Modified PD Controller to Stablize Tethered Drone

Khang Pham (A20464881)

March 22, 2025

Course: MMAE 540

Professor: Matthew Spenko

Contents

1 Introduction

Rainforests are critical ecosystems that require effective monitoring to support conservation policies. Deploying sensor packages using drones is a practical solution; however, dense canopies block radio signals, complicating communication. A simple approach is to fly a drone at high altitudes to maintain line-of-sight communication while deploying sensors using a long rope. However, this introduces instability, reducing precision during deployment.

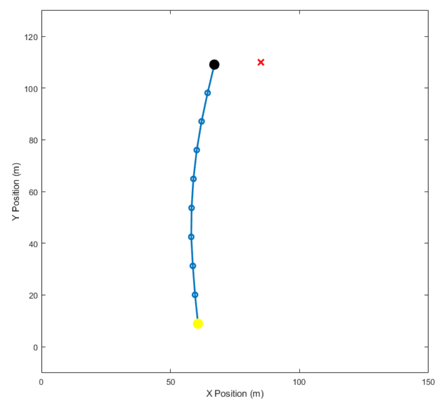


Figure 1: Simulated Environment.

This report explores two potential solutions, both include adding a smaller drone at the rope's end to assist in stabilizing the sensor package. The larger drone, responsible for carrying the system, is referred to as the ***mothership***, while the smaller drone tasked with stabilization is called the ***minion***.

2 Simulation

Assuming the ***mothership***'s thrust is sufficient to carry the entire system, its controller is modeled as a simple spring-damper system to a predetermined ***target***, with additional vertical thrust to counter the total weight of the system. The rope is simulated using 10 particles connected by spring-damper constraints, where the spring constant is set high to mimic a non-elastic rope. A damping factor is included to simulate drag and to cancel rounding errors during iterations. Both the ***mothership*** and the ***minion*** are simulated as point masses and are considered the first and last particles of the rope, respectively.

3 PD Controller

3.1 Side-Thrust Design

The first approach involves the ***minion*** having only horizontal thrust capabilities (fig. 2). A simple Proportional-Derivative (PD) controller [1] is applied to track the horizontal position of the ***mothership*** (eq. 2). This method gives promising results, as shown in Figure 3, where the ***minion*** tracks the ***mothership***'s position accurately.

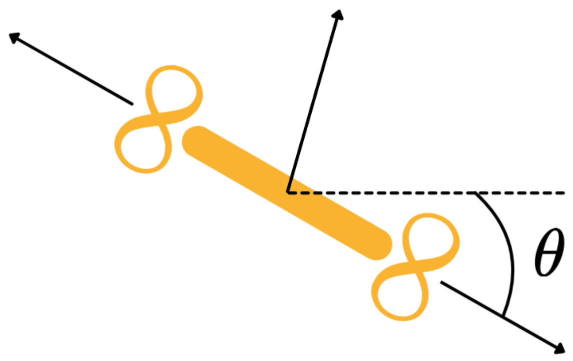


Figure 2: Horizontal-thrust design.

$$\tilde{x} = x_{\boldsymbol{mothership}} - x_{\boldsymbol{minion}} \tag{1}$$

$$F = K_p\tilde{x} + K_d\dot{\tilde{x}} \tag{2}$$

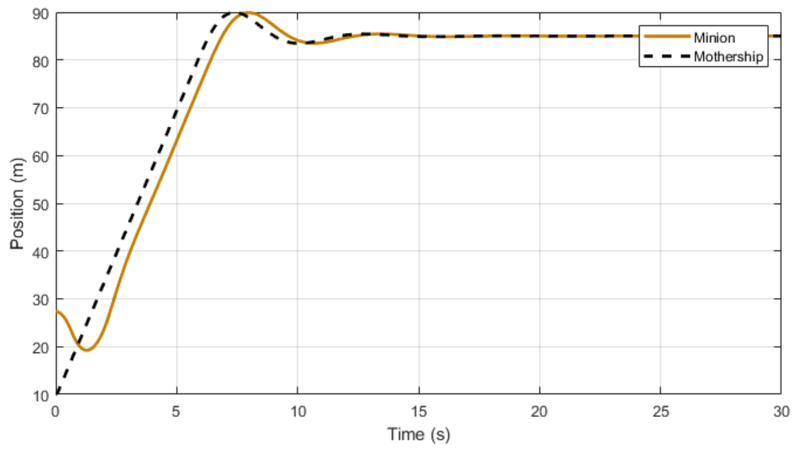


Figure 3: X position of horizontal-thrust ***minion*** and ***mothership*** over Time

However, this approach assumes that the angle θ (fig. 2)remains relatively small. Since the ***minion*** is treated as a point mass in the simulation (sec. 2), no torque is induced by environmental forces. In the event of disturbances causing a larger θ, this ***minion*** design cannot correct itself, leading to a loss of control in the horizontal plane as θ increases.

3.2 Vertical-Thrust (Traditional) Design

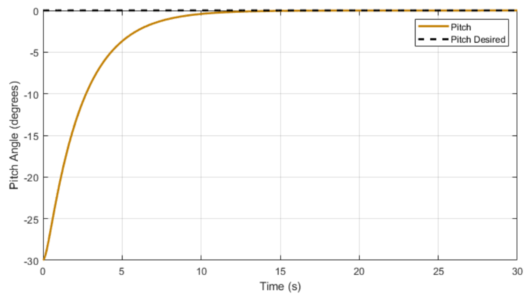
To gain control over the angle θ, a traditional vertical-thrust drone design is more sufficient. For simplicity in the simulation, instead of calculating individual thrusts for each rotor, only the total thrust and torque are considered.

In many control design cases, the error is computed relative to the final desired state. However, this approach does not work well for drones due to the coupling of position and angular dynamics. If the final desired state is defined as $x_{\boldsymbol{minion}} = x_{\boldsymbol{mothership}}$ (similar to the horizontal-thrust case in sec. 3.1) and $\theta = 0$, a control law that accounts for both position and angular error is required. The control law can be expressed as:

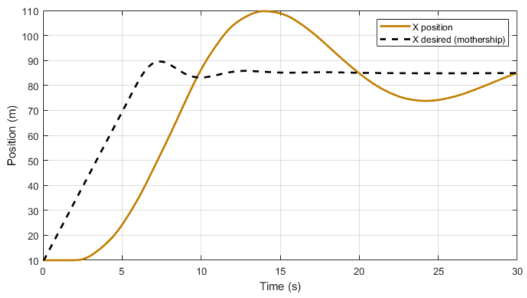
$$\mathbf{F} = \begin{bmatrix} F \\ \tau \end{bmatrix} = \mathbf{K_p} \begin{bmatrix} \tilde{x} \\ -\theta \end{bmatrix} + \mathbf{K_d} \begin{bmatrix} -\dot{x}_{\boldsymbol{minion}} \\ -\omega \end{bmatrix} \tag{3}$$

In the simulated environment, the angle θ is quickly corrected (fig. 5a), leading to the ***minion*** losing control over its x position, causing the system to behave like a pendulum (fig. 5b). Note that to prevent false position corrections, set the thrust to zero when $F = \infty$. Although performance can be improved with a large PD gain for x and a small PD gain for θ, the coupling between angle and position is not fully addressed.

An $x_{\text{threshold}}$ is introduced to address the coupling issue. If $|\tilde{x}|$ exceeds this threshold, the ***minion*** accelerates forward to catch up with the ***mothership*** ($\theta < 0$). Vice versa, when $|\tilde{x}|$ is less than the threshold, the pitch angle θ is positive and the ***minion*** decelerates. Both



(a) Pitch angle over Time



(b) X position over Time

Figure 4: Simulation results while using control law in eq. 3.

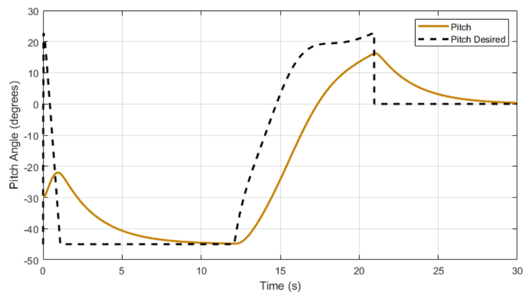
of these requirements are satisfied with the expression shown in eq. 4. The complete control law is shown in eq. 5, and the simulation result is shown in fig. 5.

$$\theta_{desired} = \mathbf{K}_\theta \left(x_{\text{threshold}} - \tilde{x} \right) \tag{4}$$

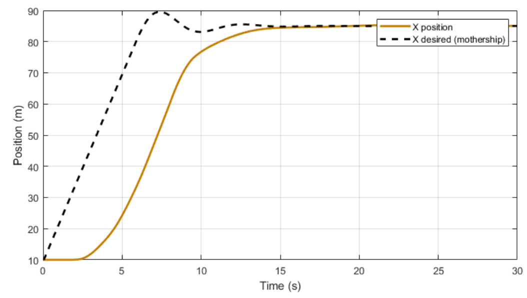
$$\mathbf{F} = \begin{bmatrix} F \\ \tau \end{bmatrix} = \mathbf{K_p} \begin{bmatrix} \tilde{x} \\ \theta_d - \theta \end{bmatrix} + \mathbf{K_d} \begin{bmatrix} -\dot{x} \\ -\omega \end{bmatrix} \tag{5}$$

Note that the steady-state pitch angle (θ) under this control law will not by default converge to $0°$ (eq. 6), but it can be manually set to $0°$ once x reaches a steady state.

$$\theta_{\text{steady}} = K_\theta x_{\text{threshold}} \tag{6}$$



(a) Pitch angle over Time



(b) X position over Time

Figure 5: Simulation results while using control law in eq. 4.

4 Discussion

This report analyzed two designs for the ***minion*** in a tethered drone system. The horizontal-thrust design demonstrated promising performance with a very simple control law, effectively tracking the ***mothership***'s position. However, its primary limitation is the inability to correct its pitch angle (sec. 3.1). The vertical-thrust design with a modified PD control law stabilizes both the ***minion***'s position and pitch angle. This approach successfully guided the ***minion*** to the desired state, addressing the pitch correction issue observed in the horizontal-thrust design.

Future improvements include developing a more accurate simulation by modeling drag as proportional to the relative velocity with the wind. Additionally, simulating the ***minion*** as a plank rather than a point mass would provide a more realistic representation of its dynamics. [2]

Acknowledgement

Large language models ChatGPT 4o from Open AI [3] are used for general research purposes and fixing syntax while programming.

References

1. Lynch, K. M. & Park, F. C. *Modern Robotics: Mechanics, Planning, and Control* ISBN: 9781107156302. `https://hades.mech.northwestern.edu/index.php/Modern_Robotics#Book` (Cambridge University Press, 2017).
2. Downey, A. B. *Modeling and Simulation in Python* ISBN: 9781718502161. `https://allendowney.github.io/ModSimPy/index.html` (No Starch Press, 2023).
3. OpenAI. *ChatGPT 4o* `https://openai.com/chatgpt`.