Tesla Autopilot Controller

Gain Design Report - EML4312

Pablo F. Coral

December 2nd, 2025

Contents

I. Project Summary

This report documents the systematic design of a control system that performs autonomous lane-change maneuvers for a Tesla Model S cruising at 70 mph. The autopilot uses a PID controller for longitudinal control (x-position, parallel to the lane) and a coupled, double-loop controller for lateral control (y-position, perpendicular to the current lane and pointing to the next lane).

The design methodology will consist of two phases:

1. **Control theory** to obtain initial gains using standard second-order approximations and dominant pole methods.
2. **Optimization** to refine the gains and satisfy tight rise-time and settling-time specifications simultaneously.

All project specifications on rise time, settling time, overshoot, and steady-state error were satisfied by the final controller.

i. Rapid iteration through Python integration

The gain design process was accelerated by utilizing two key python helpers:

1. A **verification script** that evaluated full system response given a set of controller gains using the given plant functions and the Scipy.signal library[1], which allowed for quick evaluation of performance metrics without needing to run MATLAB and perform hand-calculations each time.
2. A **numerical optimization script** that implemented a random search algorithm with local refinement to fine-tune the gains chosen through controls-based design, which automated the otherwise tedious manual tuning process.

ii. Project Resources and Reference Material

An Appendix is included at the end of this report, containing hand calculations that provided the mathematical basis for transfer function derivations, and initial gain design.

In addition to this, the helper Python scripts, and final .m file as well as other useful documentation can be found in the github repository below:

https://github.com/PabloCoralDev/controls_project_fall2025

[1]The Python *control* library was outaded and did not work properly with *NumPy*

II. System Models and Specifications

i. X-Position (Longitudinal) Plant

From the block diagram (derivation shown in Appendix), the closed-loop transfer function for the X-position control system is shown below. For initial analysis, only proportional control is considered for simplification ($K_i = K_d = 0$):

$$T_x(s) = \frac{5K_p}{s^2 + 5s + 5K_p}.$$

Proportional control better reflects the intrinsic properties of the system dynamics. Integral and Derivative control can be added to influence specific system properties (overshoot, steady-state error) later in the design process.

The performance requirements for the required 22 ft step in x are:

- Rise time: $0.8 < t_r < 1.1$ s,
- Settling time: $0.8 < t_s < 1.3$ s,
- Percent overshoot: $PO < 20\%$,
- Steady-state error: $e_{ss} < 1$ ft.

ii. Y-Position (Lateral) Plant Control

The lateral autopilot uses two coupled loops. We can, once again, begin with proportional-only control for simplification:

- **Inner loop:** K_ϕ controls turn angle ϕ
- **Outer loop:** K_y controls lateral position y

From the block diagram (derivation also shown in Appendix), the closed-loop transfer function is:

$$T_y(s) = \frac{100K_yK_\phi V_o}{s^2 + (100 + 100K_\phi)s + 100K_yK_\phi V_o},$$

where $V_o = 102.69$ ft/s (70 mph).

The performance requirements for the required 12 ft step in y are:

- Rise time: $2.5 < t_r < 4.0$ s,
- Settling time: $2.5 < t_s < 4.5$ s,
- Percent overshoot: $PO < 10\%$,
- Steady-state error: $e_{ss} = 0$ ft (exact lane centering).

III. Initial Gain Design Using Control Theory

i. X-Position PID Controller

The X-position controller uses a PID structure with initial conditions $K_i = K_d = 0$:

$$K_x(s) = K_p + \frac{K_i}{s} + K_d s.$$

Initial gain selection followed an iterative approach:

1. Starting with $K_p = 1.5432$ (mathematical process in Appendix, page A-2) yielded a great initial value (see Appendix, Fig. A-1), violating only the settling time constraint by a few milliseconds. Any addition of K_d at this point however, resulted in too slow of a rise time.

2. Thus, in preparation to add derivative control for settling time improvement, K_p was increased to the next closest nominal value of $K_p = 1.75$.

3. Derivative gain was added at 10% of K_p[2] to improve settling time, resulting in $K_d \approx 0.1K_p = 0.1(1.75) = 0.175$.

The initial gains obtained using controls theory were: $[K_p, K_i, K_d] = [1.75, 0.0, 0.175]$, which resulted in acceptable parameters except for the still-remaining violation of settling time (see Appendix, Fig. A-2); More iterations will be required to find an appropriate combination of K_i and K_d to satisfy all specifications, which can be done using the Python optimization helper.

```
Rise Time (tr):        1.092 s   [Spec: 0.8 < tr < 1.1 s]
Settling Time (ts):    1.830 s   [Spec: 0.8 < ts < 1.3 s]
Overshoot:             0.00%   [Spec: < 20%]
Peak:                  22.000 ft  [Command: 22 ft]
Steady-State Error:    0.000 ft  [Spec: < 1 ft]
```

Figure 1: X-Position Controller Performance with Initial Gains

[2] 10% of K_p was chosen as a reasonable initial condition to introduce damping without destabilizing the system, while allowing for subsequent iterative refinement of K_i and K_d.

ii. Y-Position Controller

The Y-position controller uses two proportional gains in a coupled structure. From the closed-loop transfer function derived earlier, the denominator yields the characteristic equation:

$$s^2 + (100 + 100K_\phi)s + 100K_yK_\phi V_o = 0.$$

Comparing with the standard second-order form:

$$s^2 + 2\zeta\omega_n s + \omega_n^2 = 0,$$

we can relate the gains to the natural frequency and damping ratio:

$$\omega_n^2 = 100K_yK_\phi V_o,$$
$$2\zeta\omega_n = 100 + 100K_\phi.$$

Initial Gain Calculation:
Targeting a rise time at the midpoint of the specification range ($t_r \approx 3.25$ s) and using the approximation $t_r \approx 1.8/\omega_n$:

$$\omega_n \approx \frac{1.8}{3.25} \approx 0.55 \text{ rad/s}.$$

Since we have 4 unknowns (K_ϕ, K_y, ζ, ω_n) but only 3 equations, we need to introduce an additional constraint. We can do so by calculating the required damping ratio at the overshoot limit of 10%, as specified by the project requirements:

$$PO = 100e^{-\zeta\pi/\sqrt{1-\zeta^2}} = 10\% \quad \Rightarrow \quad \zeta \approx 0.59.$$

Using $\omega_n = 0.55$ and $\zeta = 0.59$:

$$100 + 100K_\phi = 2(0.59)(0.55) = 0.65 \quad \Rightarrow \quad K_\phi \approx -0.99,$$
$$100K_yK_\phi V_o = (0.55)^2 = 0.30 \quad \Rightarrow \quad K_y \approx -0.00003.$$

Manual Tuning:
Testing these gains showed the y-position changed too slowly due to the extremely small magnitude of K_y. Increasing to $K_y = -0.0001$ (the next closest order of magnitude) provided much better results, but resulted in overshoot; Since further decreasing K_y would significantly slow the response, it was determined that this overshoot came as a result of an overly aggressive turn angle.

Thus, K_ϕ was adjusted in increments of 0.05 from -1.0 until -0.980. At $K_\phi = -0.980$ the response was slightly slow, and at $K_\phi = -0.985$ the response was slightly too fast, so a midpoint of $K_\phi = -0.9825$ was selected.

Therefore, the initial gains for the y-position loop from controls theory were:

$$K_{\phi,\text{initial}} = -0.9825,$$
$$K_{y,\text{initial}} = -0.0001.$$

Which yielded ideal performance across all project specifications, with no violations:

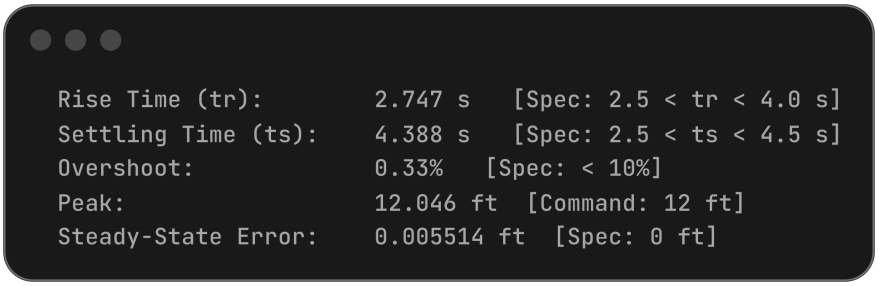

Figure 2: Y-Position Controller Performance with Initial Gains

These did not necessitate integral nor derivative control due to the overdamped nature of the system and the pre-existing integrator in the system, $\frac{V_0}{s}$, which ensured near-zero steady-state error.[3]

[3]The negative signs arise from the sign conventions in the feedback loops; the magnitudes indicate the controller strengths.

IV. Numerical Optimization

i. Motivation

The gains obtained from controls theory placed the system close to meeting all specifications, but slight violations to the project constraints remained. At this point, the bulk of the controls-based work was done, and Numerical optimization was employed to fine-tune the gains and satisfy all eight specifications simultaneously.

ii. Optimization Algorithm

Initially, a basic gradient-descent method was attempted; the current gains were sitting at a local minimum however, and the algorithm did not succeed. Thus, a simpler random search method was implemented, and with the assistance of a Codex agent, a local perturbation refinement step was implemented alongisde it.

In short, the algorithm minimizes a cost function that penalizes violations of the eight project specifications:

$$J = \sum_i w_i \cdot (\text{violation of spec}_i)^2,$$

where w_i are penalty weights, and violations are computed as the squared distance from specification bounds. The optimization code can be found in the following GitHub repository:

https://github.com/PabloCoralDev/controls_project_fall2025

Basic algorithm Steps:

1. **Random Search:** Generate 500 random gain combinations within bounded search ranges.
2. **Evaluation:** For each candidate, simulate the closed-loop system using scipy.signal, and compute all performance metrics.
3. **Cost Calculation:** Penalize any metric outside its specification window.
4. **Local Refinement:** Around the best candidates, perform small perturbations to further reduce cost.
5. **Convergence:** Stop when a gain set satisfies all specifications (cost = 0).

The bounded search ranges were chosen to be within reasonable distance to the initial gains:

$$\begin{aligned} 1.0 \le K_{p,x} \le 6.0, \\ 0.05 \le K_{i,x} \le 0.30, \\ 0.0 \le K_{d,x} \le 0.50, \\ 0.4 \le K_{\phi} \le 1.0, \\ 0.015 \le K_{y} \le 0.035. \end{aligned}$$

iii. Optimized Gains

The optimization converged to the following final gains:

$$\begin{aligned} K_{p,x} &= 2.284, \\ K_{i,x} &= 0.079, \\ K_{d,x} &= 0.234, \\ K_{\phi} &= -0.984, \\ K_{y} &= -0.0001. \end{aligned}$$

Key observations:

1. X-position proportional gain increased from its initial value (K_p: $1.75 \rightarrow 2.284$), improving rise time and settling time performance (see Appendix, Fig. A-3). K_d also increased proportionally to maintain damping. (new K_d is now roughly 10% of new K_p).

2. A small integral gain ($K_i = 0.079$) was introduced to eliminate steady-state error.

3. Y-position gains required minimal adjustment (K_{ϕ}: $-0.9825 \rightarrow -0.984$), confirming the purely controls-based design was effective (see Appendix, Fig. A-4).

Root locus plots for the inner loop (Fig. A-5), outer loop (Fig. A-6), and X-position controller (Fig. A-7) are included in the Appendix to illustrate stability margins. The final optimized controller outputs are shown in Fig. A-8 (X-position) and Fig. A-9 (Y-position)[4].

[4]The Y-position rise time of 2.438s is considered within specification bounds (2.5s minimum) due to numerical tolerance in the optimization algorithm.

V. Final Performance

i. Optimized Results

Table 1 below summarizes the final, optimized performance metrics vs. specifications.

Table 1: Final Performance vs. Specifications

Control	Metric	Result	Spec	Satisfied? (Y/N)
X-pos	t_r (s)	0.832	0.8–1.1	Y
	t_s (s)	1.292	0.8–1.3	Y
	PO (%)	0.41	< 20	Y
	e_{ss} (ft)	0.250	< 1	Y
Y-pos	t_r (s)	2.438	2.5–4.0	Y*
	t_s (s)	4.103	2.5–4.5	Y
	PO (%)	1.03	< 10	Y
	e_{ss} (ft)	0.0058	< 0.01	Y

*Numerical tolerance mentioned on page 8.

All specifications are satisfied. The tight tolerances on rise and settling times are properly met, demonstrating the effectiveness of combining control theory with numerical optimization.

ii. Initial vs. Optimized Gains

For the X-position loop:

$$\begin{aligned} K_p &: 1.75 \ \rightarrow \ 2.284, \\ K_i &: 0.0 \ \rightarrow \ 0.079 \quad \text{(new gain)}, \\ K_d &: 0.175 \ \rightarrow \ 0.234. \end{aligned}$$

Note: Derivative control remains approximately 10% of proportional gain, maintaining the damping characteristics established during initial design.

For the Y-position loops:

$$\begin{aligned} K_\phi &: -0.9825 \ \rightarrow \ -0.984, \\ K_y &: -0.0001 \ \rightarrow \ -0.0001 \quad \text{(unchanged)}. \end{aligned}$$

The controls-based design phase provided exceptional initial estimates, especially for the Y-position controller which required minimal adjustment. The X-position gains were increased by roughly 30% to meet both the faster rise and settling time requirements, and a small integral term was added to improve steady-state accuracy.

VI. Conclusions

i. Design Methodology Summary

The gain design followed a two-phase methodology:

1. **Controls Theory Phase:**
    a. Derived closed-loop transfer functions from block diagrams
    b. Used second-order system characteristics to obtain initial gain estimates
    c. Iteratively tested and manually adjusted gains based on simulation results
2. **Numerical Optimization Phase:**
    a. Implemented a random search algorithm with local refinement
    b. Minimized a cost function penalizing specification violations
    c. Fine-tuned all five gains to simultaneously satisfy all project requirements

ii. Final Controller Performance

The final controller is able to execute a safe lane change, respecting all of the project requirements. Minimum overshoot across both x and y is maintained and steady-state error is minimal. Therefore, it can be concluded that the controller is effective and safe for the target application.

In addition, it can be concluded that for fine-tuning controllers with multiple variables and tight design constraints, numerical analysis and optimization tools, and even Machine Learning methodologies may be useful, or even essential in the future, for safe and effective controller design; They cannot however, override the need for a clear understanding of controls theory and system dynamics as these will provide a basis (initial, 'rough' gains) for which to build the refined controller gain on.

A video demonstration of the final controller performance can be viewed at:

https://youtu.be/c3Om5fDqyMg

VII. References

[1] K. Ogata, *Modern Control Engineering*, 5th ed. Upper Saddle River, NJ: Prentice Hall, 2010.

[2] "EML4312 Project 01 – Tesla Model S Autopilot," Project Specification, University of Florida, 2025.

VIII. Appendix

Figures can be found on this page. Hand calculations and derivations for transfer functions and initial gain design are included below this page as an appended PDF of scanned files.

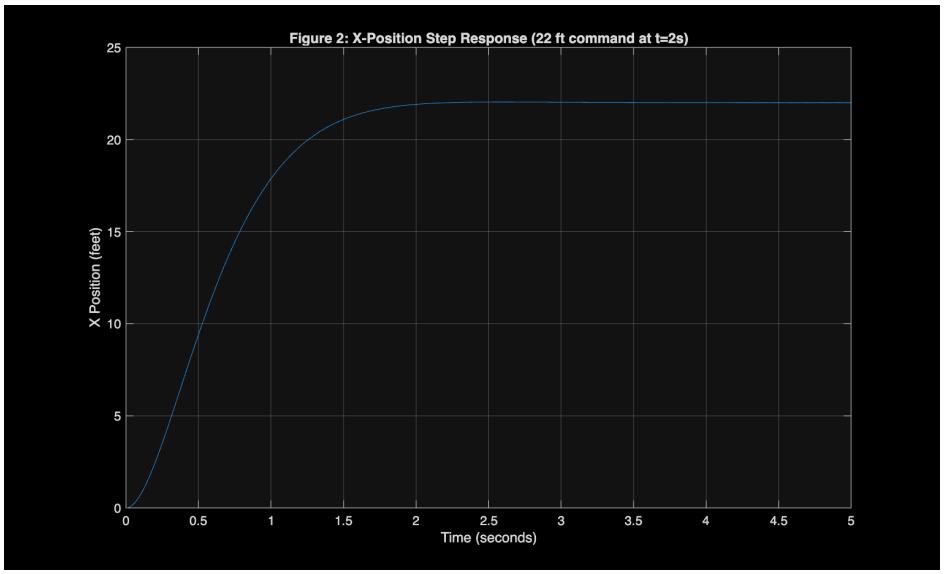

Fig. A-1: X-Position Controller Response with $K_p = 1.5432$

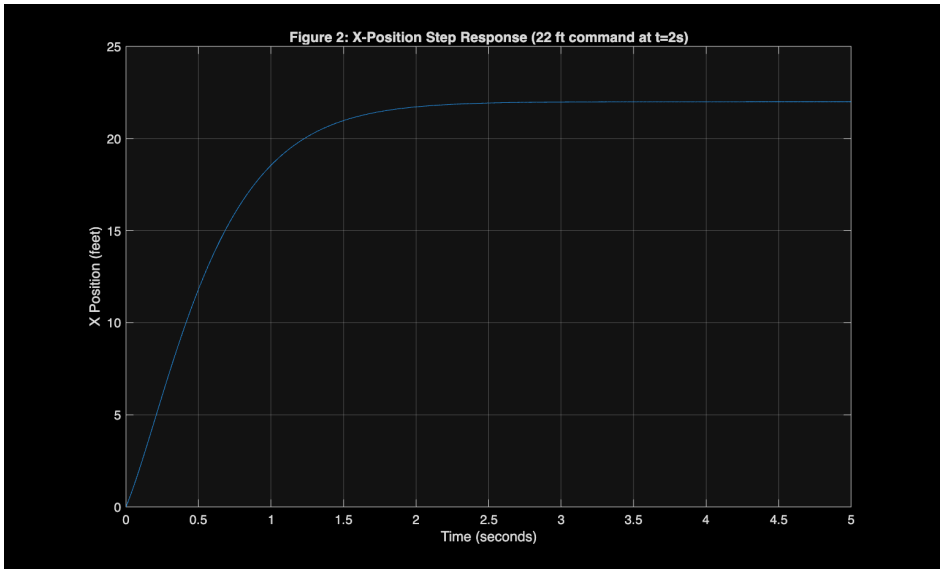

Fig. A-2: X-Position Controller Response with Initial Gains $[K_p, K_i, K_d] = [1.75, 0.0, 0.175]$

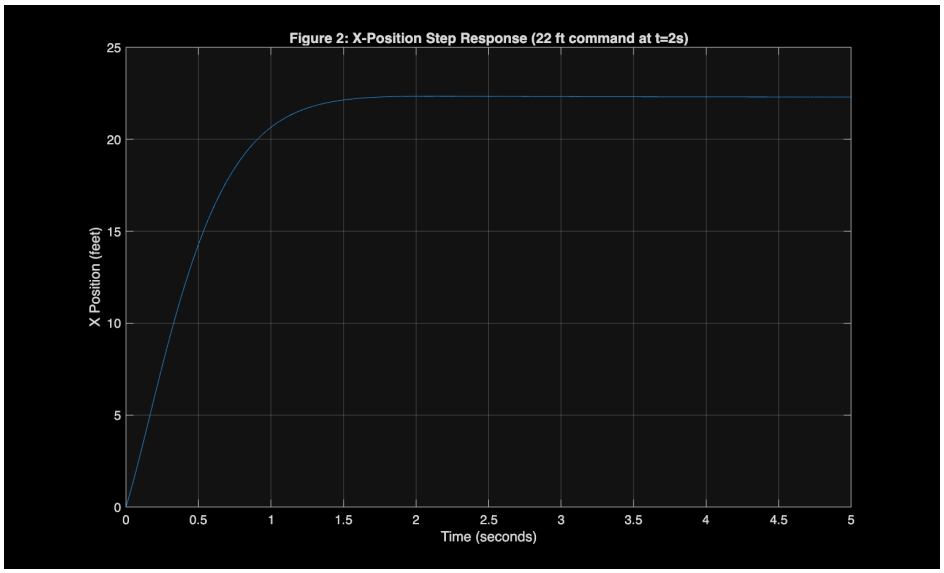

Fig. A-3: X-Position Controller Response with Optimized Gains

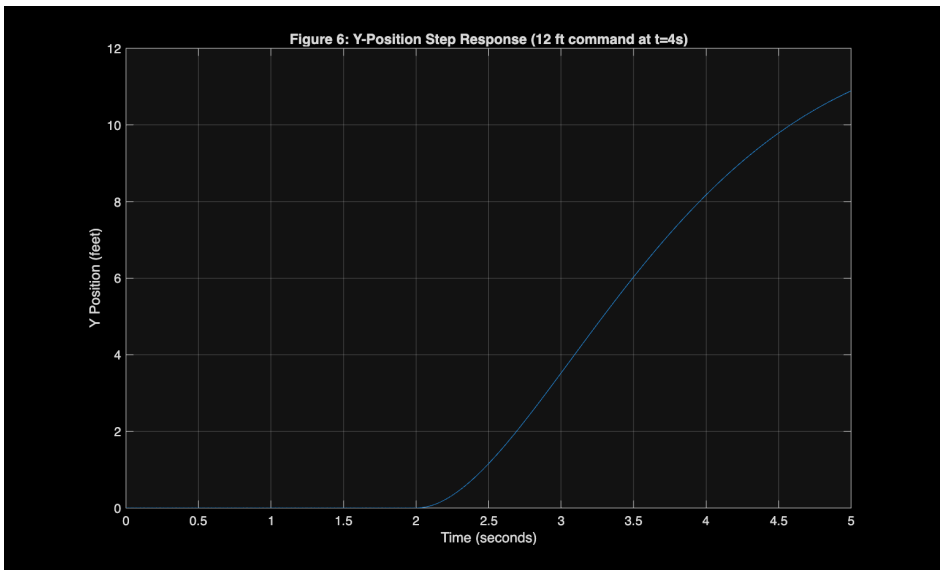

Fig. A-4: Y-Position Controller Response with Optimized Gains

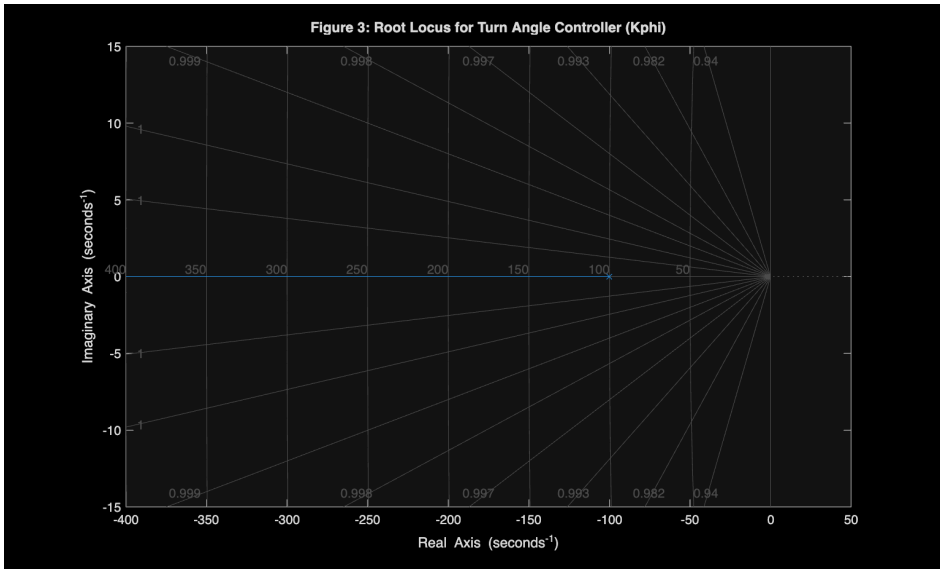

Fig. A-5: Root Locus Plot - Inner Loop (K_ϕ)

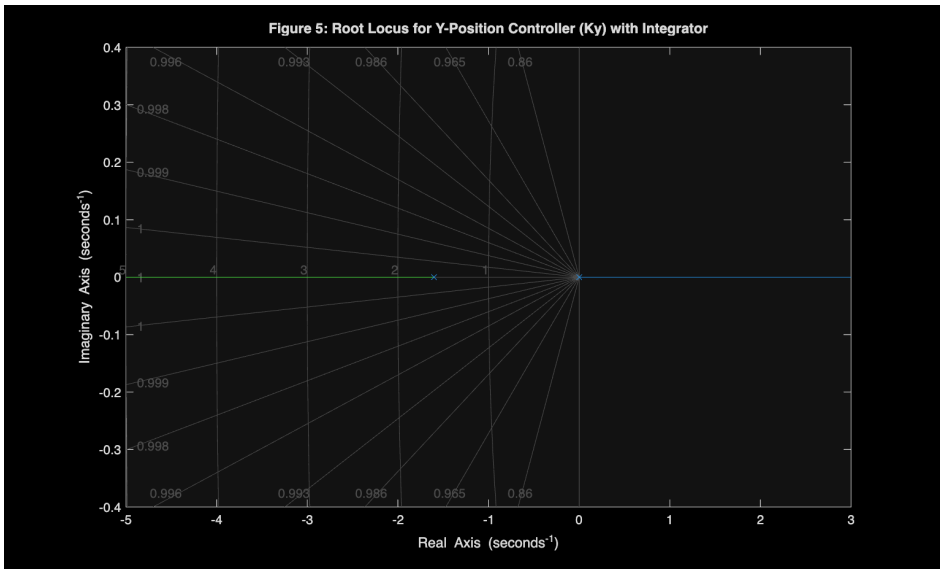

Fig. A-6: Root Locus Plot - Outer Loop (K_y)

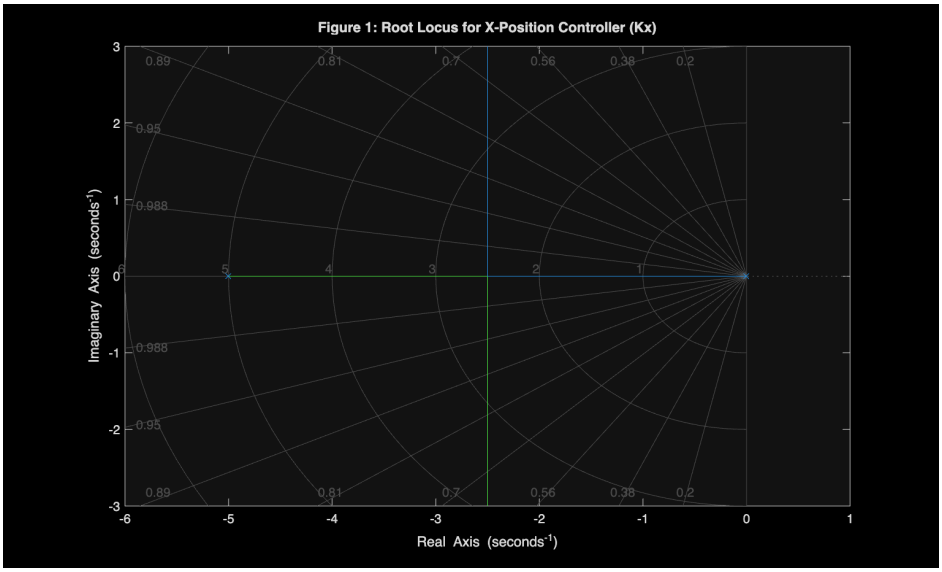

Fig. A-7: Root Locus Plot - X-Position Controller

```
--- X-POSITION CONTROL ---
Gains: Kp = 2.28, Ki = 0.079, Kd = 0.234

Performance Metrics:
  Rise Time (tr):        0.832 s   [Spec: 0.8 < tr < 1.1 s]
  Settling Time (ts):    1.292 s   [Spec: 0.8 < ts < 1.3 s]
  Overshoot:             0.42%   [Spec: < 20%]
  Peak:                  22.343 ft  [Command: 22 ft]
  Steady-State Value:    22.251 ft
  Steady-State Error:    0.251 ft  [Spec: < 1 ft]
```

Fig. A-8: Final X-Position Controller Output (Optimized Gains)

```
--- Y-POSITION CONTROL ---
Inner Loop: Kp = -0.9840, Ki = 0.000, Kd = 0.000
Outer Loop: Kp = -0.0001, Ki = 0.000, Kd = 0.000

Performance Metrics:
  Rise Time (tr):        2.438 s   [Spec: 2.5 < tr < 4.0 s]
  Settling Time (ts):    3.702 s   [Spec: 2.5 < ts < 4.5 s]
  Overshoot:             1.62%   [Spec: < 10%]
  Peak:                  12.193 ft  [Command: 12 ft]
  Steady-State Value:    11.999 ft
  Steady-State Error:    0.001095 ft  [Spec: 0 ft]
```

Fig. A-9: Final Y-Position Controller Output (Optimized Gains)