Space Telescope Control System

Concordia Department of Engineering and Computer Science

Class: SOEN 385
For: Dr. Javad Sadri
Date: April 16, 2018

Team members:
Piratheeban Annamalai - 27755708
Hoang Khang Nguyen - 27079427
Andrew Laramee - 27050925

Abstract

Space telescopes need to be able to pivot on several axis in order to be efficient and useful. In this project we built a control system for a space telescope that has a range spanning the entire upper hemisphere from the ground up with the telescope at its origin. From the main user interface, we are able to give input parameters of Phi and Theta which symbolize spherical coordinates. These inputs are processed and used to denote the direction in which the telescope should point. We are also able to input different values for P (proportional), I (integral) and D (derivative) controllers in order to provide further control over the telescope's movement. Three different control schemas are selectable from the user interface, each with its own slight differences that are to be discussed in detail in this report.

Table of Contents

Table of Figures

Introduction

Our space telescope control system was built with the intent to take photos of one hundred different stars at different positions in the sky with a speed of one photo every five seconds. For this project, Matlab was used to generate the scripts and the user interface, and Simulink was used to model and design the control system. The control system uses two separate distinct motors, one controlling rotation in a 360 degree arc flush with ground level and the other controlling elevation level in a 180 degree arc. There are three different schemas with which the space telescope can be controlled. Each schema has a different transfer function for the motors G(s), as well as different transfer functions for the sensors H(s).

The first design has a second order motor transfer function of G(s) = 1/(S-1)(S-3) and a unity feedback system. The second design has a second order motor transfer function of G(s) = 1/(S+1)(S-3) with a sensor transfer function of H(s) = 0.1. The third design has a first order motor transfer function of G(s) = 1/(S-3) and a sensor transfer function of H(s) = 0.1/(S+1).

Design Methodology

For each question, we were given a force function and a feedback function. Together they would affect the delay for the system to adjust itself. Each question also has a controller whose values can be modified in order to alternate between P, PI, PD and PID controller. However, the entire process would only be applied to a single input value.

The given data was a list of 3 value vectors; Theta, Phi and Rho. We could ignore Rho as it represented to distance between the central rotational point to the star coordinate, thus leaving us with 2 datasets to work with. WIth that in mind, we repeated the above process for the 2 dataset individually and merged them together afterwards into a single 3 value vector [Figure 1]. The animation would then render based on the vector fed to it.

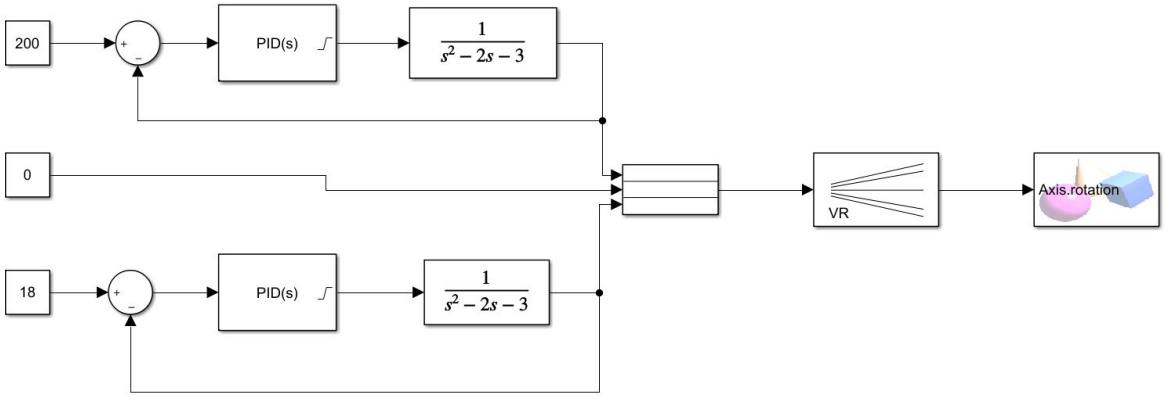

Figure 1: Simulink Diagram example for Question A

Implementation Details

We used the set_param function in Matlab to allow users to change the values of P, I, and D. If the input is set to 0, it means that the component is not used. For example if P and I have values other than 0 and D is set to 0, then the user wants to look at the telescope with the PI controller.

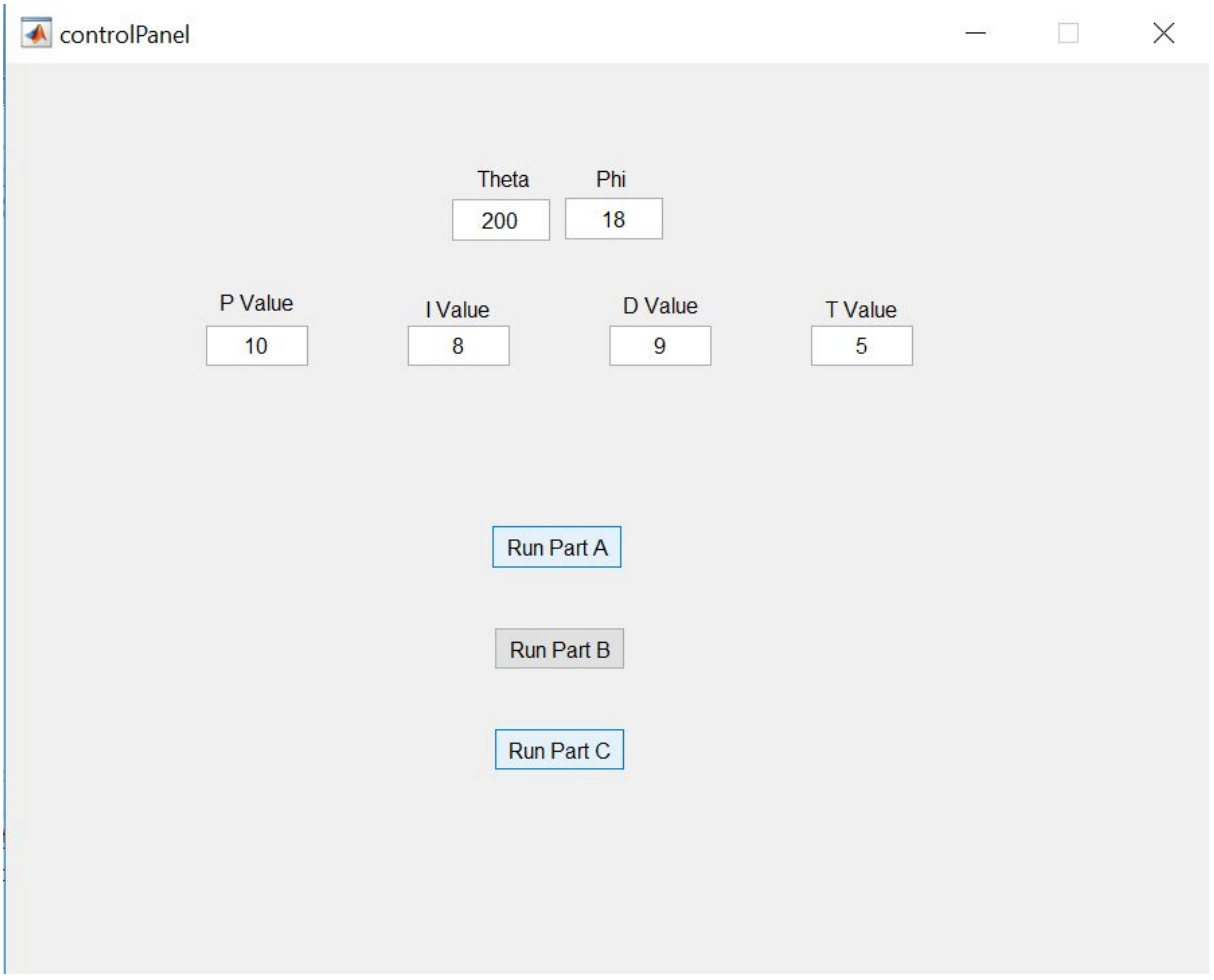

Figure 2: GUI so users can choose different values for P,I,D,T and Theta and Phi. Users can run each question (A, B, C).

Furthermore, setting the values for Theta and Phi will run set_params and change the values of the constant blocks within the simulink diagram [Figure

3]. This allows the user to run multiple attempts for different Theta and Phi values.

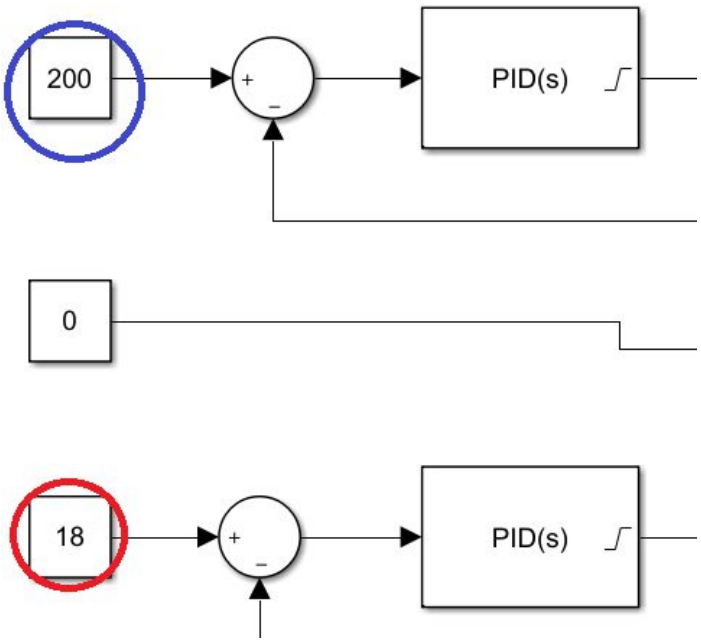

Figure 3: Constant block's value are set to equal to the input values. The block circled in blue is the Theta value and the red is the Phi value.

In each question, the input values will pass through the system with different functions respectively depending on the user's selection. From the GUI [Figure 2], the user will select the Part to run ranging from A, B or C. Each "part" has its own respective function seen in the example below [Figure 1, 4, 5].

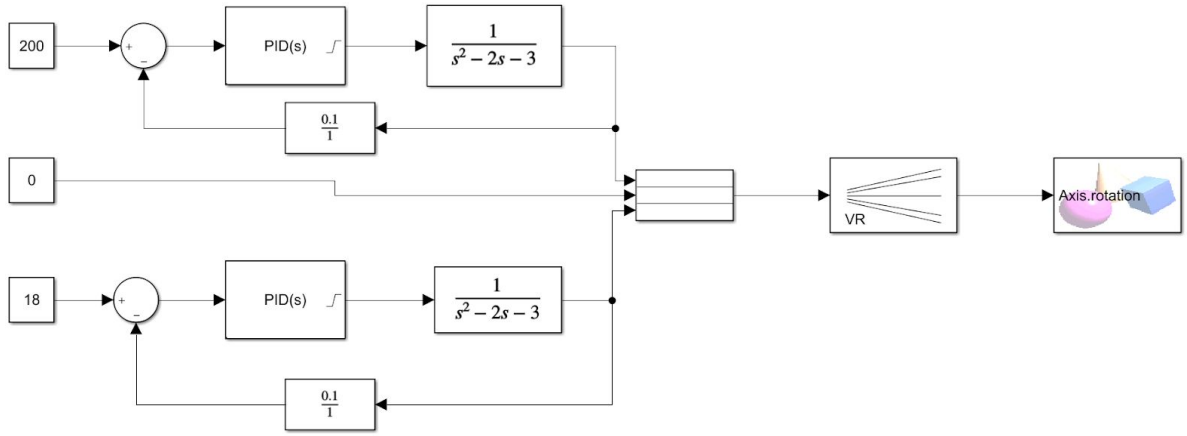

Figure 4: Question B, notice the response function

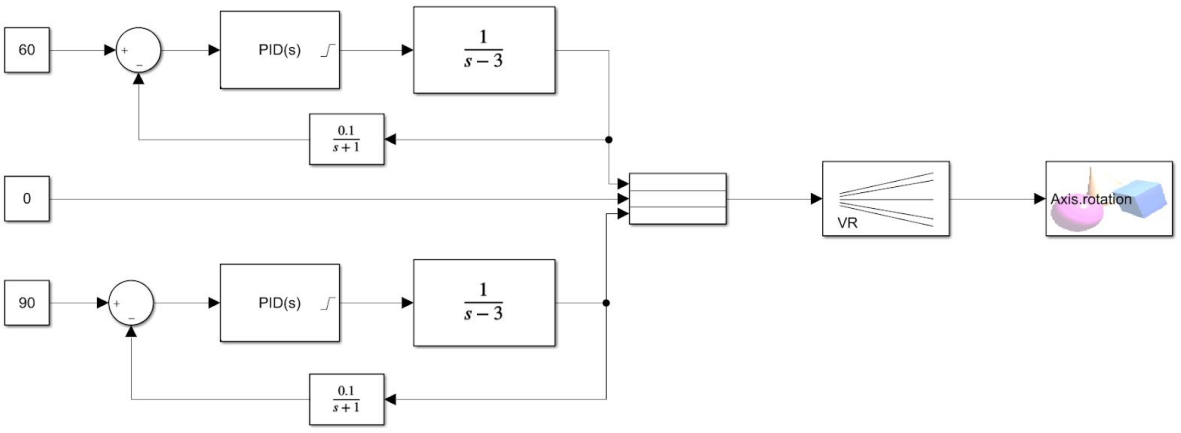

Figure 5: Question C

The final goal is to pass a vector of 4 values to the VR sink which will render our simulation. In order to animate, we fed that vector to the Axis rotation field of the VR sink which accepts a vector of 4 values [Figure 6].

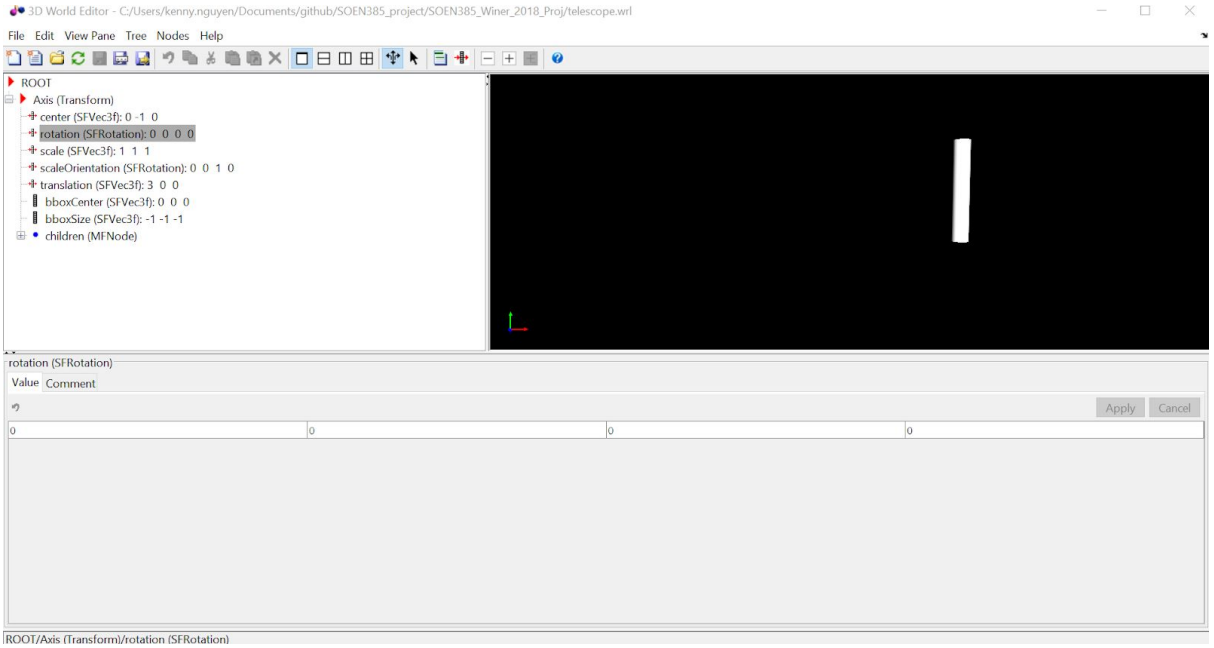

FIgure 6: VR edit mode, here we can notice at the bottom section that there are 4 values for the rotation attribute of the object.

Simulations and Experiments

Our control system involves a PID controller, which was provided to us by Simulink and this consists of 3 components:

1) The proportional (P) component reduces the rise time but will never be able to eliminate the steady-state error.
2) The integral (I) component can eliminate the steady-state error for either a constant or step input. However, reaching this point may cause the transient response to be slower.
3) The derivative (D) component can increase the stability of our control system while reducing the overshoot and making the transient response faster.

With the 3 components mentioned above, we can have 4 different controllers: PID, PI, PD, and P. These controllers are all affected by the time-delay factor (T).

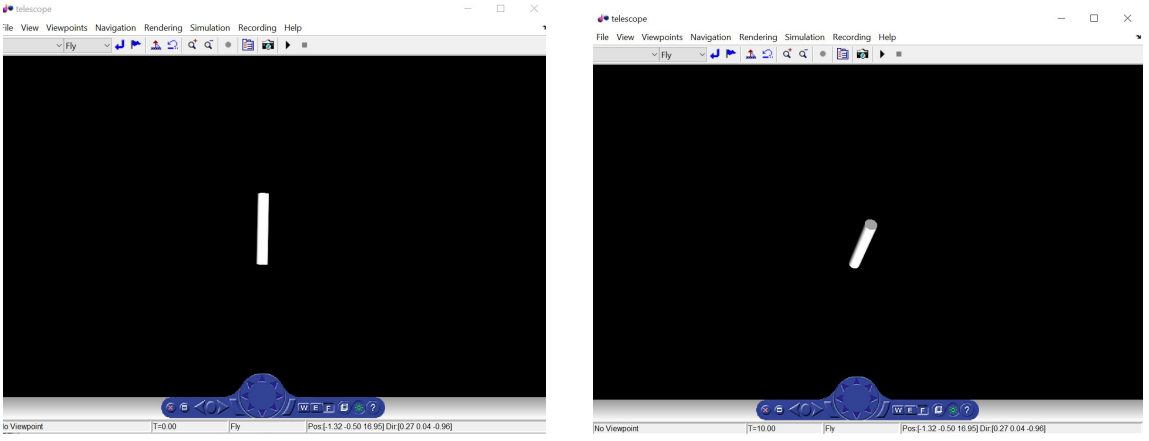

Figure 7: Question A before running telescope Question A after running telescope

Our space telescope control system was tested and it is clear that it works most effectively with PID and PD controllers.

Conclusion

In conclusion, we had a good time working on this project and it was a great learning experience. We learned a lot about how control systems work and how they can be used and implemented in the real world. This project allowed us to understand the class material a lot better because we were able to observe the patterns of a control system.

References

[1] Forum page, Matlab focused, http://www.edaboard.com/

[2] Matlab page, https://www.mathworks.com/

[3] PDF files provided by Dr. Javad Sadri.