Ann and H.J. Smead
Aerospace Engineering Sciences
UNIVERSITY OF COLORADO **BOULDER**

ASEN 5014 Spacecraft Orbit Control Project Final Report

Brady Sivey
Jared Steffen

December 2, 2025

Author Contributions:

Brady Sivey

- Part A question 1
- Worked with Jared to determine reasonable control objectives for stability augmentation, reference input tracking, and disturbance rejection/cancellation.
- Luenberger Observer Design and write up (question 5)
- Worked with Jared to implement and tune LQR controller.

Jared Steffen

- Worked with Brady to determine reasonable control objectives for stability augmentation, reference input tracking, and disturbance rejection/cancellation.
- Simulated the open loop response of the system and determined the reachability and observability of the system and the bases for the corresponding subspaces.
- Implemented the manual pole placement controller designs.
- Worked with Brady to implement and tune the LQR controller.

1 Introduction

Spacecraft missions often adjust orbital parameters for maintaining a desired orbit in the presence of disturbance forces, orbit injection errors, or to complete mission objectives. This process is done via an on-board control law that utilizes thrusters to change orbital parameters through delta-V maneuvers. This project aims to investigate spacecraft orbit control utilizing low-thrust ion thrusters that implement stability augmentation, reference tracking, and disturbance rejection through the use of different control system designs.

2 Part A: Linear System Analysis

2.1 Question 1: Linear System Description

The dynamical system being analyzed is a spacecraft in orbit around the Earth. The non-linear state elements are represented by $[\, r \;\; \dot{r} \;\; \theta \;\; \dot{\theta} \,]^T$ where r is the distance from the center of the Earth to the spacecraft and θ is the angle with respect to the reference direction in the orbital plane. The non-linear control inputs for this system are defined as u_1 and u_2. These are control accelerations produced by the thrusters.

This analysis is done by assuming the spacecraft to be a point mass which obeys asimple inverse square law. The mathematical state space model was derived through eqs. (1) and (2). This system is then linearized around the nominal trajectory

$$[r_0, 0, 0, \sqrt{\frac{\mu}{r_0^3}}]$$

Where $r_0 = 6678\ km$ and $\mu = 398,600\ \frac{km^3}{s^2}$. This was done by taking the partial derivatives of eqs. (1) and (2) to produce the Jacobian matrices as seen in eq. (3). The A, B, C, and D matrices in eq. (4) were then produced by plugging the nominal values into the computed Jacobians.

$$\ddot{r} = r\dot{\theta}^2 - \frac{\mu}{r^2} + u_1 \tag{1}$$

$$\ddot{\theta} = -\frac{2\dot{\theta}\dot{r}}{r} + \frac{1}{r}u_2 \tag{2}$$

$$\tilde{A} = \frac{\partial f}{\partial x} = \begin{bmatrix} \frac{\partial \dot{r}}{\partial r} & \frac{\partial \dot{r}}{\partial \dot{r}} & \frac{\partial \dot{r}}{\partial \theta} & \frac{\partial \dot{r}}{\partial \dot{\theta}} \\ \frac{\partial \ddot{r}}{\partial r} & \frac{\partial \ddot{r}}{\partial \dot{r}} & \frac{\partial \ddot{r}}{\partial \theta} & \frac{\partial \ddot{r}}{\partial \dot{\theta}} \\ \frac{\partial \dot{\theta}}{\partial r} & \frac{\partial \dot{\theta}}{\partial \dot{r}} & \frac{\partial \dot{\theta}}{\partial \theta} & \frac{\partial \dot{\theta}}{\partial \dot{\theta}} \\ \frac{\partial \ddot{\theta}}{\partial r} & \frac{\partial \ddot{\theta}}{\partial \dot{r}} & \frac{\partial \ddot{\theta}}{\partial \theta} & \frac{\partial \ddot{\theta}}{\partial \dot{\theta}} \end{bmatrix}$$

$$\tilde{B} = \frac{\partial f}{\partial u} = \begin{bmatrix} \frac{\partial \dot{r}}{\partial u_1} & \frac{\partial \dot{r}}{\partial u_2} \\ \frac{\partial \ddot{r}}{\partial u_1} & \frac{\partial \ddot{r}}{\partial u_2} \\ \frac{\partial \dot{\theta}}{\partial u_1} & \frac{\partial \dot{\theta}}{\partial u_2} \\ \frac{\partial \ddot{\theta}}{\partial u_1} & \frac{\partial \ddot{\theta}}{\partial u_2} \end{bmatrix} \tag{3}$$

$$\tilde{C} = \frac{\partial y}{\partial x} = \begin{bmatrix} \frac{\partial(\delta r)}{\partial r} & \frac{\partial(\delta r)}{\partial \dot{r}} & \frac{\partial(\delta r)}{\partial \theta} & \frac{\partial(\delta r)}{\partial \dot{\theta}} \\ \frac{\partial(\delta \theta)}{\partial r} & \frac{\partial(\delta \theta)}{\partial \dot{r}} & \frac{\partial(\delta \theta)}{\partial \theta} & \frac{\partial(\delta \theta)}{\partial \dot{\theta}} \end{bmatrix}$$

$$\tilde{D} = \frac{\partial y}{\partial u} = \begin{bmatrix} \frac{\partial(\delta r)}{\partial u_1} & \frac{\partial(\delta r)}{\partial u_2} \\ \frac{\partial(\delta \theta)}{\partial u_1} & \frac{\partial(\delta \theta)}{\partial u_2} \end{bmatrix}$$

$$\tilde{A} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ \frac{3\mu}{r_0^3} & 0 & 0 & 2\sqrt{\frac{\mu}{r_0^3}} \\ 0 & 0 & 0 & 1 \\ 0 & -2\sqrt{\frac{\mu}{r_0^3}} & 0 & 0 \end{bmatrix}, \tilde{B} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & \frac{1}{r_0} \end{bmatrix}, \tilde{C} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \tilde{D} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{4}$$

After linearization, the state elements are represented by $[\,\delta r \;\; \delta \dot{r} \;\; \delta \theta \;\; \delta \dot{\theta}\,]^T$. The linearized control inputs are defined as $[\,\delta u_1(t) \;\; \delta u_2(t)\,]^T$. Finally, the linearized sensed outputs are $[\,\delta r \;\; \delta \theta\,]^T$. These are all small perturbations about the nominal trajectory.

Ultimately, the purpose of this system is to maintain a desired orbit in the presence of small disturbances. This will be done by applying control accelerations through the thrusters. Because this model is linearized, it is only suitable for small deviations from the nominal orbit. This means that the model neglects any higher order terms from assuming a point mass and neglecting other dynamical elements such as SRP, Earth oblateness, etc. There is also no process or measurement noise considered in this analysis.

2.2 Question 2: Control System Objectives and Open Loop Response

When looking at the linearized system above, we see that the open-loop poles of the system are $\lambda_{1,2} = 0$ and $\lambda_{3,4} = 0 \pm 0.0012i$. This means that the natural system response is marginally stable. One of our control objectives is to make the system asymptotically stable by having the closed loop poles in the left-hand side of the complex plane.

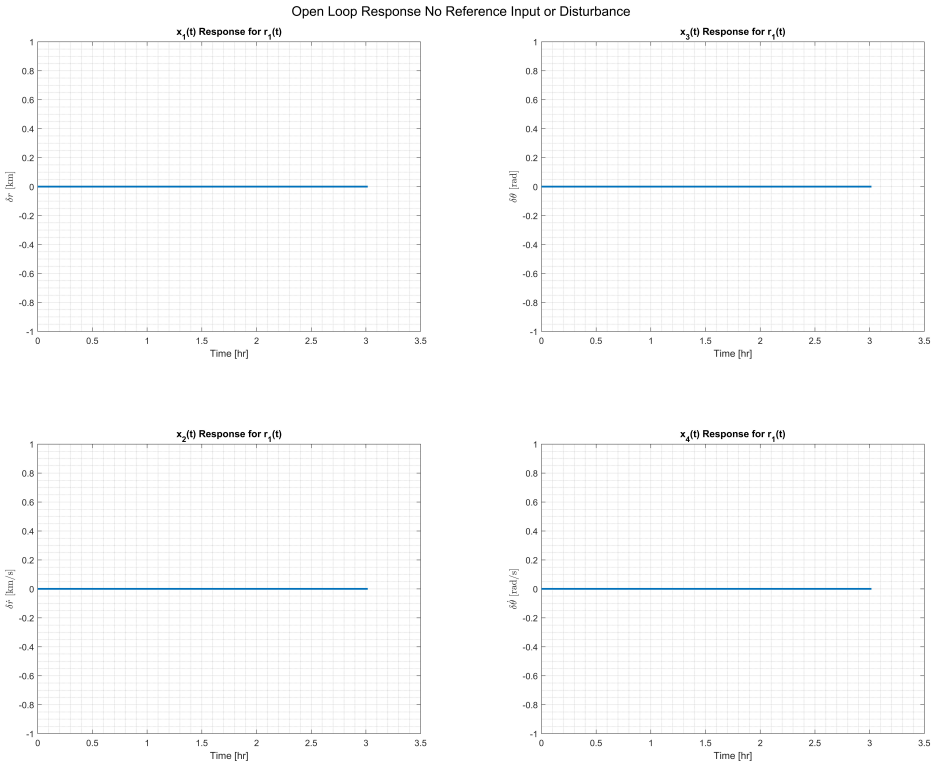

Figure 1: System Response for No Inputs and No Disturbance

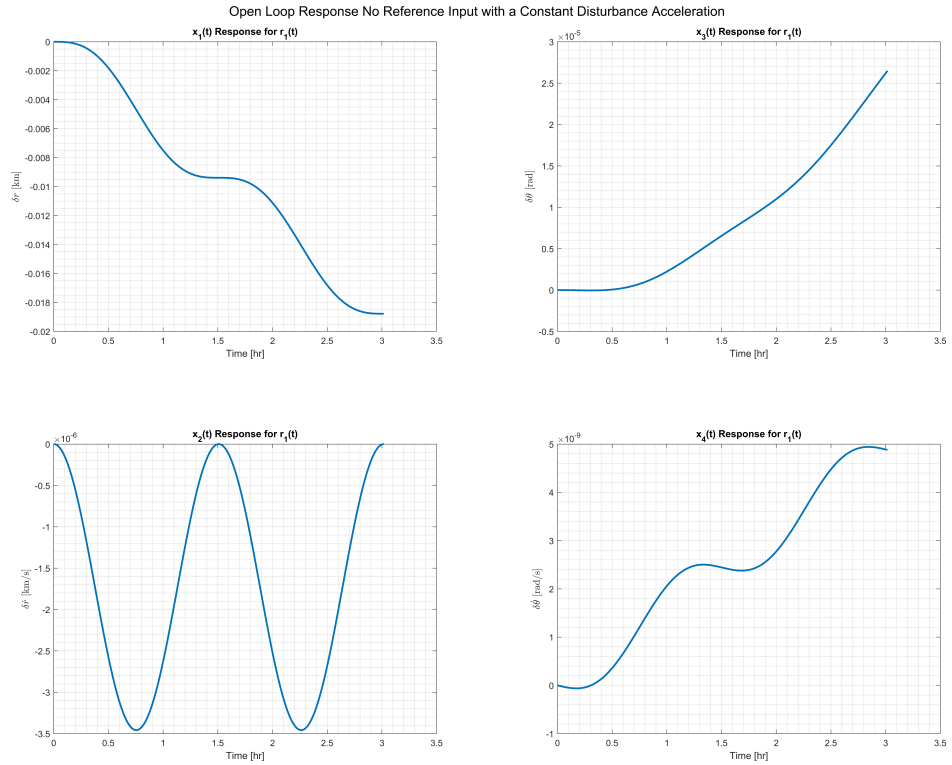

Figure 2: System Response for No Inputs and with Constant Disturbance of $-1 \times 10^{-9} \frac{km}{s^2}$

Additionally, a spacecraft in LEO at 300 km above Earth's surface will experience some constant disturbances. Figs. 1 and 2 show the resulting perturbed state vector of the spacecraft completely undisturbed and with a constant disturbance respectively. As seen in the figures, even a small constant disturbance can lead to large deviations in perturbed state variables, which invalidates any linearized analysis. For the purpose of this project, one of the desired control objectives is disturbance rejection of a constant acceleration. This will be modeled as a "drag-like" acceleration $d(t) = -1 \times 10^{-9} \hat{\theta} \ \frac{km}{s^2} \quad \forall \ t$ that opposes the tangential velocity component. Since this is initially a circular orbit and all perturbations are small, it is a reasonable assumption that the entire orbit is circular and therefore the entire velocity vector is tangential. In the open loop model, this

disturbance is accounted for by augmenting the $\tilde{B}$ and $\tilde{D}$ matrices as seen in eq. (5).

$$\tilde{B_d} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \frac{1}{r_0} \end{bmatrix}, \tilde{B}_{aug,OL} = \begin{bmatrix} \tilde{B} & \tilde{B_d} \end{bmatrix}, \tilde{D}_{aug,OL} = \begin{bmatrix} \tilde{D} & 0_{2x1} \end{bmatrix}, \delta\vec{u}_{aug}(t) = \begin{bmatrix} \delta u_1(t) \\ \delta u_2(t) \\ d(t) \end{bmatrix} \tag{5}$$

Lastly, an additional objective will be reference tracking. For this project, there will be two separate reference input profiles:

$$\delta r_1(t) = [\delta r_{desired}(t) = 0.2 * 1(t), \delta\theta_{desired}(t) = 0]^T$$
$$\delta r_2(t) = [\delta r_{desired}(t) = 0, \delta\theta_{desired}(t) = 1 \times 10^{-4} * 1(t)]^T$$

Where $1(t)$ is a unit step input. The spacecraft is assumed to be using a higher end ion thruster. Due to the low thrust capabilities of ion thrusters, it is unrealistic to expect reference values to be achieved within minutes or seconds without exceeding the maximum thrust capabilities of the thruster. Current day ion thrusters can produce between 80-600 mN of thrust. Some assumptions about the spacecraft these are hosted on must be made in order to determine the maximum input accelerations. Going forward, it is assumed there is a 1:100 ratio of N:kg, which yields a maximum input acceleration of $u_{max} = 1 \times 10^{-3} \frac{m}{s^2} = 1 \times 10^{-6} \frac{km}{s^2}$.

Fig. 3 show the perturbed states to the open loop response and reference input profiles $r_1(t)$ and $r_2(t)$ given $\delta x_0 = [0.01 \;\; 1 \times 10^{-6} \;\; 5 \times 10^{-6} \;\; 1 \times 10^{-9}]^T$ and no controller. As expected, the reference inputs are not able to tracked via an open loop model. It can also be seen that the δr terms are sinusoidal and have the same period as the orbital period (roughly 1.5 hours). With a non-zero δx_0, it is expected that the values for δr grow until apoapsis and then shrink until periapsis. Similarly, for the $\delta\theta$ term, it is expected that this will continue to drift over time because this is an error that accumulates over time and does not come back to it's original value every orbit period. These also align with the open loop plant poles. The oscillatory behavior comes from the purely imaginary poles and the drift behavior comes from the poles located at zero.

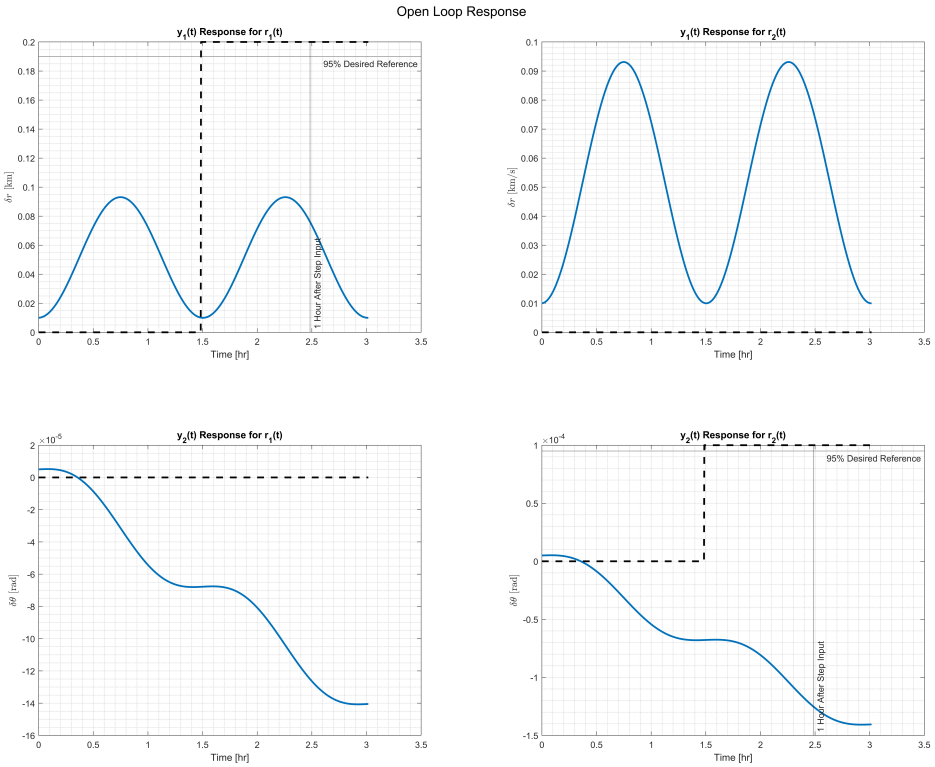

Figure 3: System Outputs for $r_1(t)$ and $r_2(t)$ with No Control and Constant Disturbance of $-1 \times 10^{-9} \frac{km}{s^2}$

A summary of all the control objectives are as follows:

1. All closed loop poles shall be in the left-hand plane.
2. The spacecraft shall return to its initial orbit given a non-zero δx_0.
3. The spacecraft shall reach 95% of desired reference values within 1 hour.
4. The spacecraft shall achieve zero steady state error, even with a constant disturbance.
5. The spacecraft shall achieve less than 10% overshoot.
6. The spacecraft shall settle to 98% of its reference values within 1.5 hours.
7. The closed loop thruster response shall not exceed $1 \times 10^{-6} \frac{km}{s^2}$.

2.3 Question 3: System Reachability and Observability

Taking the linearized system from eq. (4), the reachability matrix and observability matrix can be formed via eqs. (6) and (7) in order to determine if the system is completely reachable and observable.

$$P = [\tilde{B} \quad \tilde{A}\tilde{B} \quad \tilde{A}^2\tilde{B} \quad \tilde{A}^3\tilde{B}] \tag{6}$$

$$O = [\tilde{C} \quad \tilde{C}\tilde{A} \quad \tilde{C}\tilde{A}^2 \quad \tilde{C}\tilde{A}^3]^T \tag{7}$$

Both of the matrices have a rank of 4, and because $n = 4$, the system is both completely reachable and observable. These subspaces will be equivalent to the range space of their respective matrices. Using MATLAB's orth() command, the (orthonormal) basis of the reachable and observable subspaces are as follows:

$$\text{Range(P)} = \text{span}\left\{ \begin{bmatrix} 0 \\ -1 \\ -3.4648 \times 10^{-7} \\ 0 \end{bmatrix}, \begin{bmatrix} -1 \\ 0 \\ 0 \\ 3.4648 \times 10^{-7} \end{bmatrix}, \begin{bmatrix} 0 \\ 3.4648 \times 10^{-7} \\ -1 \\ 0 \end{bmatrix}, \begin{bmatrix} 3.4648 \times 10^{-7} \\ 0 \\ 0 \\ 1 \end{bmatrix} \right\}$$

$$\text{Range}(O) = \text{span}\left\{ \begin{bmatrix} -1.6783 \times 10^{-8} \\ 0 \\ 0 \\ -0.0646 \\ -0.9979 \\ 0 \\ 0 \\ 3.4576 \times 10^{-7} \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ -1 \\ 0 \\ 0 \\ 3.4648 \times 10^{-7} \\ 1.3384 \times 10^{-6} \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \\ 0 \\ -2.5986 \times 10^{-7} \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \right\}$$

3 Part B: Linear Control Design

3.1 Question 4: Manual Pole Placement

3.1.1 Feedforward Input Conditioning

The first iteration of the manual pole placement controller design was done via feedforward input conditioning. This is done by looking at the Final Value Theorem and realizing that because the number of inputs equals the number of outputs, to ensure no steady state error, the feedforward gain can be selected to be $F = [C(-A + BK)^{-1}B]^{-1}$. However, in order to do so, the closed loop matrices had to be formulated via eqs. (8)-(10).

$$\begin{aligned} \delta\dot{x}(t) &= \tilde{A}\,\delta x(t) + \tilde{B}\,\delta u(t) + B_d d \\ \delta y(t) &= \tilde{C}\,\delta x(t) + \tilde{D}\,\delta u(t) \\ \delta u(t) &= -K\,\delta x(t) + F\,\delta r(t) \end{aligned} \tag{8}$$

$$\delta\dot{x}(t) = (\tilde{A} - \tilde{B}K)\;\; \delta x(t) + BF\,\delta r(t) + B_d d \tag{9}$$

$$\begin{aligned} \tilde{A}_{aug,CL} &= \tilde{A} - \tilde{B}K \\ \tilde{B}_{aug,CL} &= [\tilde{B}F \quad B_d] \\ \delta r_{aug}(t) &= [\delta r(t) \quad d]^T \\ \tilde{C}_{aug,CL} &= \tilde{C} \\ \tilde{D}_{aug,CL} &= \tilde{D} \end{aligned} \tag{10}$$

The feedback gain matrix K was iteratively refined using MATLAB's place() command. This process began by choosing poles very close to the imaginary axis due to the fact that the maximum control acceleration allowed by the ion thrusters is very small. After some initial tuning, the best poles for this controller design were selected to be $s_1 = -0.00277$, $s_2 = -0.00276$,$s_3 = -0.00275$, and $s_4 = -0.0018$. Poles selected to be any lower caused the rise time to take much longer and poles selected to be higher increased thruster responses.

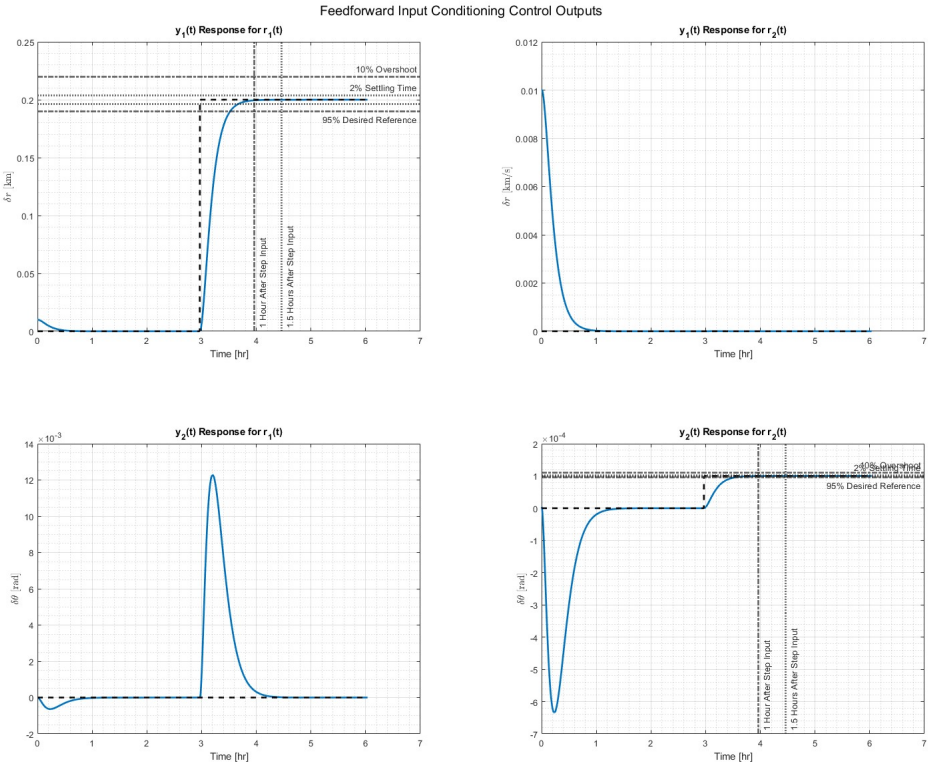

Figure 4: System Outputs for $r_1(t)$ and $r_2(t)$ with Feedforward Input Conditioning Control and Constant Disturbance of $-1 \times 10^{-9} \frac{km}{s^2}$

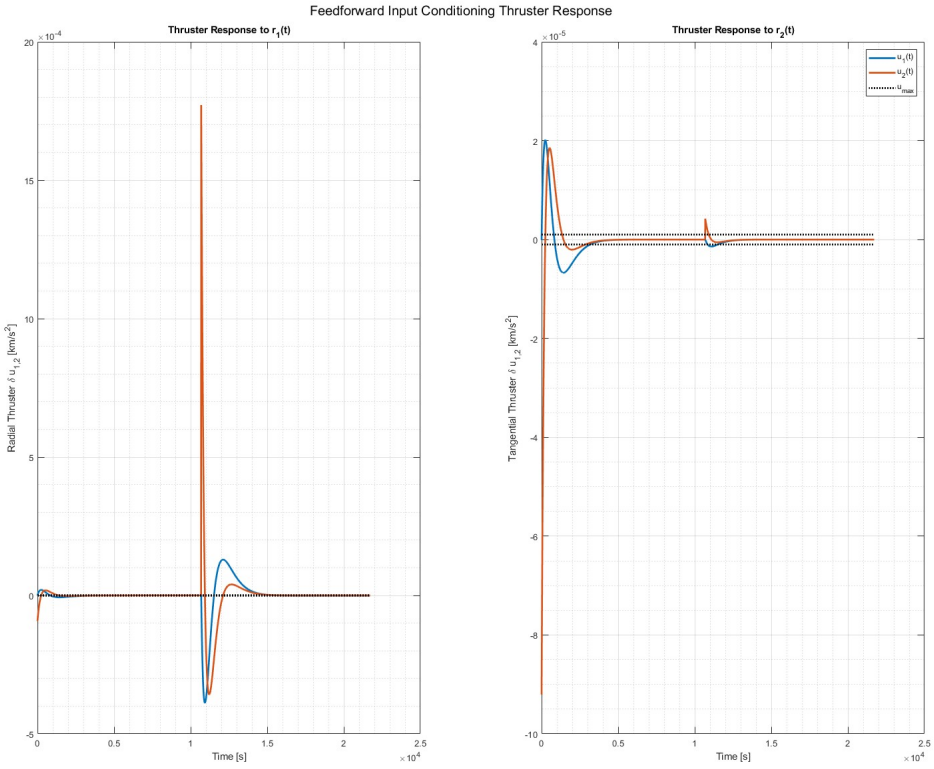

Figure 5: Thruster Response for $r_1(t)$ and $r_2(t)$ with Feedforward Input Conditioning Control and Constant Disturbance of $-1 \times 10^{-9} \frac{km}{s^2}$

Figs. 4 and 5 show the resulting outputs for both reference profiles and the resulting thruster control accelerations with an initial condition of $\delta x_0 = [0.01 \;\; 1 \times 10^{-6} \;\; 5 \times 10^{-6} \;\; 1 \times 10^{-9}]^T$. It can be seen that the all system objectives are met besides the limit on the control accelerations. Any further tuning resulted in at least one of transient response objectives not being met or the continued violation of the control acceleration limit objective.

3.1.2 Integral Control

With the failure of feedforward input conditioning control, the next logical step was to attempt integral control. For integral control, the state vector is augmented to create new states from $\delta \dot{x_I} = \delta r(t) - \delta y(t) = \delta r(t) - \tilde{C}\delta x(t) - \tilde{D}\delta u(t)$, where $\delta u(t) = -K\delta x(t)$, which tracks the error between the reference and the output. Eqs. (11)-(12) yield the augmented closed loop dynamics.

$$\begin{bmatrix} \delta\dot{x}(t) \\ \delta\dot{x_I}(t) \end{bmatrix} = \underbrace{\begin{bmatrix} \tilde{A} & 0_{4\times2} \\ -\tilde{C} & 0_{2\times2} \end{bmatrix}}_{A_{aug,OL}} \overbrace{\begin{bmatrix} \delta x(t) \\ \delta x_I(t) \end{bmatrix}}^{\delta x_{aug,OL}} + \underbrace{\begin{bmatrix} \tilde{B} \\ 0_{2\times2} \end{bmatrix}}_{B_{aug,OL}} \delta u(t) \; + \begin{bmatrix} B_d \\ 0_{2\times1} \end{bmatrix} d \; + \begin{bmatrix} 0_{4\times2} \\ I \end{bmatrix} \delta r(t) \tag{11}$$

$$\begin{aligned} \tilde{A}_{aug,CL} &= \tilde{A}_{aug,OL} - \tilde{B}_{aug,OL}K \\ \tilde{B}_{aug,CL} &= \begin{bmatrix} 0_{4\times2} & B_d \\ I & 0_{2\times1} \end{bmatrix} \\ \delta r_{aug}(t) &= [\delta r(t) \;\; d]^T \\ \tilde{C}_{aug,CL} &= \begin{bmatrix} \tilde{C} & 0_{2\times2} \end{bmatrix} \\ \tilde{D}_{aug,CL} &= \begin{bmatrix} \tilde{D} & 0_{2\times2} \end{bmatrix} \end{aligned} \tag{12}$$

This also meant that reachability had to be re-checked with the equivalent augmented open loop matrices. Doing so proved that the rank of the reachability matrix was now 6, therefore this augmented system is fully reachable. Then, utilizing the same methodology as in the feedforward input conditioning control section, but with an additional two pole for the integral error tracking states, led to the poles $s_1 = -0.00279$, $s_2 = -0.00278$, $s_3 = -0.00277$, $s_4 = -0.00276$, $s_5 = -0.00275$, and $s_6 = -0.0018$.

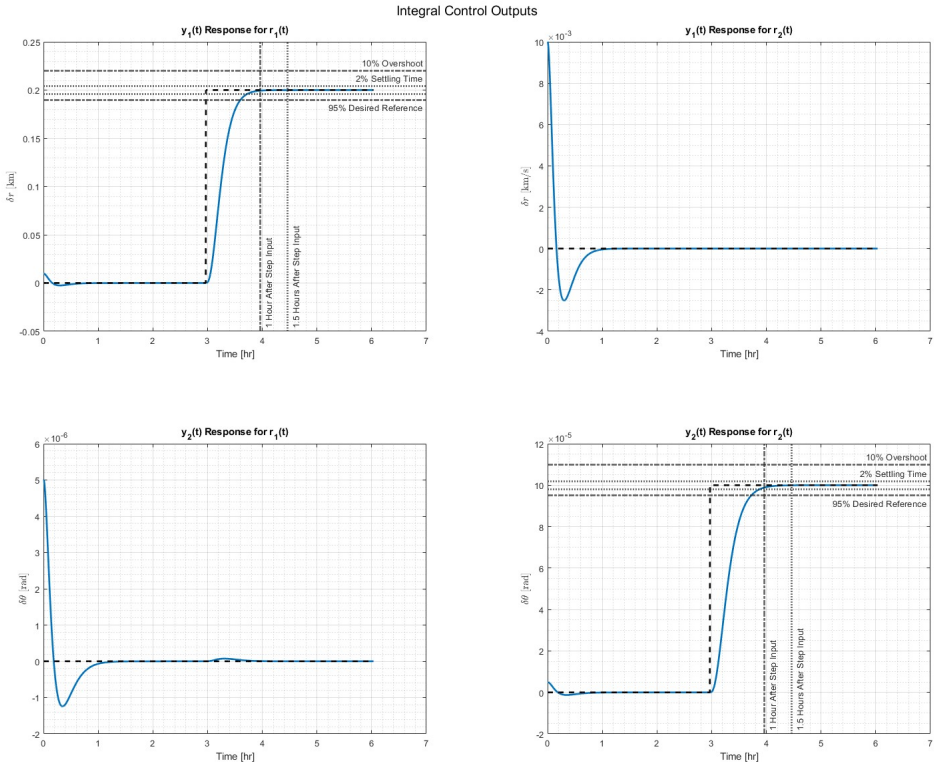

Figure 6: System Outputs for $r_1(t)$ and $r_2(t)$ with Integral Error Control and Constant Disturbance of $-1 \times 10^{-9} \frac{km}{s^2}$

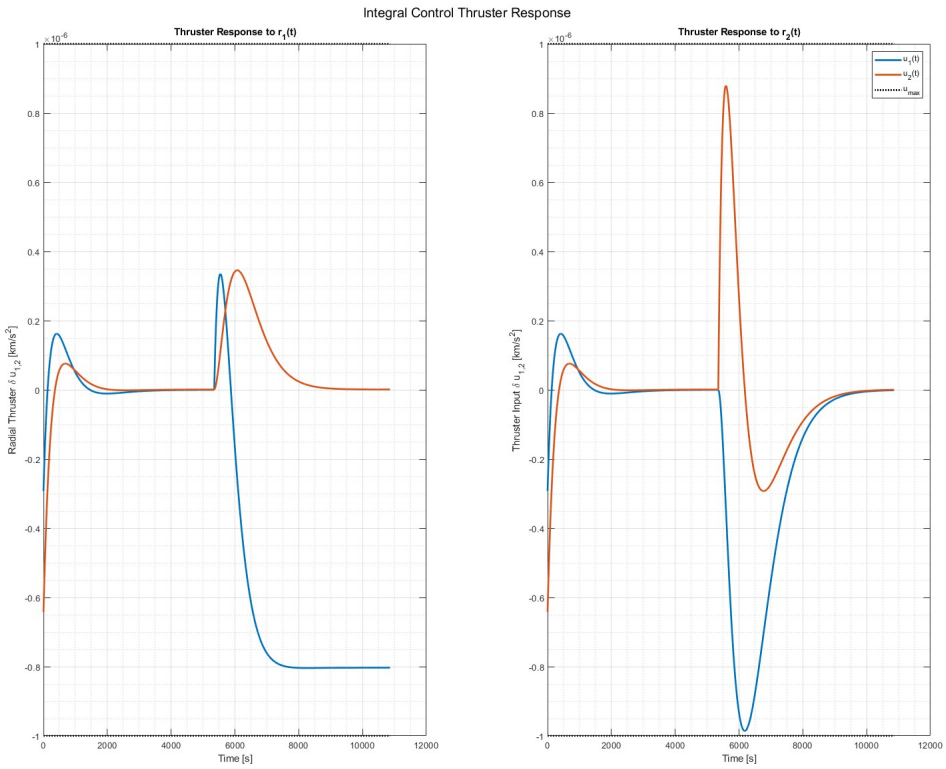

Figure 7: Thruster Response for $r_1(t)$ and $r_2(t)$ with Integral Error Control and Constant Disturbance of $-1 \times 10^{-9} \frac{km}{s^2}$

Figs. 6 and 7 show the resulting outputs for both reference profiles and the resulting thruster control accelerations with an initial condition of $\delta x_0 = [0.01 \;\; 1 \times 10^{-6} \;\; 5 \times 10^{-6} \;\; 1 \times 10^{-9}]^T$. It can be seen that all system objectives are met when switching to integral control. What is interesting is how different the thruster response is when switching to integral control. For feedforward input conditioning, the thruster response was 1-2 magnitudes larger than the allowable accelerations. With integral control using the same poles, plus two new poles right next to the non-dominant poles, the thruster accelerations are within the limits. This goes to show how useful integral control can be on low-effort limited systems.

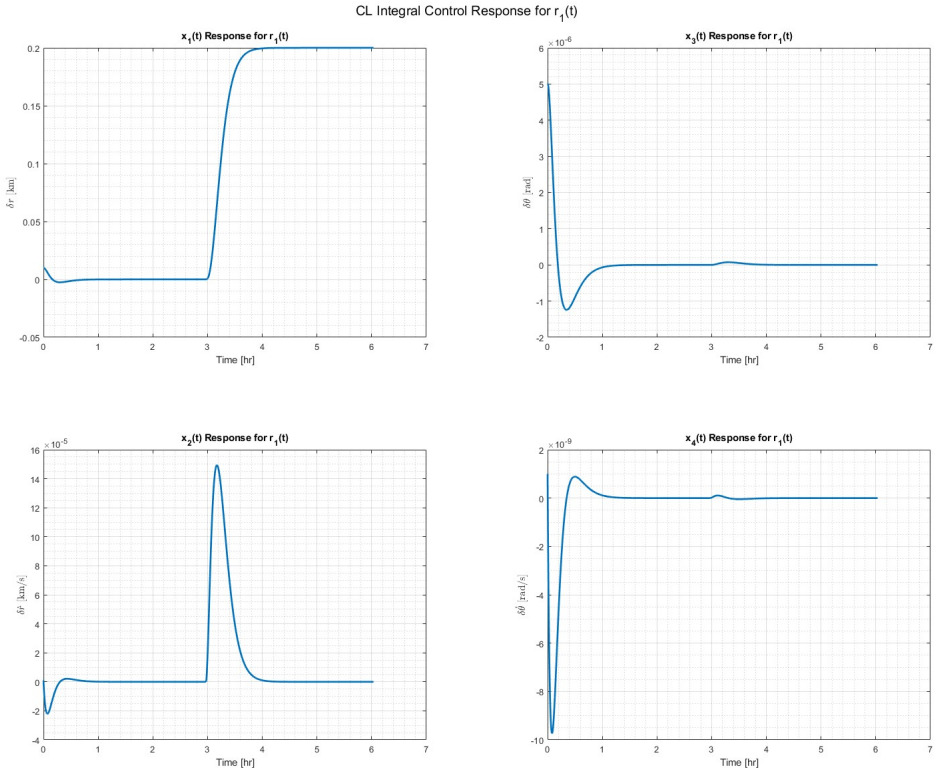

Figure 8: State History for $r_1(t)$ with Integral Error Control and Constant Disturbance of $-1 \times 10^{-9} \frac{km}{s^2}$

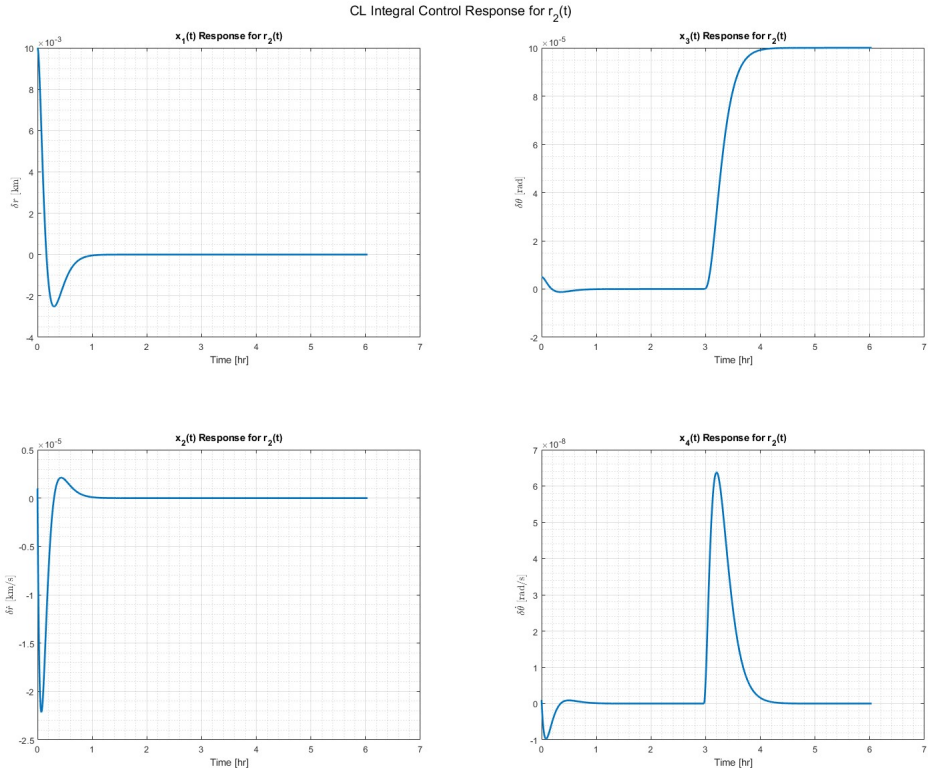

Figure 9: State History for $r_2(t)$ with Integral Error Control and Constant Disturbance of $-1 \times 10^{-9} \frac{km}{s^2}$

Figs. 8 and 9 show the full dynamical state of the system with the implemented integral control. We can see that even for the unobserved states, the response is reasonable.

3.2 Question 5: Luenberger Observer Design

The Luenberger Observer design began with deriving the proper augmented matrices. Eqs. (13)-(21) highlight the augmentation of matrices to include the observer. See Appendix 5.2 for where these derivations came from.

$$A_{\text{aug,CL,obs}} = \begin{bmatrix} A_{\text{aug,CL}} & B_{\text{aug}} K_x \\ \mathbf{0}_{5\times 6} & A - LC \end{bmatrix} \tag{13}$$

$$B_{\text{aug,CL,obs}} = \begin{bmatrix} F_{\text{aug}} \\ \mathbf{0}_{5\times 2} \end{bmatrix} \tag{14}$$

$$C_{\text{aug,CL,obs}} = \begin{bmatrix} C & \mathbf{0}_{2\times 2} & \mathbf{0}_{2\times 5} \end{bmatrix}, \qquad D_{\text{aug,CL,obs}} = \mathbf{0}_{2\times 2}. \tag{15}$$

$$A_{\text{aug,CL}} = A_{\text{aug}} - B_{\text{aug}} K_{\text{aug}}, \qquad K_x = K_{\text{aug}}(:, 1:4) \tag{16}$$

$$A_{\text{aug}} = \begin{bmatrix} A & \mathbf{0}_{4\times 2} \\ -C & \mathbf{0}_{2\times 2} \end{bmatrix}, \qquad B_{\text{aug}} = \begin{bmatrix} B \\ \mathbf{0}_{2\times 2} \end{bmatrix} \tag{17}$$

$$F_{\text{aug}} = \begin{bmatrix} \mathbf{0}_{4\times 2} \\ I_2 \end{bmatrix} \tag{18}$$

$$X(t) = \begin{bmatrix} x_{\text{aug}}(t) \\ e(t) \\ e_d(t) \end{bmatrix} \in R^{11} \tag{19}$$

$$\dot{X}(t) = A_{\text{aug,CL,obs}} X(t) + B_{\text{aug,CL,obs}}\, r(t), \tag{20}$$

$$Y(t) = C_{\text{aug,CL,obs}} X(t) + D_{\text{aug,CL,obs}}\, r(t), \qquad Y(t) \in R^2. \tag{21}$$

The next step in the Luenberger Observer design revolved around picking the desired observer poles. The poles for the observer were selected based on the dominant poles of the closed-loop system. Ultimately, the desired observer poles were chosen to be $2 * [-0.00279 - 0.00278 - 0.00277 - 0.00276 - 0.00275]$. This ensures that the observer poles lie to the left of the closed loop poles. This will allow the plant error to decay twice as fast than the plant dynamics, ultimately minimizing coupling between the estimation and control loops. Multiplying by more than 2, however, could risk sensor sensitivity to noise. Sensitivity to noise could have negative impacts on the practical feasibility of this design depending on the sensors being used, so the desired observer poles being used here are a safe estimate. Figs. 10-12 display the simulation of the system after adding in the Luenberger Observer for a non-zero initial error while figs. 13-15 display the same for a zero initial error. Fig 10 displays the error decaying to zero over time after giving the system some initial error. The error decays to zero for each state around 1,800 seconds (0.5 hours). In order to minimize this, the system would have to take in higher magnitude observer poles which would ultimately have a negative affect on the actuator output due to increased sensitivity to noise. Fig. 11 shows the limitations of the previously selected K_{aug} value combined with a non zero initial condition. This forces a large initial correction burn that lies outside the magnitude of our actuator limit as well as an overshoot of the actuator limit. This means that this type of control would not be feasible for the requirements of this system as listed in section 2.2. However, this should be minimized when implementing LQR in the next part.

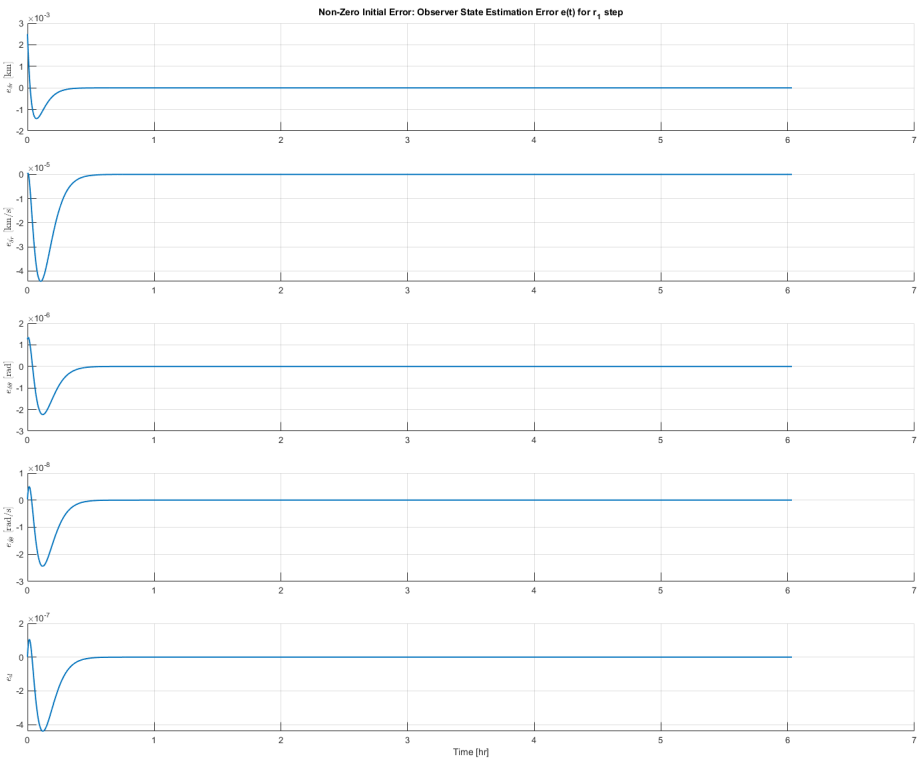

Figure 10: Non-zero Initial Error: Error Plots

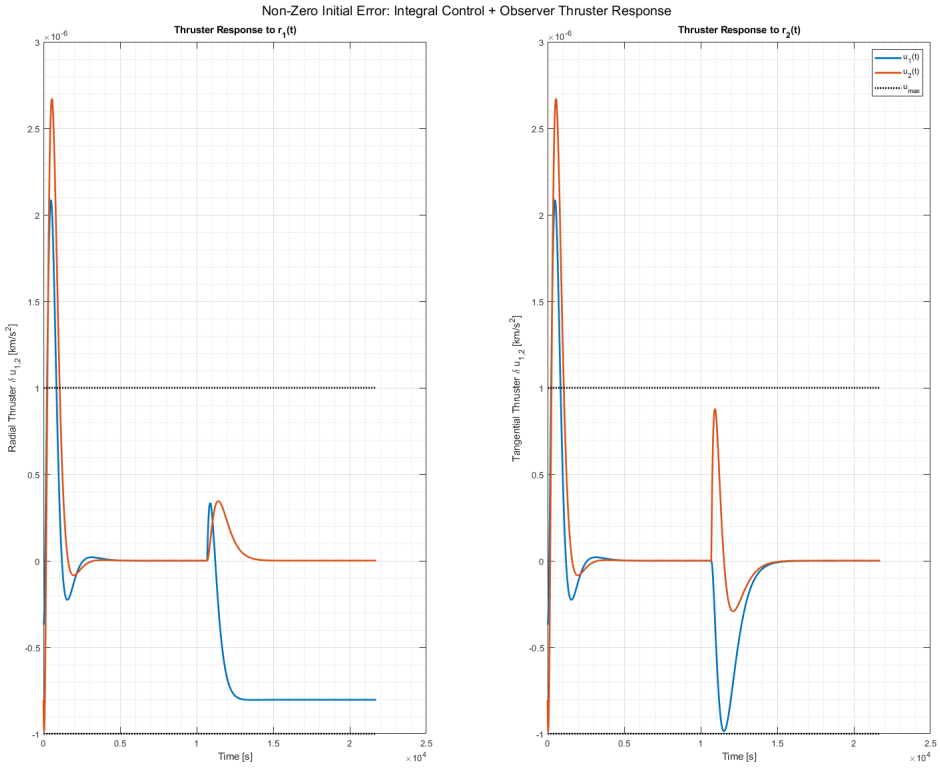

Figure 11: Non-zero Initial Error: Thruster Plots

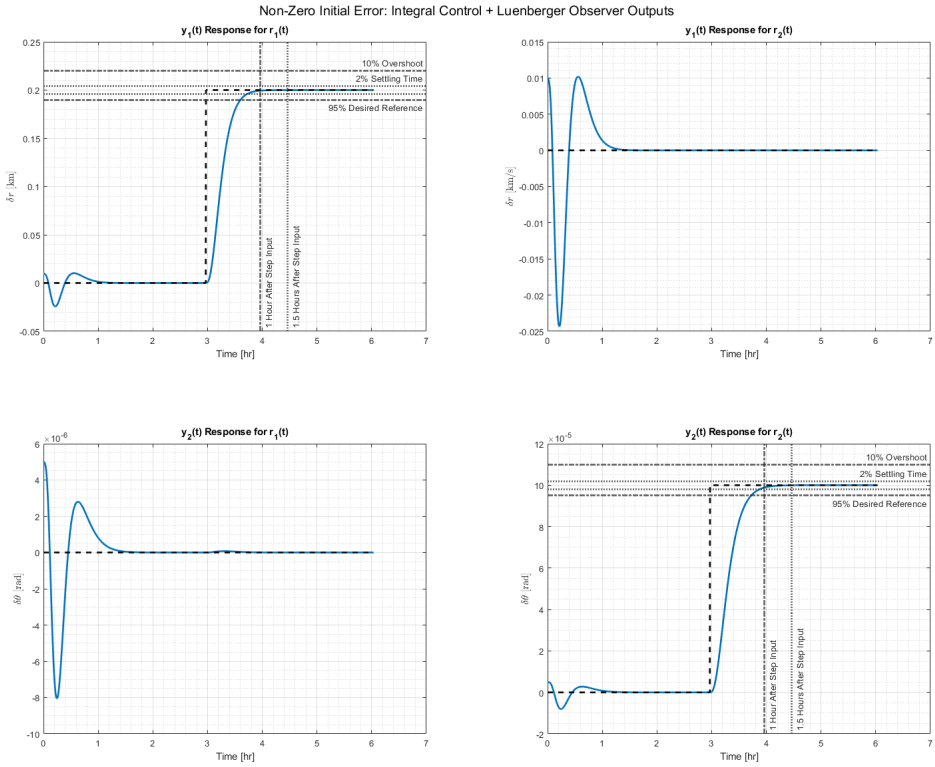

Figure 12: Non-zero Initial Error: State Outputs

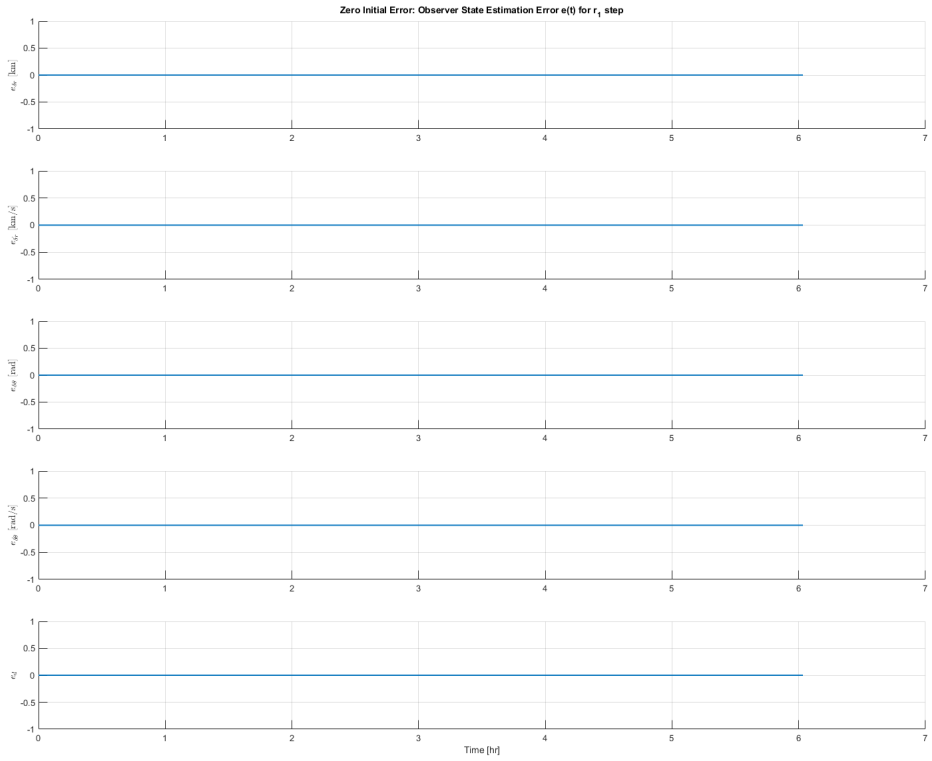

Figure 13: Zero Initial Error: Error Plots

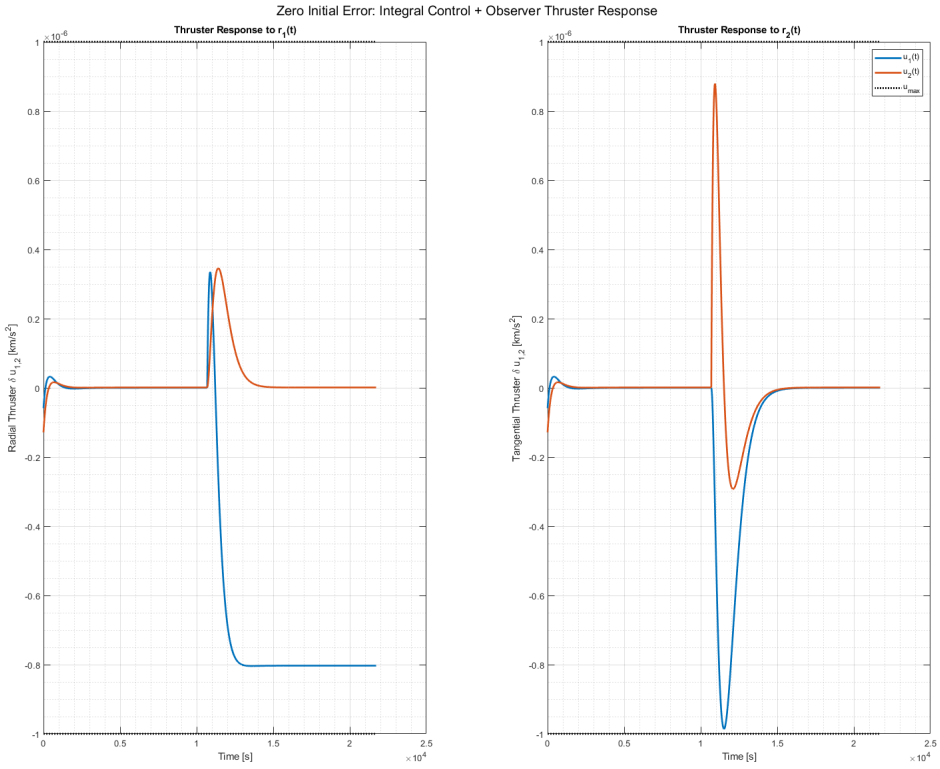

Figure 14: Zero Initial Error: Thruster Plots

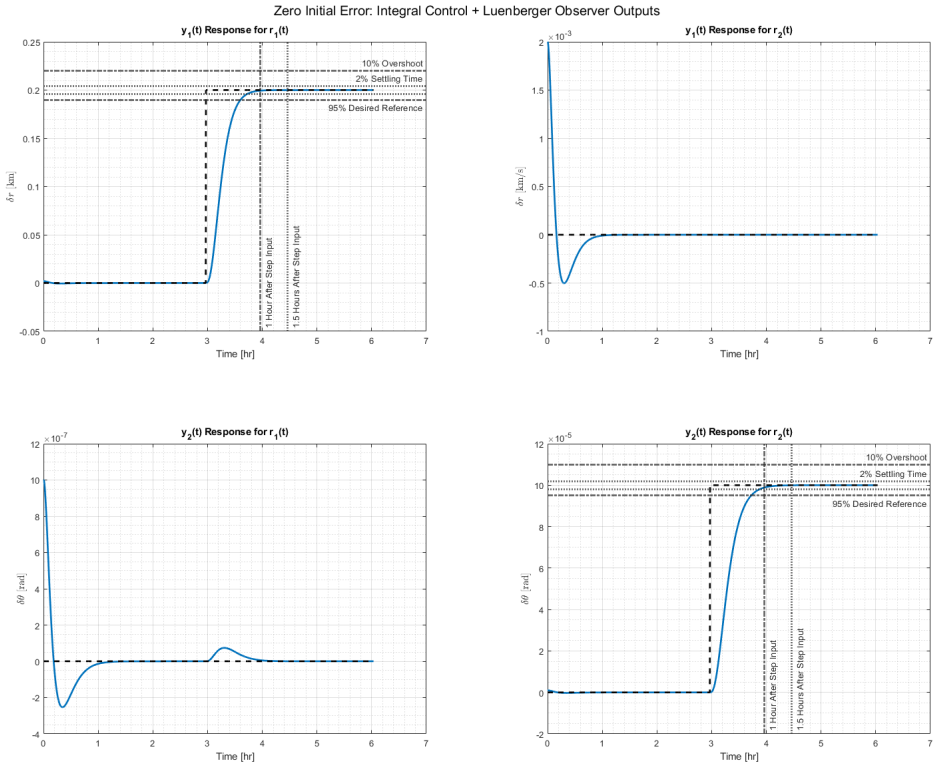

Figure 15: Zero Initial Error: State Outputs

The outputs of Fig. 6 (from question 4) and Fig. 15 are nearly identical. This shows that the observer does not change the plant dynamics and the separation principle holds. The practical feasibility of this design with respect to the observer is conditional. The design uses small initial error values $0.25 * [0.01 \;\; 1e \times 10^{-6} \;\; 5 \times 10^{-6} \;\; 1 \times 10^{-9} \;\; 1 \times 10 - 9]$, or 25% of our perturbed initial conditions. Making the initial error values much larger than that makes the error of the system blow up, so it is important to minimize the initial error. Additionally, as stated previously, moving the observer poles farther left would make the error decay to zero faster, but would risk sensitivity to noise. Two times the desired poles from pole placement is a modest guess that would be feasible for a real life system.

3.3 Question 6: LQR Design

In order to tune the LQR controller, Bryson's rule was utilized to create a methodical way of penalizing state and control deviations. It is also important to note that the LQR controller was designed separately from the Luenberger observer and then once tuned, "attached" to the observer. This was done via the separation principle. With the integral controller augmented state, the Q

and R matrices are shown in eqs. (22)-(23).

$$Q = \text{diag}\left(\frac{\alpha_1^2}{(\delta x_{1,\max})^2}, \frac{\alpha_2^2}{(\delta x_{2,\max})^2}, \frac{\alpha_3^2}{(\delta x_{3,\max})^2}, \frac{\alpha_4^2}{(\delta x_{4,\max})^2}, \frac{\alpha_5^2}{(\delta x_{5,\max})^2}, \frac{\alpha_6^2}{(\delta x_{6,\max})^2} \right) \tag{22}$$

$$R = \rho \cdot \text{diag}\left(\frac{\beta_1^2}{(\delta u_{1,\max})^2}, \frac{\beta_2^2}{(\delta u_{2,\max})^2} \right) \tag{23}$$

This allows for the tuning via $\vec{\alpha}$, $\vec{\beta}$, and ρ. To begin this process, logical values for state deviations and control accelerations were decided to be $\delta\vec{x}_{max} = [0.22\ km\ \ 2.2 \times 10^{-2}\ km/s\ \ 1.1 \times 10^{-5}\ rad\ \ 1.1 \times 10^{-6}\ rad/s\ \ 1\ km \cdot s\ \ 1.55 \times 10^{-4}\ rad \cdot s]$ and $\delta\vec{u}_{max} = [1 \times 10^{-6} km/s^2\ \ 1 \times 10^{-6} km/s^2]$. The values for $\delta x_{1,\max}$ and $\delta x_{3,\max}$ comes from the 10% overshoot requirement. $\delta x_{2,\max}$ and $\delta x_{4,\max}$ were chosen to be 1 magnitude smaller. $\delta x_{5,\max}$ and $\delta x_{6,\max}$ are not as intuitive since they are integral error states, however, since they represent accumulated error, the idea was to keep their max deviations low. In the end, these values for $\delta x_{5,\max}$ and $\delta x_{6,\max}$ were also used as tuning knobs. The values for $\delta\vec{u}_{max}$ come from the acceleration limit of the ion thruster.

The next step was to decide on values for $\vec{\alpha}$, $\vec{\beta}$, and ρ. It is known that if R is large relative to Q, control is expensive and therefore $||\delta u||^2$ has to be small, leading to a slower response. For the ion thrusters, the accelerations are very limited in magnitude and the control objectives allow for a fair amount of time for rise time and settling time requirements. This indicated that a large R relative to Q is desirable. To begin, ρ was scaled until a "good enough" thruster response was achieved, particularly at the beginning of the simulation due to the non-zero initial conditions. Then $\vec{\beta}$ was initially set to have equal weights between the thrusters. For $\vec{\alpha}$, initial values were set based on control requirements. Values for α_1 and α_3 were set much higher because those states had very specific requirements for tuning, while values for α_2, α_4, α_5, and α_6 were set much lower because no control objectives were directly tied to these state variables. While they still have an impact, relative to α_1 and α_3, they are not as important.

Eventually, through the iterative process of changing $\vec{\alpha}$, $\vec{\beta}$, and ρ one at a time and viewing the impact on the results, the final tuning parameters were $\vec{\alpha} = [0.4444\ \ 0.1481\ \ 0.2963\ \ 0.0444\ \ 0.0444\ \ 0.0222]$, $\vec{\beta} = [0.5\ \ 0.5]$, and $\rho = 80$. With this tune, the closed loop poles for the system are $s_{1,2} = -8.1238 \times 10^{-4} \pm 7.2579 \times 10^{-4} i$, $s_{3,4} = -0.009 \pm 0.0022i$, and $s_{5,6} = -0.0018 \pm 0.0007i$ with the same observer poles from question 5. Figs. 16-17 show the outputs responses and the thruster responses for both reference tracking profiles, and figs. 18-19 show the state variables for both reference tracking profiles.

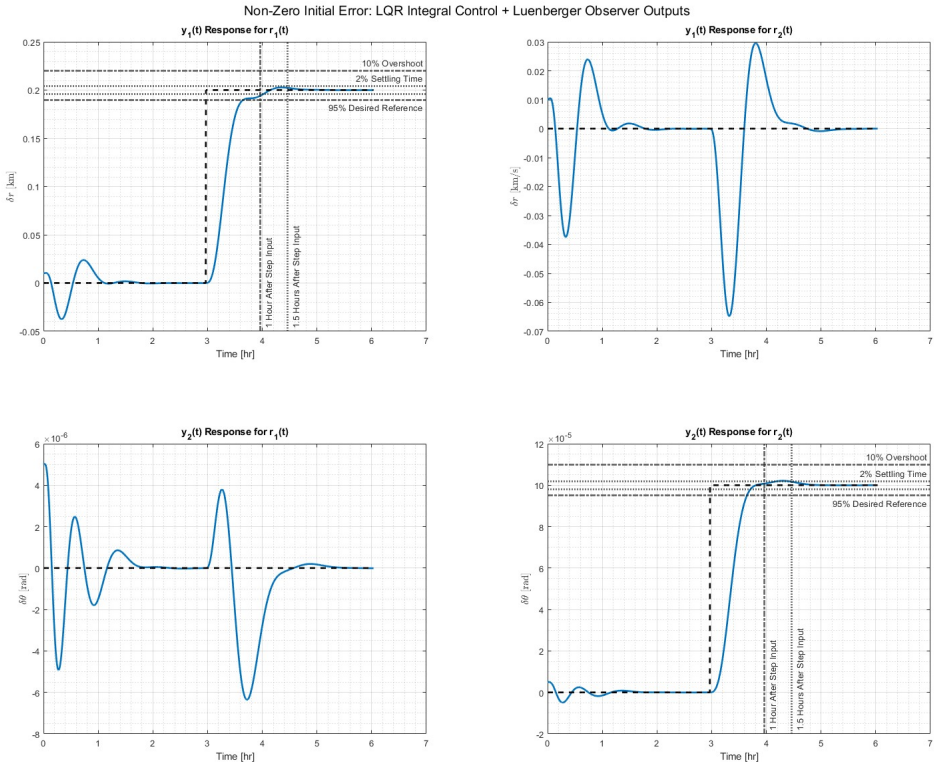

Figure 16: Non-Zero Initial Error: LQR Integral Control + Luenberger Observer Outputs

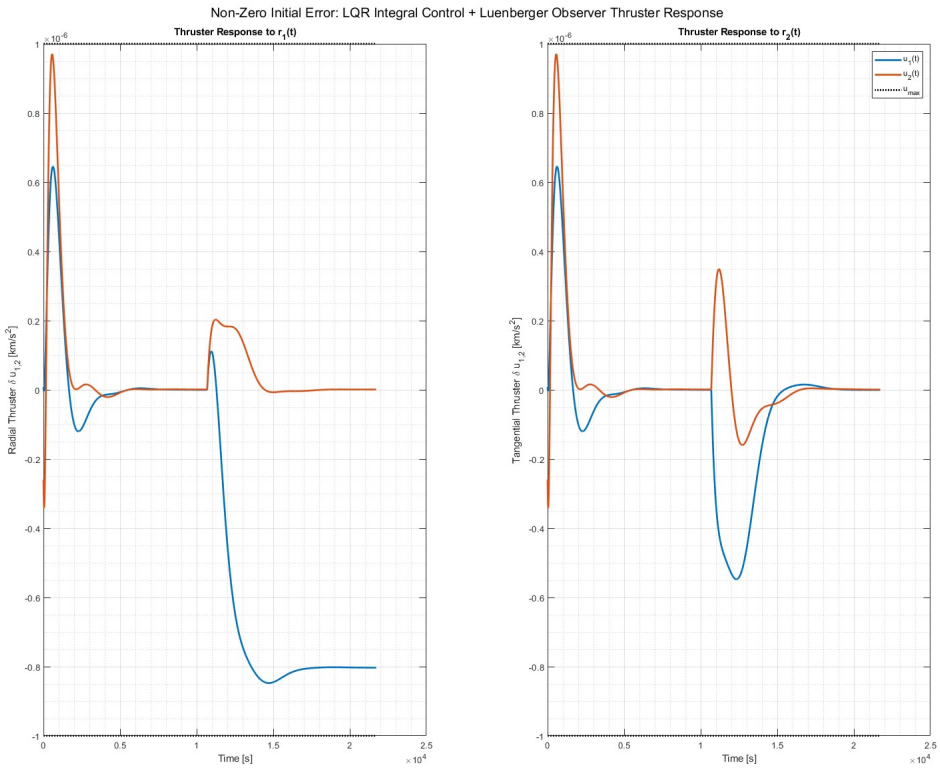

Figure 17: Non-Zero Initial Error: LQR Integral Control + Luenberger Observer Thruster Responses

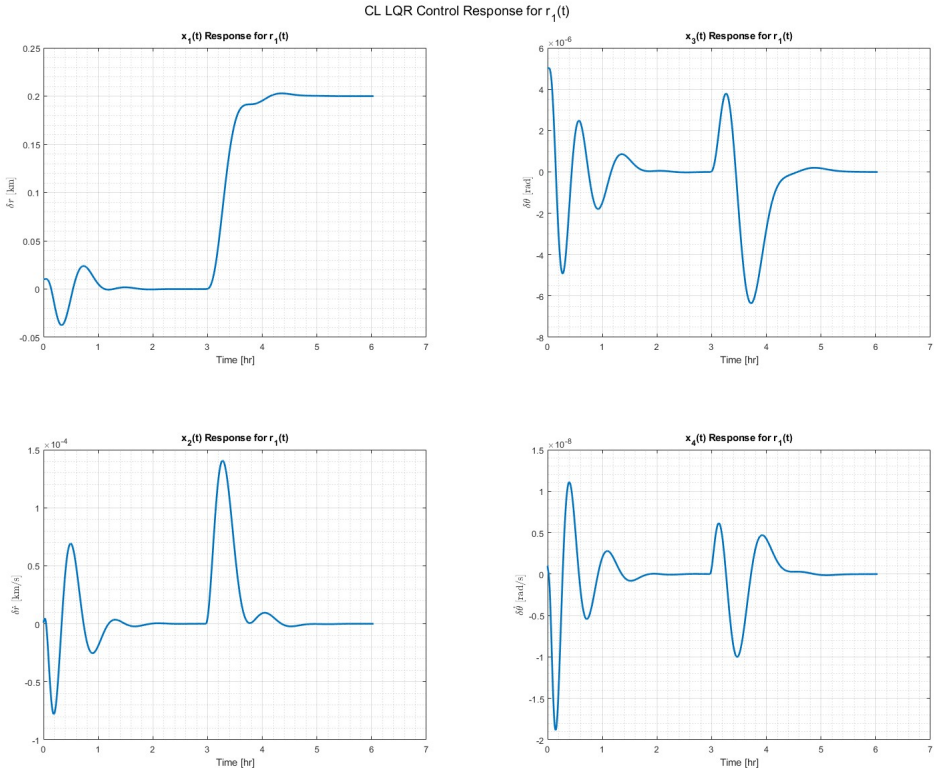

Figure 18: Non-Zero Initial Error: LQR Integral Control + Luenberger Observer State History for $r_1(t)$

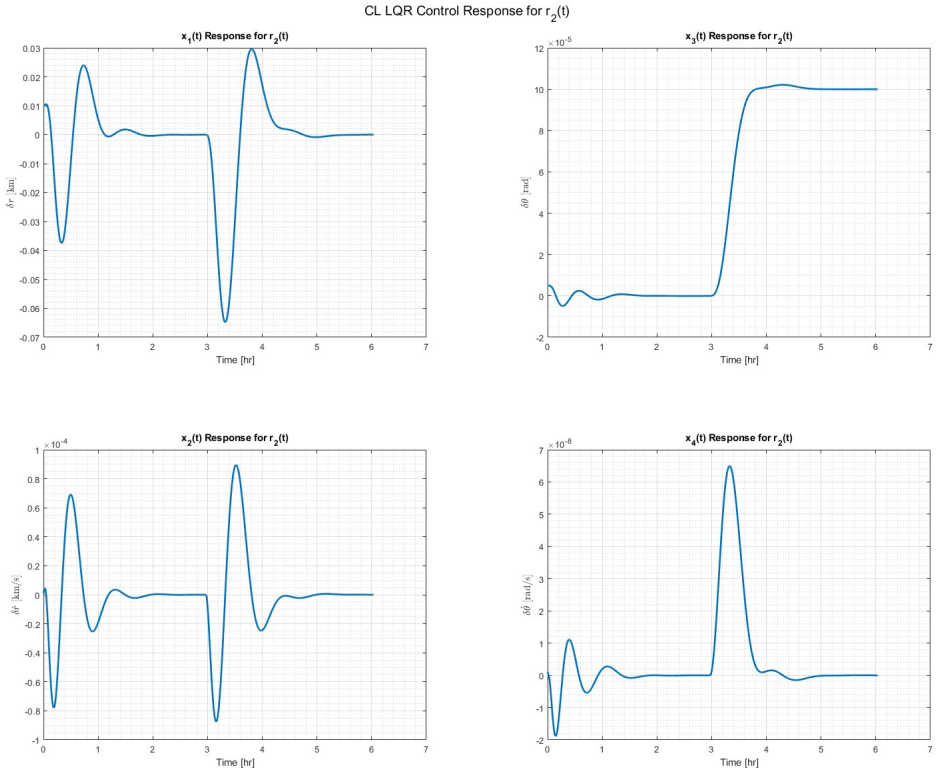

Figure 19: Non-Zero Initial Error: LQR Integral Control + Luenberger Observer State History for $r_2(t)$

First, we compared the outputs between the system with integral control + observer from question 5 (fig. 12) and the system with LQR integral control + observer from question 6 (fig. 16). The main observable difference is the added oscillations in the outputs when using LQR. The K matrices can be used to understand this. While LQR is "optimizing" the gains, some K values become smaller, therefore decreasing damping. This can be seen in entries (2,3), (2,4), and (1,4) (see K matrices below) all getting smaller when LQR is implementing. Those entries affect $\delta\theta$ outputs, so it makes sense for that one to be experiencing more oscillations.

$$K_{\text{aug\_LQR}} = \begin{bmatrix} 9.4299 \times 10^{-6} & 3.7949 \times 10^{-3} & -2.19813 \times 10^{-2} & 5.4669 & -4.6004 \times 10^{-9} & 1.7136 \times 10^{-5} \\ 7.4232 \times 10^{-6} & 8.1864 \times 10^{-1} & 2.2307 \times 10^{-2} & 2.09043 \times 10^{1} & -5.3122 \times 10^{-9} & -1.4840 \times 10^{-5} \end{bmatrix}$$

$$K_{\text{aug}} = \begin{bmatrix} 2.68680 \times 10^{-5} & 8.2800 \times 10^{-3} & 1.85979 \times 10^{-11} & 1.54517 \times 10^{1} & -2.10243 \times 10^{-8} & -2.59072 \times 10^{-14} \\ 1.234601 \times 10^{-7} & -2.2917 \times 10^{-3} & 1.187495 \times 10^{-1} & 4.92169 \times 10^{1} & -1.720273 \times 10^{-10} & -9.32326 \times 10^{-5} \end{bmatrix}$$

When comparing the thruster responses between the system with integral control + observer from question 5 (fig. 11) and the system with LQR integral control + observer from question 6 (fig.

17) it is easy to see that using LQR produces a more ideal thrust response. Fig. 11 displays the thrust outputs surpassing the thruster limits for both reference inputs while fig. 17 displays the thrust outputs remaining within the desired boundaries for both reference inputs. Now, comparing the thruster responses between the system with integral control (fig. 7) and the system with LQR integral control + observer from question 6 (fig. 17), there are not too many significant differences. The main noticeable difference is the increased oscillatory thruster responses in fig. 17 as compared to fig. 7. This is due to the decreased damping as mentioned before. Additionally, fig. 17. displays a larger spike in thruster output at the start, but smaller spikes in control effort later on as compared to fig. 7. This is due to LQR preferring a more aggressive correction early, when the state error is the largest. Ultimately, both of these remain within the desired actuator bounds, so both could be feasible for this system.

4 Appendix

4.1 Code

```
clc; clear; close all

%% OL SS System

% Constants
mu = 3.986e5; % km^3/s^2
r0 = 6678; % km

% ICs
% delta_x0 = [0 0 0 0]';
delta_x0 = [0.01 1e-6 5e-6 1e-9]';
delta_x0_int_cont = [0.01 1e-6 5e-6 1e-9 0 0]';

% Linearized SS System
A = [0 1 0 0;
     (3*mu)/r0^3 0 0 2*sqrt(mu/r0);
     0 0 0 1;
     0 -2*sqrt(mu/r0^5) 0 0];

B = [0 0;
     1 0;
     0 0;
     0 1/r0];

C = [1 0 0 0;
     0 0 1 0];

D = [0 0;
     0 0];

%% Thruster Max/Reference Input Profile
% Time Vector
T = (2*pi)/sqrt(mu/r0^3);
delta_t = 10;
tspan = 0:delta_t:4*T;

% Max Acceleration from Thruster
u_max = 1e-3 * 1e-3;

% Reference Tracking
rt_km = 0.2*sign(double(tspan > 10680));
rt_rad = 0.0001*sign(double(tspan > 10680));
```

```
%% Disturbance Profile
% Constant Bias (due to Drag?)
dmag = -1e-9;
dt = dmag.*ones(length(tspan),1);

%% Determine OL Poles and Augment for Disturbances
% OL Poles
V = eig(A);

% Augment for Disturbances
Bd = [0 0 0 1/r0]';
Baug_OL = [B,Bd];
Dd = [0,0]';
Daug_OL = [D,Dd];

% OL Augmented System
OLsys = ss(A,Baug_OL,C,Daug_OL);

%% Full Input Profiles
delta_rt1 = [rt_km(:),zeros(length(tspan),1),dt(:)]; % First thruster only
delta_rt2 = [zeros(length(tspan),1),rt_rad(:),dt(:)]; % Second thruster only

%% Simulate OL Dynamics
[y_OL1,~,x_OL1] = lsim(OLsys,zeros(size(delta_rt1)),tspan,delta_x0);
[y_OL2,~,x_OL2] = lsim(OLsys,zeros(size(delta_rt2)),tspan,delta_x0);

%% Determine Reachability and Observability (and Their Subspaces)
% Determine Reachability and Observability
P = ctrb(A,B);
rank(P)
Ob = obsv(A,C);
rank(Ob)

% Determine Subspaces
rangeP = orth(P);
rangeOb = orth(Ob);

%% Manual Pole Placement Simulation
% % Desired Pole Locations and Feedback Gain
% des_poles = [-0.00277 -0.00276 -0.00275 -0.0018];
% K = place(A,B,des_poles);
%
% % Feedforward Gain
% F = inv(C/(-A+B*K)*B);
%
```

```
% % CL SS Dynamics
% A_CL = A−B∗K;
% B_CL = [B∗F,Bd];
% C_CL = C;
% D_CL = zeros(2,3);
% CLsys = ss(A_CL,B_CL,C_CL,D_CL);
%
% % Simulate Augmented CL Dynamics
% [yaug_CL1,˜,xaug_CL1] = lsim(CLsys,delta_rt1,tspan,delta_x0);
% [yaug_CL2,˜,xaug_CL2] = lsim(CLsys,delta_rt2,tspan,delta_x0);
%
% % Calculate Input u(t)
% u1_aug = −K∗xaug_CL1'+F∗delta_rt1(:,1:2)';
% u2_aug = −K∗xaug_CL2'+F∗delta_rt2(:,1:2)';
% u1_aug = u1_aug';
% u2_aug = u2_aug';

% Integral Control Augmentation
A_aug_OL = [A zeros(4,2);
            −C zeros(2,2)];
B_aug_OL = [B;
            zeros(2,2)];

C_aug_OL = [C zeros(2,2)];
D_aug_OL = zeros(2,2);

P_OL = ctrb(A_aug_OL,B_aug_OL);
O_OL = obsv(A_aug_OL,C_aug_OL);
rank(P_OL)


% Unity feedforward gain
F_aug = [zeros(size(B));
         eye(2)];

% Desired Pole Locations and Feedback Gain
des_poles_aug = [−0.00279 −0.00278 −0.00277 −0.00276 −0.00275 −0.0018];
K_aug = place(A_aug_OL,B_aug_OL,des_poles_aug);

% CL Augmented SS Matrices
A_aug_CL = A_aug_OL−B_aug_OL∗K_aug;
B_aug_CL = [F_aug,[Bd;0;0]];
C_aug_CL = C_aug_OL;
D_aug_CL = [D_aug_OL,[0;0]];
aug_CL_sys = ss(A_aug_CL,B_aug_CL,C_aug_CL,D_aug_CL);
```

```
% Simulate Augmented CL Dynamics
[yaug_CL1,~,xaug_CL1] = lsim(aug_CL_sys,delta_rt1,tspan,delta_x0_int_cont);
[yaug_CL2,~,xaug_CL2] = lsim(aug_CL_sys,delta_rt2,tspan,delta_x0_int_cont);

% Calculate Input u(t)
u1_aug = -K_aug*xaug_CL1';
u2_aug = -K_aug*xaug_CL2';
u1_aug = u1_aug';
u2_aug = u2_aug';

%% Plots
% OL Ouput Reponse
Plot_Outputs(tspan,[y_OL1,y_OL2],[delta_rt1,delta_rt2],'Open Loop Response')

% Output Responses CL Integral Control
% Plot_Outputs(tspan,[yaug_CL1,yaug_CL2],[delta_rt1,delta_rt2],...
% 'Feedforward Input Conditioning Control Outputs')
Plot_Outputs(tspan,[yaug_CL1,yaug_CL2],[delta_rt1,delta_rt2],...
    'Integral Control Outputs')

% Actuator Responses CL Integral Control
% Plot_Thruster_Reponses(tspan,[u1_aug,u2_aug],u_max,...
% 'Feedforward Input Conditioning Thruster Response')
Plot_Thruster_Reponses(tspan,[u1_aug,u2_aug],u_max,...
    'Integral Control Thruster Response')

% States CL Integral Control
% Plot_States(tspan,xaug_CL1,'r_1(t)',...
% 'CL Feedforward Input Conditioning Control Response for r_1(t)')
% Plot_States(tspan,xaug_CL2,'r_2(t)',...
% 'CL Feedforward Input Conditioning Control Response for r_2(t)')
Plot_States(tspan,xaug_CL1,'r_1(t)','CL Integral Control Response for r_1(t)')
Plot_States(tspan,xaug_CL2,'r_2(t)','CL Integral Control Response for r_2(t)')

%% Question 5: Luenberger Observer
des_poles_obs = 2*[-0.00279 -0.00278 -0.00277 -0.00276 -0.00275]; %4 used initially

Aext = [A Bd;
        zeros(1,4) 0];

Cext = [C zeros(2,1)];

Kobs = [K_aug(:,1:4) zeros(2,1)];

%observer gain matrix
%4x4
```

```
L = place(Aext', Cext', des_poles_obs)';

Aerr = Aext − L ∗ Cext;% 5x5

Axe = B_aug_OL ∗ Kobs; % 6x2 ∗ 2x5 = 6x5

% full A matrix (11x11): [x; z; ex; ed]
A_aug_CL_obs = [ A_aug_CL Axe;
                 zeros(5,6) Aerr ];

Bd_aug = [Bd; 0; 0]; % 6x1
B_aug_CL_obs = [F_aug, Bd_aug;
                zeros(5,3) ];

C_aug_CL_obs = [C zeros(2,2) zeros(2,5) ]; % 2x11

D_aug_CL_obs = zeros(2,3);

aug_CL_obs_sys = ss(A_aug_CL_obs, B_aug_CL_obs, C_aug_CL_obs, D_aug_CL_obs);

%initial values
%nonzero initial error
e0 = 0.25.∗[0.01 1e−6 5e−6 1e−9 1e−9]'; % 25% of ICs
z0 = [0;0];

x0_obs = [delta_x0; z0; e0];

robs1 = delta_rt1;
robs2 = delta_rt2;

[y_obs1, ˜, x_obs1] = lsim(aug_CL_obs_sys, robs1, tspan, x0_obs);
[y_obs2, ˜, x_obs2] = lsim(aug_CL_obs_sys, robs2, tspan, x0_obs);

%actuator efforts
K_comb = [K_aug, K_aug(:,1:4), zeros(2,1)]; % 2x11

U_obs1 = −K_comb ∗ x_obs1';
U_obs2 = −K_comb ∗ x_obs2';
U_obs1 = U_obs1';
U_obs2 = U_obs2';

%plotting
%output response
Plot_Outputs(tspan,[y_obs1,y_obs2],[delta_rt1,delta_rt2],...
    'Non−Zero Initial Error: Integral Control + Luenberger Observer Outputs');
% thruster response
```

```
Plot_Thruster_Reponses(tspan,[U_obs1,U_obs2],u_max,...
    'Non−Zero Initial Error: Integral Control + Observer Thruster Response');

% extracting error
e1_obs = x_obs1(:, 7:10);
e1_d = x_obs1(:, 11);
e2_obs = x_obs2(:, 7:10);
e2_d = x_obs2(:, 11);

figure;
state_labels = {'$e_{\delta r}$ [km]', '$e_{\dot{\delta r}}$ [km/s]',...
    '$e_{\delta \theta}$ [rad]', '$e_{\dot{\delta \theta}}$ [rad/s]', '$e_d$'};

for i = 1:5
    subplot(5,1,i);
    hold on;
    grid on;
    if i <= 4
        plot(tspan/3600, e1_obs(:,i),'LineWidth',1.4);
    else
        plot(tspan/3600, e1_d,'LineWidth',1.4);
    end
    ylabel(state_labels{i},'Interpreter','latex');
    if i==1
        title('Non−Zero Initial Error: Observer State Estimation Error e(t) for r_1 step');
    end
    if i==5
        xlabel('Time [hr]');
    end
end


%zero error
e0 = zeros(5,1);
z0 = [0;0];

x0_obs = [0.2*delta_x0; z0; e0];

robs1 = delta_rt1;
robs2 = delta_rt2;

[y_obs1, ~, x_obs1] = lsim(aug_CL_obs_sys, robs1, tspan, x0_obs);
[y_obs2, ~, x_obs2] = lsim(aug_CL_obs_sys, robs2, tspan, x0_obs);

%extracting states
x1_obs = x_obs1(:, 1:4);
```

```
x2_obs = x_obs2(:, 1:4);

%actuator efforts
K_comb = [K_aug, K_aug(:,1:4), zeros(2,1)]; % 2x11

U_obs1 = -K_comb * x_obs1';
U_obs2 = -K_comb * x_obs2';
U_obs1 = U_obs1';
U_obs2 = U_obs2';

%plotting
%output response
Plot_Outputs(tspan,[y_obs1,y_obs2],[delta_rt1,delta_rt2], ...
    'Zero Initial Error: Integral Control + Luenberger Observer Outputs');
%thruster response
Plot_Thruster_Reponses(tspan,[U_obs1,U_obs2],u_max, ...
    'Zero Initial Error: Integral Control + Observer Thruster Response');

% extracting error
e1_obs = x_obs1(:, 7:10);
e1_d = x_obs1(:, 11);
e2_obs = x_obs2(:, 7:10);
e2_d = x_obs2(:, 11);

figure;
state_labels = {'$e_{\delta r}$ [km]', '$e_{\dot{\delta r}}$ [km/s]', ...
    '$e_{\delta \theta}$ [rad]', '$e_{\dot{\delta \theta}}$ [rad/s]', '$e_d$'};

for i = 1:5
    subplot(5,1,i);
    hold on;
    grid on;
    if i <= 4
        plot(tspan/3600, e1_obs(:,i),'LineWidth',1.4);
    else
        plot(tspan/3600, e1_d,'LineWidth',1.4);
    end
    ylabel(state_labels{i},'Interpreter','latex');
    if i==1
        title('Zero Initial Error: Observer State Estimation Error e(t) for r_1 step');
    end
    if i==5
        xlabel('Time [hr]');
    end
end
```

```
%% LQR

% Tuning knobs
alphas = [30 10 20 3 3 1.5];
alphas = alphas./sum(alphas);
xmax = [0.22 2.2e-2 1.1e-5 1.1e-6 1 1.55e-4];
Q = diag(alphas.^2./xmax.^2);
rho = 80;
R = rho.*diag([0.5/(u_max^2) 0.5/(u_max^2)]);

% No Observer
[Kaug_LQR,Waug_LQR,CLevals] = lqr(A_aug_OL,B_aug_OL,Q,R);

% CL Augmented SS Matrices
A_augCL_LQR = A_aug_OL-B_aug_OL*Kaug_LQR;
B_aug_CL = [F_aug,[Bd;0;0]];
C_aug_CL = C_aug_OL;
D_aug_CL = [D_aug_OL,[0;0]];
aug_CL_sys = ss(A_augCL_LQR,B_aug_CL,C_aug_CL,D_aug_CL);

% Simulate Augmented CL Dynamics
[yaug_CL1_LQR,~,xaug_CL1_LQR] = lsim(aug_CL_sys,delta_rt1,tspan,delta_x0_int_cont);
[yaug_CL2_LQR,~,xaug_CL2_LQR] = lsim(aug_CL_sys,delta_rt2,tspan,delta_x0_int_cont);

% Calculate Input u(t)
u1_aug_LQR = -Kaug_LQR*xaug_CL1_LQR';
u2_aug_LQR = -Kaug_LQR*xaug_CL2_LQR';
u1_aug_LQR = u1_aug_LQR';
u2_aug_LQR = u2_aug_LQR';

% Add Observer via Seperation Principle
Faug = [zeros(size(B));
        eye(2);
        zeros(4,2)];

%augment for observer error states
A_augCL_LQR = [A_aug_OL-B_aug_OL*Kaug_LQR, B_aug_OL*[Kaug_LQR(:,1:4),zeros(2,1)];
               zeros(5,6) Aerr];

aug_CL_sysLQR = ss(A_augCL_LQR,[B_aug_CL;zeros(5,3)],[C_aug_CL,zeros(2,5)],D_aug_CL);

%initial values
%nonzero initial error
e0 = 0.25.*[0.01 1e-6 5e-6 1e-9 1e-9]'; % 25% of ICs
z0 = [0;0];
```

```
x0_obs = [delta_x0; z0; e0];

[yaug_CL1_LQR2,~,xaug_CL1_LQR2] = lsim(aug_CL_sysLQR, robs1, tspan, x0_obs);
[yaug_CL2_LQR2,~,xaug_CL2_LQR2] = lsim(aug_CL_sysLQR, robs2, tspan, x0_obs);


%actuator efforts
K_comb = [Kaug_LQR, Kaug_LQR(:,1:4), zeros(2,1)]; % 2x11

U_obs1 = -K_comb * xaug_CL1_LQR2';
U_obs2 = -K_comb * xaug_CL2_LQR2';
U_obs1 = U_obs1';
U_obs2 = U_obs2';

%plotting
%output response
Plot_Outputs(tspan,[yaug_CL1_LQR2,yaug_CL2_LQR2],[delta_rt1,delta_rt2], ...
    'Non-Zero Initial Error: LQR Integral Control + Luenberger Observer Outputs');
%thruster response
Plot_Thruster_Reponses(tspan,[U_obs1,U_obs2],u_max, ...
    'Non-Zero Initial Error: LQR Integral Control + Luenberger Observer Thruster Response');
%states
Plot_States(tspan,xaug_CL1_LQR2,'r_1(t)','CL LQR Control Response for r_1(t)')
Plot_States(tspan,xaug_CL2_LQR2,'r_2(t)','CL LQR Control Response for r_2(t)')
```

4.2 Augmented Closed Loop Observer Matrices Derivation

We define the augmented state as

$$x_{\text{aug}} = \begin{bmatrix} x \\ z \end{bmatrix} \in R^6. \tag{24}$$

The augmented dynamics are

$$\dot{x}_{\text{aug}} = A_{\text{aug}} x_{\text{aug}} + F_{\text{aug}} r. \tag{25}$$

The observer equations are

$$\dot{\hat{x}} = A\hat{x} + Bu + L(y - \hat{y}), \tag{26}$$

$$y = Cx, \qquad \hat{y} = C\hat{x}. \tag{27}$$

Define the estimation error

$$e = x - \hat{x}. \tag{28}$$

Taking the derivative,

$$\dot{e} = \dot{x} - \dot{\hat{x}} \tag{29}$$

$$= (Ax + Bu) - \left(A\hat{x} + Bu + L(x - \hat{x})\right) \tag{30}$$

$$= (A - LC)e. \tag{31}$$

The augmented control law is

$$u = -K_{\text{aug}} \begin{bmatrix} x \\ z \end{bmatrix} = -K_{\text{aug}} x_{\text{aug}} \tag{32}$$

which may also be written as

$$u = -K_x x_{\text{aug}} + K_z e. \tag{33}$$

—

Closed-Loop Augmented Dynamics

Starting from

$$\dot{x}_{\text{aug}} = A_{\text{aug}} x_{\text{aug}} + B_{\text{aug}} u + F_{\text{aug}} r, \tag{34}$$

and substituting the control input,

$$\dot{x}_{\text{aug}} = A_{\text{aug}} x_{\text{aug}} + B_{\text{aug}} \big(- K_{\text{aug}} x_{\text{aug}} + K_z e \big) + F_{\text{aug}} r \tag{35}$$

$$= \left(A_{\text{aug}} - B_{\text{aug}} K_{\text{aug}} \right) x_{\text{aug}} + B_{\text{aug}} K_z e + F_{\text{aug}} r. \tag{36}$$

Define the closed-loop augmented matrix

$$A_{\text{aug,CL}} = A_{\text{aug}} - B_{\text{aug}} K_{\text{aug}}. \tag{37}$$

—

Final Augmented System

Define the full state

$$x = \begin{bmatrix} x_{\text{aug}} \\ e \end{bmatrix} \in R^{11}. \tag{38}$$

Then the combined dynamics become

$$\dot{x} = \begin{bmatrix} \dot{x}_{\text{aug}} \\ \dot{e} \end{bmatrix} = \begin{bmatrix} A_{\text{aug,CL}} & B_{\text{aug}} K_z \\ 0 & A - LC \end{bmatrix} \begin{bmatrix} x_{\text{aug}} \\ e \end{bmatrix} + \begin{bmatrix} F_{\text{aug}} \\ 0 \end{bmatrix} r. \tag{39}$$

Thus,

$$A_{\text{aug,CL,obs}} = \begin{bmatrix} A_{\text{aug,CL}} & B_{\text{aug}} K_z \\ 0 & A - LC \end{bmatrix}, \qquad B_{\text{aug,CL,obs}} = \begin{bmatrix} F_{\text{aug}} \\ 0 \end{bmatrix}. \tag{40}$$

—

Output Equation

$$y = Cx, \tag{41}$$

where the augmented output matrix is

$$C_{\text{aug,CL,obs}} = \begin{bmatrix} C & 0 \end{bmatrix}, \qquad D_{\text{aug,CL,obs}} = \mathbf{0}. \tag{42}$$