10 \*0.0159, therefore s2 and s3 are the dominant poles.

Setting time = 4/ζwn = 251.57seconds

The setting time is less than 300 seconds therefore it passes this design specification.

* For =

G(s) =

Td(s) =

:

A black and white image of numbers and symbols

Description automatically generated

Poles:

-1.9558 + 0i

-0.0220 + 0.0250i

-0.0220 - 0.0250i

-0.0002 + 0i

-0.0000 + 0i

Since 1.95583 > 10 \* 0.02208, therefore s2 and s3 are the dominant poles.

Setting time = 4/ζwn = 181.159 seconds

The setting time is less than 300 seconds therefore it passes this design specification.

## DS3: The percentage of overshoot of response to step inputs must be less than 25%.

* For =

0Percentage overshoot =e^( -) = 0.0189 =1.89 percent

The percentage overshoot is greater than 25 percent therefore it fails this design specification.

* For =

G(s) =

=

16000s3 + 32000s2 + 1400s +35 = 0

To find the poles

S1,2,3 =-1.95583, -0.02208 +0.02512j and -0.02208 - 0.02512j

Percentage overshoot =e^( -) = 0.0632 = 6.32 percent

The percentage overshoot is less than 25 percent therefore it passes this design specification.

## DS4:

## The graphs revealed that the disturbance's impact on the output in K1 exceeded 0.5, whereas the other controllers remained below this threshold. Consequently, K2 and K3 meet the DS4 design specification, whereas K1 does not.

## DS5:

In accordance with this design specification, I analyzed the graph depicting the output θ(t) in response to a step reference input of 5 degrees (equivalent to 5π/180 radians) as shown in Fig1, 2, and 3. I computed the average distance over a time interval from 0 to 172,800 seconds. Given that distance is directly proportional to work done, a greater distance traveled in terms of θ(t) implies a higher power requirement to manipulate the control wheel. The MATLAB code presented below facilitated the calculation of the total distance covered with each controller, as outlined further.

function distance\_total = distance(a,b,c,d) %k = (as+b)/(cs+d)

W0= 7.28\*10^(-5);%In this paragraph, constants are defined.

SqW0= W0^(2);

I=400;

t1=0:100:172800;

Ks= tf([a b],[c d]); %In this paragraph,

Tds= 10^(-4)\*tf([1, 0],[1, 0, SqW0]); %individual blocks are defined.

Gs=tf(1,[I,0,0]);

TdsplusKs = Ks + Tds; %In th

OLTs = series(TdsplusKs,Gs);

CLTsGr1= feedback(OLTs,1);

ref\_in=((5\*pi)/180);

y1= ((ref\_in\*step( CLTsGr1,t1))\*180)/pi;

y1Pos=abs(y1)

plot(t1,y1)

subplot(221);

xlabel('t in s');

ylabel('y1 in degrees');

title('Output angle y1 for step response of 5 deg');

xlabel('t in s');

ylabel('y1 in degrees');

title('Output angle y1Pos for step response of 5 deg');

plot(t1,y1Pos)

distance\_total = sum(y1Pos)/(172800/100);

end

* For =

It has a very high number due to its instability therefore it fails this design specification.

* For =

The average distance travelled using the K2 controller is 5.0008.

* For =

The average distance travelled using the K3 controller is 5.0004.

# Final Decision:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Controller | DS1(Pass/Fail) | DS2(Pass/Fail) | DS3(Pass/Fail) | DS4(Pass/Fail) | DS5(avg distance) |
| K1 | Fail | Fail | Fail | Fail | NA |
| K2 | Pass | Pass | Pass | Pass | 5.0008 |
| K3 | Pass | Pass | Pass | Pass | 5.0004 |

Table1: Decision table

From the decision table, K1 fails all the design specifications while K2 and K3 pass all the design specifications. I also noticed that the K2 controller had a higher average distance compared toK3 therefore, the K3 controller has less power consumption therefore K3 is the best fit for the project.

# Analysis Limitation

The omission of the thruster misalignment torque disturbance in the analysis reduces the accuracy of the simulation to real world application. Additionally, the analysis is constrained due to presence of round-off errors within the MATLAB simulation system.

# Reference

1. https://moodle.concordia.ca/moodle/pluginfile.php/6214939/mod\_resource/content/2/project-statement.pdf