Investigation of Closed-Loop Control Strategies for a Martian Quadcopter

Rohit Bhattiprolu

Bloomfield Hills High School

Abstract

With the increasing focus on human and robotic exploration of Mars, the development of extraterrestrial rotorcraft has accelerated. Inspired by the success and limitations of NASA's Ingenuity Helicopter, this research investigates closed-loop control strategies, specifically Proportional-Derivative (PD) control, for application in a hypothetical two-dimensional Martian quadcopter. First the paper applies a PD control law to a basic spring-mass damper system. Then the paper models quadcopter dynamics and develops a PD controller to bring the quadcopter to a desired state. Using the MarsBird-VII quadcopter design as a baseline, this study employs simulations to compare quadcopter dynamics and different PD gains in both Earth and Martian environments. The results demonstrate the necessity for environment-specific tuning of control gains.

Introduction

NASA’s Ingenuity Helicopter arrived on Mars on the Perseverance Rover in February 2021. After 72 flights, the first of its kind technology demonstration ended in January 2024 [1]. The Ingenuity helicopter was an outstanding success, proving the feasibility of extraterrestrial flights. The Ingenuity helicopter wasn’t perfect though. Its navigation system relied heavily on detailed terrain as reference points, so when it was instructed to land on seemingly featureless terrain it crashed [2]. There was also a communications blackout between the helicopter and Earth around the time of the crash, which could’ve also contributed to the crash. Because either of these two things could’ve both caused the crash, the only way to make sure something like that doesn’t happen in the future is to fix both problems. A revised closed-loop control system and navigation system could have prevented the crash.

The Ingenuity helicopter implemented a closed-loop control system, which means it self-adjusted to discrepancies in its operations, helping it work without human intervention, a helpful feature given the communication delay between Earth and Mars. Closed loop control systems make automatic adjustments to the system based on the error between the desired and current states of the system. The Ingenuity helicopter’s control system combines state estimates with commands from a human operator to actuate the vehicle. The Guidance system provides the trajectory of the flight path. The Navigation system determines where the vehicle is located with respect to its environment and desired trajectory based on state estimates. The Control system shapes the actuator input to direct the vehicle towards the desired trajectory based on a control law [3].

Several revised navigation systems have also been proposed. Using satellites and X-ray pulsars, which are magnetized neutron stars that emit X-ray waves, exact positions on the ground can be triangulated. While this seems promising, this technology is still theoretical [4,5].

Ingenuity's success as the first extraterrestrial helicopter has inspired many missions like it, not only in the Martian atmosphere but in other atmospheres as well. One of these projects is the Sample Recovery Helicopter, which will potentially be used in NASA's Mars Sample Return (MSR) program. Some budget concerns might halt the development of the SRH, but if completed, it would build upon the success of the Ingenuity helicopter. The new mission would aim to collect samples of Martian soil with an updated navigation system [6]. Rotorcraft like the Ingenuity aren't the only potential extraterrestrial aircraft, aero robots like Y4-TR are plane-like and can take off vertically like a rotorcraft while being able to cruise like a plane [7]. Entomopter-based flight has also been explored, which utilizes a flapping motion to create lift [8].

The PlanetarY Telemetric Helicopter for Investigation and Analysis (PYTHIA) is another Mars helicopter project that is focused on exploring Mars's lava tubes. A quadcopter design, contrary to the single-rotor design on the Ingenuity helicopter, was chosen for the PYTHIA mission. It will also be significantly bigger than the Ingenuity helicopter [9].

The MarsBird-VII is a potential Mars quadcopter design that can fly autonomously and collect samples [10]. Trajectories are received from Earth, but it uses attitude sensors, lidar, and several cameras as reference for navigation, which fixes the Ingenuity Helicopter's problem of using major landmarks for navigation. This quadcopter will be used as a reference for this paper.

To motivate further research in this area, this paper investigates closed-loop control strategies that can be implemented in extraterrestrial rotorcraft. In this vein, Section 1 introduces this field of extraterrestrial rotorcraft and current works, Section 2 frames the specific problem this paper will be investigating and introduces key concepts being explored through this paper, Section 3 implements concepts introduced in Section 2 in a simulated spring-mass damper system, Section 4 applies the same concepts into a simulated two-dimensional quadcopter system, Section 5 simulates the system in a Martian environment with the specs of the MarsBird-VII and discusses the results.

Problem Statement

Quadcopters are a type of rotorcraft that utilizes four rotors for propulsion. Quadcopter flight controllers include PID controllers which utilize closed-loop control to correct errors due to external disturbances. A PID controller is composed of three different components that all contribute to the acronym PID. The "proportional" component of the PID controller is responsible for identifying and correcting present errors. The "derivative" component predicts future errors and can counter the proportional component to avoid them. The main goal of integral control is to minimize steady-state error, which is the difference between the goal state and the current state when the system has reached a constant state. [11]. Mathematically, PID control is in the form,

$$u = k_p e(t) + k_i \int e(t) + k_d \dot{e}(t) \; (1)$$

Where k_p, k_i, and k_d are gains which can be adjusted depending on the amount of control needed.

As illustrated in Figure 1, the system state is used to calculate an error value, which is then used in a PID control law which results in an input vector u(t). This input vector is implemented by the system – in our case quadcopter actuators – to move the system to the desired trajectory. This process repeats until the desired trajectory is reached.

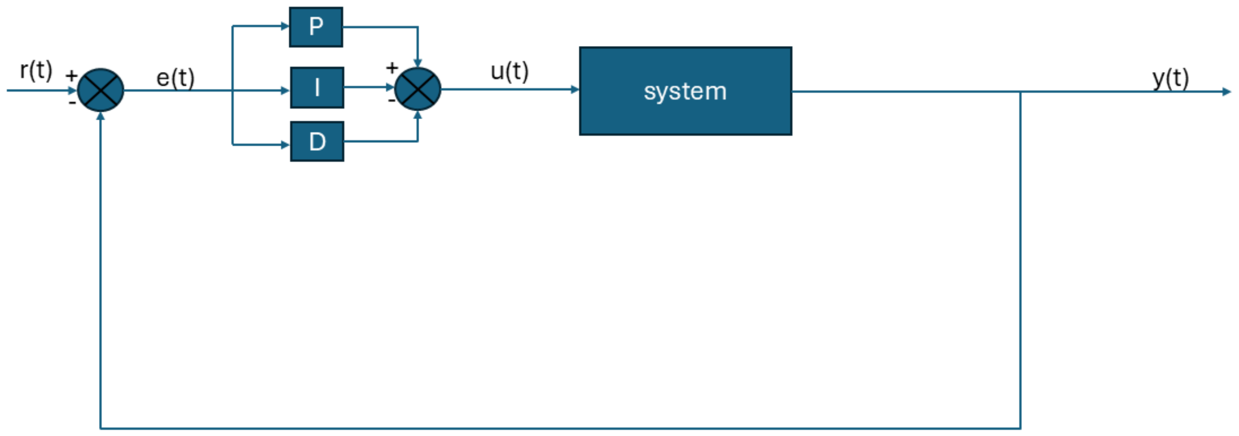


Figure 1: PID Control Block Diagram

Linear systems are commonly described mathematically in the form:

$$\dot{\vec{x}} = A\vec{x} + B\vec{u} \; (2)$$

Where $\vec{x}$ is the current system state and $\vec{u}$ is the input vector. In an open-loop system, $\vec{u}$ would be user input, but in a closed loop system it is replaced by the result of the control law [12].

State space is a set of characteristics that define the dynamics of a system [13]. In the prior form, $\vec{x}$ is a state vector that represents all the state variables in the state space of that given system. State space models are commonly used to simulate engineering systems and will be used throughout this paper. System stability is how a system maintains a certain behavior over time [14]. A more complex analysis of state space is required to compute system stability, so for the purposes of this paper this concept will be overlooked. Through this paper, Proportional-Derivative (PD) control methods will be explored and implemented on a 2D Martian quadcopter simulation. The gains will be selected appropriately in both Martian and Earth environments.

Controlling a Spring-Mass Damper System

Before implementing a control system in a drone, control system fundamentals must be explored using a simpler system, like a spring mass damper. A spring mass damper, shown in Figure 2, is a system consisting of a mass attached to a spring and a damper which counteracts the spring's effect.

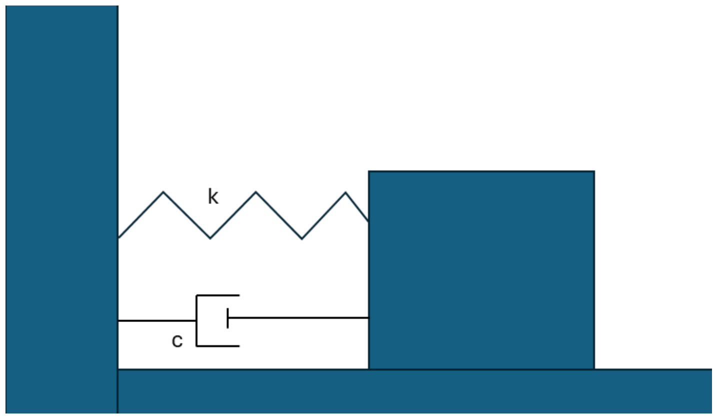


Fig. 2: Spring Mass Damper System

With no external forces or friction, the system contains four forces, as illustrated in Figure 3, the spring force, the damper force, gravity, and a normal force from the ground. Gravity and the normal force cancel each other out and won't affect the dynamics of this system.

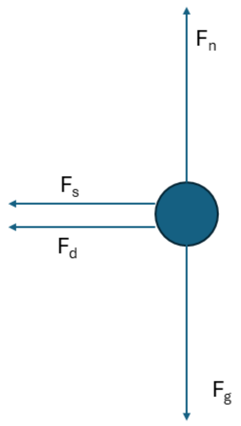


Figure 3: Spring Mass Damper Free-Body Diagram

The laws that determine the spring and damper forces respectively are,

$$F_s = -kx \ (3)$$

$$F_d = -c\dot{x} \ (4)$$

where k is the spring constant and c is the damper constant. Newton's second law of motion states that an object's acceleration is dependent on its mass and the net force acting upon it, mathematically it is represented as,

$$\Sigma F = m\ddot{x} \ (5)$$

Substituting equations 3-4 into 5 yields the equation of motion.

$$m\ddot{x} = -kx - c\dot{x} \ (6)$$

Rearranging the equation of motion to define $\ddot{x}$ results in:

$$\ddot{x} = -\frac{k}{m}x - \frac{c}{m}\dot{x} \ (7)$$

State space form is a succinct way to simulate engineering systems. Therefore, a state vector and matrices must be defined according to the equation of motion. The state vector $\vec{x}$ is:

$$\vec{x} = \begin{bmatrix} x \\ \dot{x} \end{bmatrix} \ (8)$$

Which, when derived, results in

$$\dot{\vec{x}} = \begin{bmatrix} \dot{x} \\ \ddot{x} \end{bmatrix} (9)$$

To map $\vec{x}$ to $\dot{\vec{x}}$, it needs to be in the form $\dot{\vec{x}} = \mathrm{A}\vec{x}$.

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & -\frac{c}{m} \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \end{bmatrix} (10)$$

Now that the system is defined, a control law can be implemented. Using a PD control law allows the control designer to influence the behavior of the system by changing the gains rather than adding a whole new spring or damper. A PD control law is the same as a PID control law, but without the integral term.

$$u = k_p x + k_d \dot{x} \ (11)$$

The open loop system is:

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & -\frac{c}{m} \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \end{bmatrix} + \begin{bmatrix} 0 \\ -\frac{1}{m} \end{bmatrix} u \ (12)$$

u is the input force the $-\frac{1}{m}$ term is there because u is a force acting on the mass and acceleration is obtained by dividing force by mass. The equation of motion with the open-loop input is

$$\ddot{x} = -\frac{k}{m}x - \frac{c}{m}\dot{x} - \frac{1}{m}u \ (13)$$

Substituting equation 11 into equation 13 results in an equation of motion,

$$\ddot{x} = -\frac{k}{m}x - \frac{c}{m}\dot{x} - \frac{1}{m}\left(k_p x + k_d \dot{x}\right) (14)$$

With some algebraic manipulation the equation of motion is

$$\ddot{x} = \left(-\frac{k}{m} - \frac{k_p}{m}\right)x + \left(-\frac{c}{m} - \frac{k_d}{m}\right)\dot{x} \ (15)$$

Which, in matrix form is,

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{k}{m} - \frac{k_p}{m} & -\frac{c}{m} - \frac{k_d}{m} \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \end{bmatrix} \ (16)$$

Now the system can be controlled by modifying the gains.

To simulate the system, an integration scheme is needed to calculate the system state at every time instance. Euler's method is used and explained below. The definition of derivative is,

$$\lim_{t \to 0} \dot{\overrightarrow{x_k}} = \frac{\Delta \vec{x}_k}{\Delta t} (17)$$

By multiplying both sides by Δt on both sides, this leads to an expression for the change in state at time k.

$$\Delta \vec{x}_k = \dot{\overrightarrow{x_k}} \triangle t \ (18)$$

The next state can be calculated by adding the change in state to the current state,

$$\overrightarrow{x_{k+1}} = \overrightarrow{x_k} + \Delta \overrightarrow{x_k} \ (19)$$

Substituting equation 18 into 19 the update step is found to be,

$$\overrightarrow{x_{k+1}} = \overrightarrow{x_k} + \dot{\overrightarrow{x_k}} \triangle t \ (20)$$

Plugging in $\dot{\vec{x}} = A\vec{x}$ into equation 20,

$$\overrightarrow{x_{k+1}} = \overrightarrow{x_k} + A\,\overrightarrow{x_k} \triangle t \ (21)$$

This is Euler's method, where you can estimate the next point of a function using the previous point and chosen time step. In most cases, a simulation time step of $\triangle t \leq 0.01$ seconds is sufficient for accurate numerical integration. However, the integration time step differs on a case-to-case basis based on system dynamics. In this paper, a simulation time step of 0.01 seconds is chosen.

When simulated, a mass $m = 1\ kg$, has been chosen, and will remain constant for every simulation discussed. A system with a spring constant $k = -0.5$ a dampening constant $c = 0$, and gains $k_p = k_d = 0$, will continuosly push the mass away from the origin exponentially because there is no damping force and the spring is pushing away as illustrated in Figure 4.

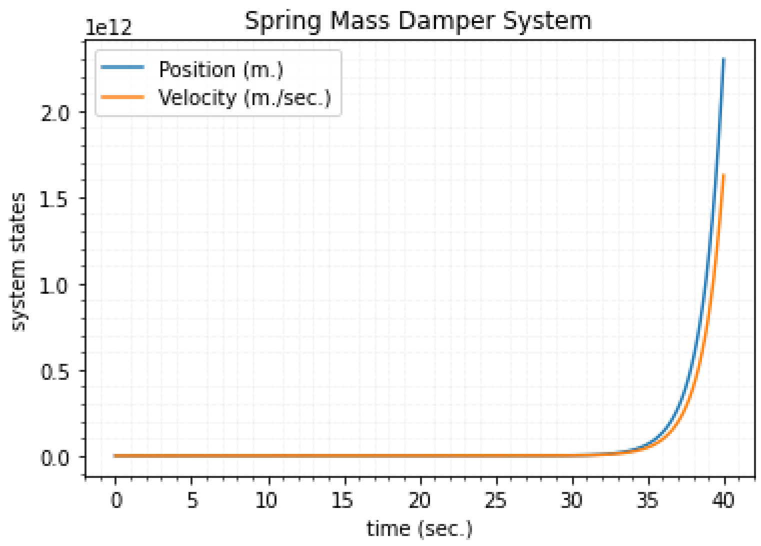


Figure 4: Spring with negative stiffness and no control

With the same spring and damping constant, but adding gains $k_p = 1$ and $k_d = 0.1$, the system reaches the origin in 40 seconds in Figure 5.

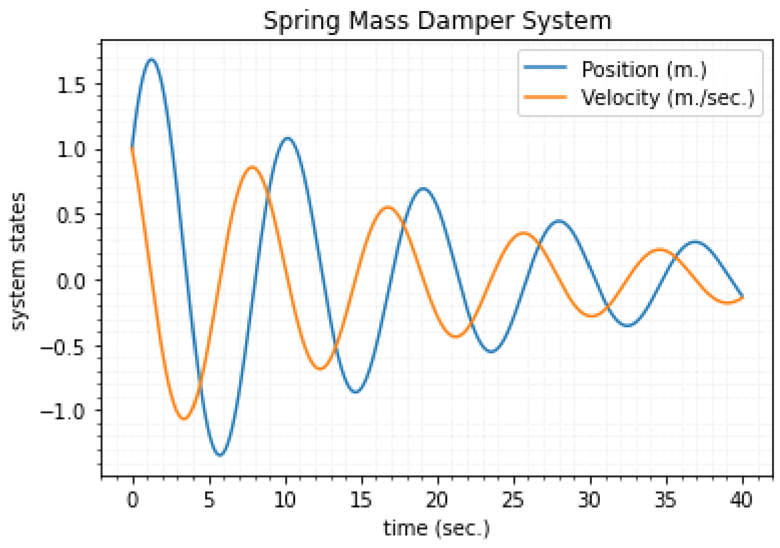


Figure 5: Spring with negative stiffness and some control

But by increasing the derivitave component to 0.5, the system becomes stable much faster, reaching the origin in roughly 20 seconds, as shown in Figure 6.

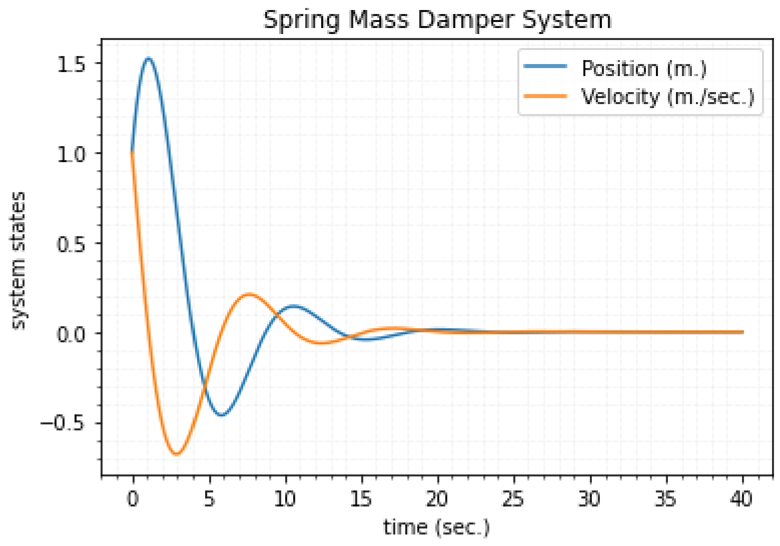


Figure 6: Spring with negative stiffness and higher gains

Simulating and Creating a Control Law for a 2-D Quadcopter

Now that a basic PD controller has been made and simulated in a spring-mass damper, the same concepts can be used in a hypothetical quadcopter system. Using the two-dimensional quadcopter model illustrated in Figure 7, the forces and torques are easily identified. F_r is the thrust on the right side of the quadcopter and F_l is the thrust on the left. Using Newton's 2nd Law gets the equations of motion for the quadcopter

$$\sum \vec{F} = \vec{F_r} + \vec{F_l} = \vec{f} = m\vec{a} \; (22)$$

$$\sum \tau = \tau = L(F_r - F_l) = I_{xx}\alpha \; (23)$$

Breaking down the forces and torques in the z and y axis and writing them as differential equations results in

$$-f sin(\Phi) = m\frac{d^2y}{dt^2} \; (24)$$

$$f cos(\Phi) - mg = m\frac{d^2z}{dt^2} \; (25)$$

$$\tau = I_{xx}\frac{d^2\Phi}{dt^2} \; (26)$$

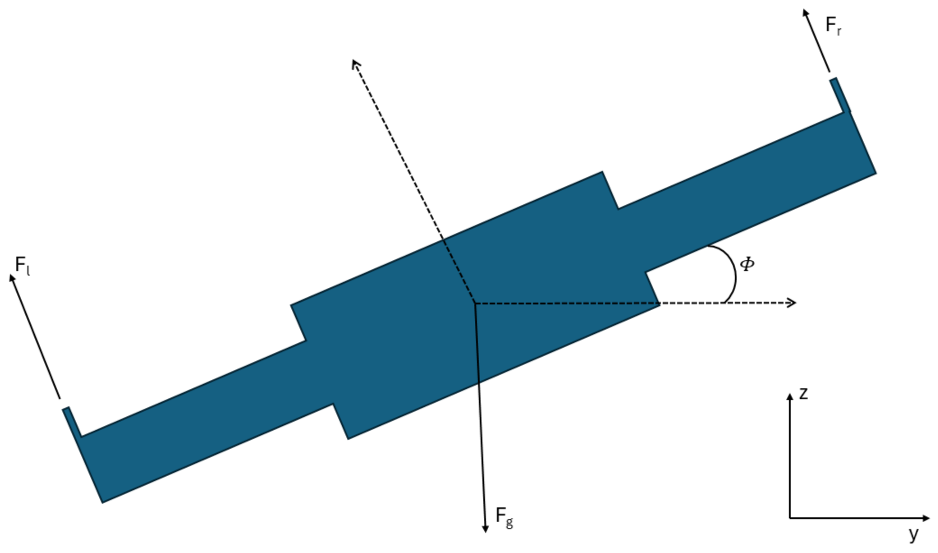


Figure 7: 2-D Drone System

There are 3 degrees of freedom so there is a 6-D state space. The state-vector

$$\vec{x} = \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ x_4(t) \\ x_5(t) \\ x_6(t) \end{bmatrix} = \begin{bmatrix} y(t) \\ z(t) \\ \Phi(t) \\ \dot{y}(t) \\ \dot{z}(t) \\ \dot{\Phi}(t) \end{bmatrix} \; (27)$$

Which when derived is,

$$\dot{\vec{x}} = \begin{bmatrix} \dot{x_1}(t) \\ \dot{x_2}(t) \\ \dot{x_3}(t) \\ \dot{x_4}(t) \\ \dot{x_5}(t) \\ \dot{x_6}(t) \end{bmatrix} = \begin{bmatrix} \dot{y}(t) \\ \dot{z}(t) \\ \dot{\Phi}(t) \\ \ddot{y}(t) \\ \ddot{z}(t) \\ \ddot{\Phi}(t) \end{bmatrix} \; (28)$$

In an open loop model the input vector,

$$\vec{u} = \begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix} = \begin{bmatrix} f(t) \\ \tau(t) \end{bmatrix} \quad (29)$$

Using the state vector for reference, equations 24-26 can be rewritten.

$$-u_1 \sin(x_3) = m\dot{x_4} \quad (30)$$

$$u_1 \cos(x_3) - mg = m\dot{x_5} \quad (31)$$

$$u_2 = I_{xx}\dot{x_6} \quad (32)$$

Rearranging equations 30-32 in terms of the acceleration gets,

$$\dot{x_4} = -\frac{1}{m} u_1 \sin(x_3) \quad (33)$$

$$\dot{x_5} = \frac{1}{m} u_1 \cos(x_3) - g \quad (34)$$

$$\dot{x_6} = \frac{1}{I_{xx}} u_2 \quad (35)$$

These can be plugged into the equation of motion resulting in,

$$\dot{\vec{x}} = \begin{bmatrix} \dot{x_1}(t) \\ \dot{x_2}(t) \\ \dot{x_3}(t) \\ \dot{x_4}(t) \\ \dot{x_5}(t) \\ \dot{x_6}(t) \end{bmatrix} = \begin{bmatrix} x_4(t) \\ x_5(t) \\ x_6(t) \\ -\frac{1}{m} u_1\,(t)\sin\left(x_3(t)\right) \\ \frac{1}{m} u_1(t)\cos\left(x_3(t)\right) - g \\ \frac{1}{I_{xx}} u_2(t) \end{bmatrix} \quad (36)$$

While PD control can be implemented in non-linear systems, it is easier to create the control law using linearized dynamics.

The linearized system must be in the form $\dot{\vec{x}} = A\vec{x} + B\vec{u}$, around the hover position, where $\Phi = 0, f = 0,$ and $\tau = 0$. Using a first order Taylor Polynomial, $\sin(\Phi) \approx 0$ and $\cos\,(\Phi) \approx 1$.
With the linearized system matrices the equation of motion is,

$$\begin{bmatrix} x_4(t) \\ x_5(t) \\ x_6(t) \\ -gx_3(t) \\ \frac{1}{m}u_1(t) - g \\ \frac{1}{I_{xx}} u_2(t) \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & -g & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ x_4(t) \\ x_5(t) \\ x_6(t) \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ -\frac{1}{m} & 0 \\ 0 & \frac{1}{I_{xx}} \end{bmatrix} \begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix} \quad (37)$$

Now that the system is linearized a PD control law can be made. The control law needs to bring the drone to a desired goal state,

$$\begin{bmatrix} \Phi_{desired} \\ \dot{\Phi}_{desired} \\ \ddot{\Phi}_{desired} \\ z_{desired} \\ \dot{z}_{desired} \\ \ddot{z}_{desired} \end{bmatrix} \text{ (38)}$$

The controller must be in the form $u = k_p e + k_d \dot{e}$, where the error e is the current state subtracted from the desired state. The control law is,

$$\ddot{z}_{commanded} = \ddot{z}_{desired} + k_{pz}(z_{desired} - z) + k_{dz}(\dot{z}_{desired} - \dot{z}) \text{ (39)}$$

$$\ddot{\Phi}_{commanded} = \ddot{\Phi}_{desired} + k_{p\Phi}(\Phi_{desired} - \Phi) + k_{d\Phi}(\dot{\Phi}_{desired} - \dot{\Phi}) \text{ (40)}$$

$$\ddot{y}_{commanded} = \ddot{y}_{desired} + k_{py}(y_{desired} - y) + k_{dy}(\dot{y}_{desired} - \dot{y}) \text{ (41)}$$

Using equation 37, the inputs can be solved for,

$$u_1 = mg + m\ddot{z}_{commanded} \text{(42)}$$

$$u_2 = I_{xx}\ddot{\Phi}_{commanded} \text{ (43)}$$

$$\Phi_{commanded} = -\frac{1}{g}\ddot{y}_{commanded} \text{(44)}$$

The PD control law can be substituted in, leaving the inputs being,

$$u_1 = mg + m[\ddot{z}_{desired} + k_{pz}(z_{desired} - z) + k_{dz}(\dot{z}_{desired} - \dot{z})] \text{ (45)}$$

$$u_2 = I_{xx}[\ddot{\Phi}_{desired} + k_{p\Phi}(\Phi_{desired} - \Phi) + k_{d\Phi}(\dot{\Phi}_{desired} - \dot{\Phi})] \text{ (46)}$$

$$\Phi_{commanded} = -\frac{1}{g}[\ddot{y}_{desired} + k_{py}(y_{desired} - y) + k_{dy}(\dot{y}_{desired} - \dot{y})] \text{ (47)}$$

Because Φ cannot be directly controlled, let $\Phi_{commanded} = \Phi_{desired}$. The first and second derivatives of Φ can be approximated as 0. This leaves the input vector

$$u = \begin{bmatrix} mg + m[\ddot{z}_{desired} + k_{pz}(z_{desired} - z) + k_{dz}(\dot{z}_{desired} - \dot{z})] \\ I_{xx}\left[k_{p\Phi}\left(\left(-\frac{1}{g}\left(\ddot{y}_{desired} + k_{py}(y_{desired} - y) + k_{dy}(\dot{y}_{desired} - \dot{y})\right)\right) - \Phi\right) + k_{d\Phi}(-\dot{\Phi})\right] \end{bmatrix} \text{ (48)}$$

This input vector can now be implemented in a non-linear system to simulate the control law. Euler's method can also be used to simulate this system.

Analyzing the System in Different Environments

The simulation created in the previous section ignores air resistance and air pressure for the sake of simplicity. In reality, these forces need to be considered while creating a control law.

To properly compare the drone's performance in Earth and Martian environments, the initial and target states will be held constant. The initial condition will be,

$$\begin{bmatrix} y(t) \\ z(t) \\ \Phi(t) \\ \dot{y}(t) \\ \dot{z}(t) \\ \dot{\Phi}(t) \end{bmatrix} = \begin{bmatrix} 5 \\ 2 \\ 4 \\ 5 \\ 1.2 \\ 6 \end{bmatrix} \quad (49)$$

and the goal state will be

$$\begin{bmatrix} y_{desired} \\ z_{desired} \\ \dot{y}_{desired} \\ \ddot{z}_{desired} \\ \ddot{y}_{desired} \\ \ddot{z}_{desired} \end{bmatrix} = \begin{bmatrix} 0 \\ 4 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad (50)$$

This goal state will make the drone move up by 2 meters and have the rest of the states return to the origin.

Using the proposed MarsBird-VII quadcopter as a baseline, the mass is 4 kg. The moment of inertia can be assumed to be $1\ kg \cdot m^2$. The simulation time will be 40 seconds. First, an Earth environment will be simulated, the gravitational acceleration $g = 9.81\ m/s^2$. With the gains, $k_{pz} = 0.4, k_{dz} = 1, k_{p\Phi} = 18, k_{d\Phi} = 15, k_{py} = 0.4$, and $k_{dy} = 1$, the path of the quadcopter is:

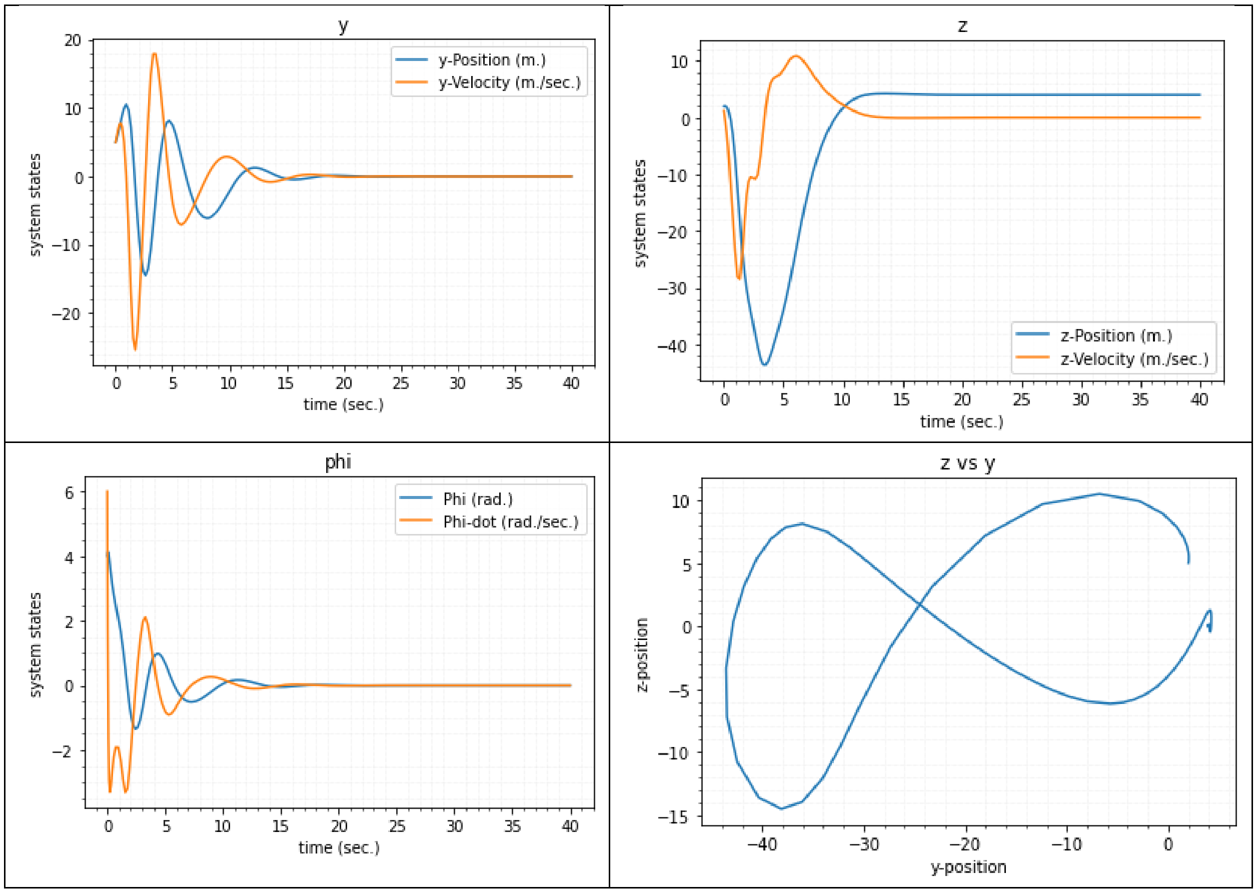


Figure 8: Quadcopter dynamics on Earth with tuned gains

With these same gains in a Martian environment: $g = 3.73\, m/s^2$, the path is:

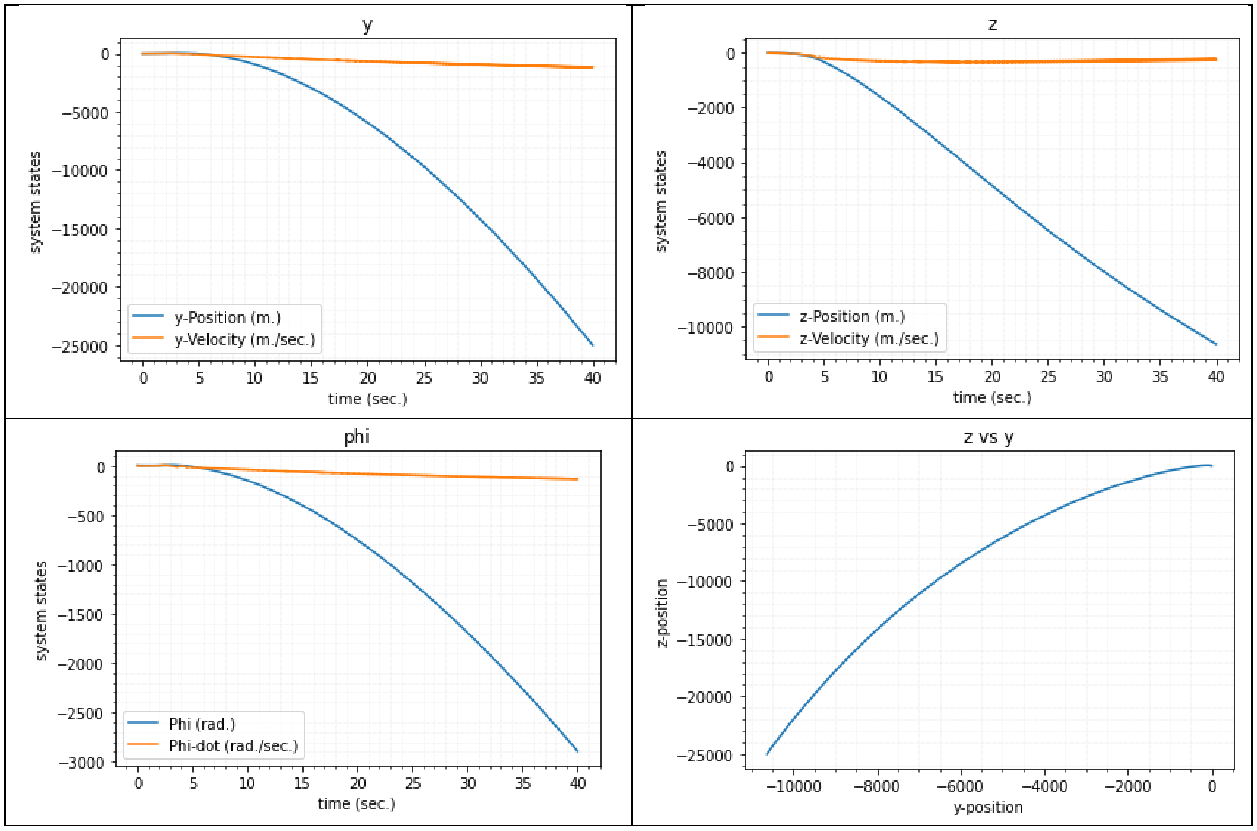
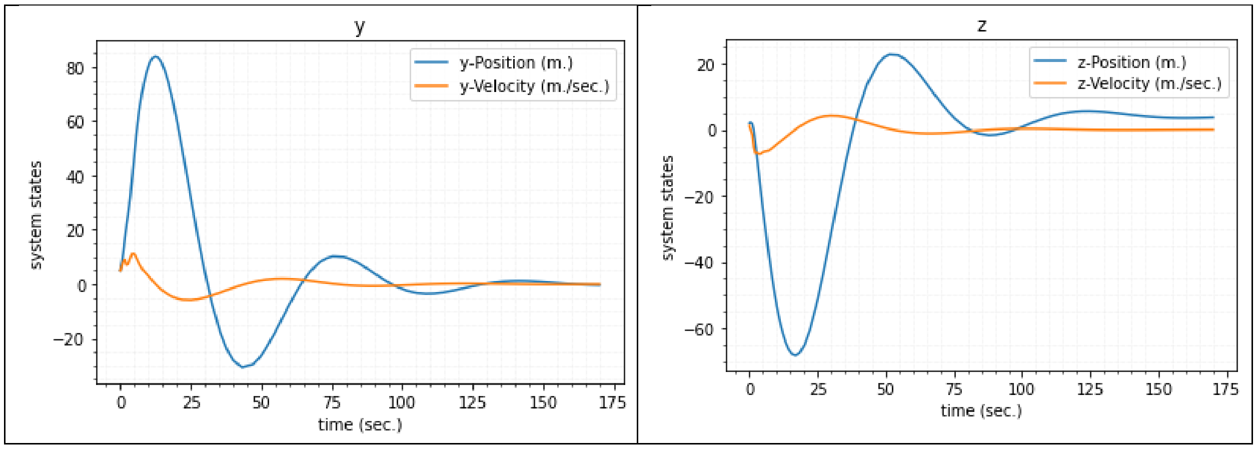


Figure 9: Quadcopter dynamics on Mars with same gains

In the Martian environment the quadcopter veers off. So obviously the gains that worked on Earth won't work on Mars. Each gain will need to be changed.

With the gains, $k_{pz} = 0.009, k_{dz} = 0.07, k_{p\Phi} = 1.5, k_{d\Phi} = 1.2, k_{py} = 0.01$, and $k_{dy} = 0.07$, and a time span of 175 seconds in the Martian environment, the path is:



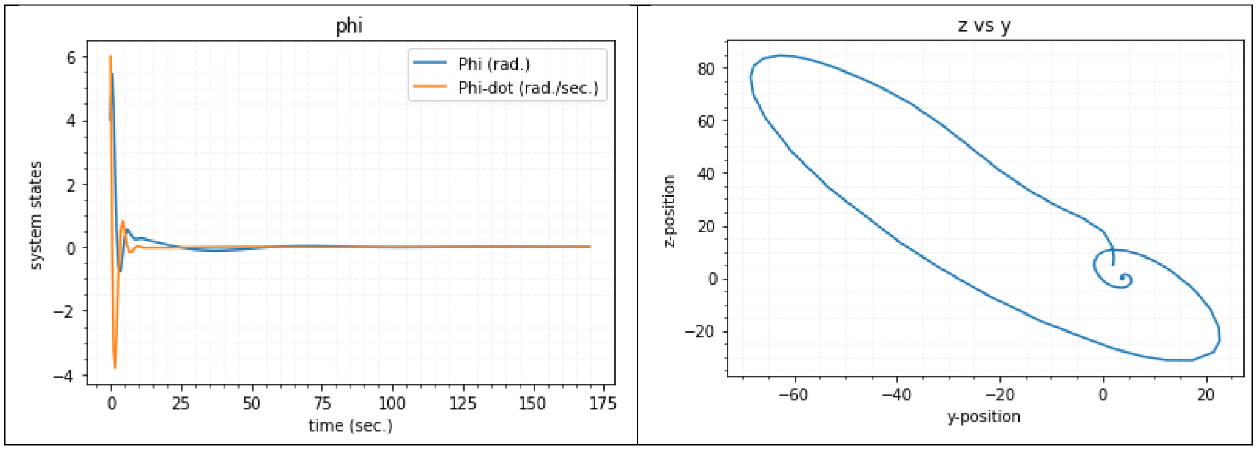


Figure 10: Quadcopter dynamics on Mars with tuned gains

d

Using these gains on Earth, the path is identical, but not as efficient as the prior gains used on Earth.

These results show the impact of gravitational acceleration on quadcopter dynamics. On Earth with the proposed gains, the quadcopter reached the desired state in about 20 seconds. However, using those same gains on Mars lead the quadcopter to veer off and not reach the target state at all. But when the gains for Martian gravity were decreased, the quadcopter reached the desired state in about 125 seconds. While this adjustment did help reach the objective, further tuning can be done to maximize efficiency.

Conclusion

This paper investigated closed-loop control strategies, specifically PD control, and how it can be effectively applied in a two-dimensional quadcopter model in both an Earth and Martian atmospheres. The field of extraterrestrial rotorcraft was explored, PID control was explained and was subsequently implemented in a spring-mass damper system, then it was used in a two-dimensional quadcopter system which was then simulated and its performance compared in both Earth and Martian environments. Different gains were needed in each environment to efficiently reach the target state. There were limitations with this investigation, for example the exclusion of air pressure and resistance, the omission of the integral term in PID control, and reliance on numerical integration rather than exact solutions. Future work could address these limitations by examining the effects of aerodynamic forces such as air resistance and pressure, investigating disturbances like wind with the control model, further tuning the gains to improve system performance, and exploring alternative closed-loop control methods. These investigations could provide deeper insights into the dynamics and control of extraterrestrial rotorcraft.

References

1. “Ingenuity Mars Helicopter - NASA Science.” NASA, science.nasa.gov/mission/mars-2020-perseverance/ingenuity-mars-helicopter/.
2. Chiou, Lyndie, “NASA’s Plans for Next-Generation Mars Helicopters Are Up in the Air”, Scientific American, May 8, 2024
3. Wall, Mike, “'It's sort of been invincible until this moment:' Mars helicopter Ingenuity pilot says 'bland' terrain may have doomed NASA chopper”, space.com, January 25, 2024
4. Wei, Erhu. Jin, Shuanggen. Zhang, Qi. Et al. “Autonomous navigation of Mars probe using X-ray pulsars: Modeling and results”, Advances in Space Research, October 10, 2012.
5. Yu, Zhengshi. Cui, Pingyuan. Crassidis, John L. “Design and Optimization of Navigation and Guidance Techniques for Mars Pinpoint Landing: Review and Prospect”, Progress in Aerospace Sciences, September, 2017
6. Lam, Johnny N. Bayard, David. Conway, Dylan T. et al. “Flight Control System For NASA’s Mars Helicopter”, JPL dartslab, 2019
7. Forino, Francesca, “SPACE DRONES: Design and Control of UAVs for planetary exploration”, Politecnico Di Torino, July, 2020
8. Michelson, Robert C. Naqvi, Messam A. “Extraterrestrial Flight (Entomopter-Based Mars Surveyor)”, angel-strike.com
9. Kallstrom, Kristen. Weist, Lauren. Schatzman, Natasha. Et al. “PlanetarY Telemetric Helicopter for Investigation and Analysis (PYTHIA): A Rotorcraft for Martian Lava Tube Exploration”, NASA Ames Research Center
10. Kaijie Zhu, Qiquan Quan, Dewei Tang, Yachao Dong, Kaiyi Wang, Bo Tang, Qi Wu, Zongquan Deng, “A Mars quadcopter capable of autonomous flight and sample collection: Structure and avionics”, Acta Astronautica
11. Bayisa, Ayele, “Controlling Quadcopter Altitude using PID-Control System”, International Journal of Engineering Research & Technology, December, 2019
12. Trentelman, H., Stoorvogel, A. A., & Hautus, M. (2002). Control theory for linear systems. University of Groningen, Johann Bernoulli Institute for Mathematics and Computer Science.
13. Friedland, Bernard, “Control System Design: A Introduction to tate-Space Models”, Dover Publications, 1986.
14. Chen, Guanrong, “Stability of Nonlinear Systems”, Department of Electronic Engineering City University of Hong Kong Kowloon, December, 2004
15. Coursera, Robotics: Aerial Robotics