USING PID CONTROLLERS TO PROPULSIVELY LAND A ROCKET IN SIMULINK

Report submitted to SASTRA Deemed to be University as the requirement for the course

MCT 207 : SYSTEM MODELLING, DYNAMICS AND CONTROL

Submitted by

NAIMISH MANI B (Reg. No.: 122009154, B.Tech Mechanical)

June 2021

SCHOOL OF MECHANICAL ENGINEERING
THANJAVUR, TAMIL NADU, INDIA - 613 401

1. Abstract

Rockets, although being very expensive to build, have become an integral (yet not always so obvious) technology to the enabling of our modern lives. From GPS navigation to real-time Satellite TV, in the absence of rockets launching satellites into space we would not have access to such luxuries. It was estimated in 1998 that successfully managing to retrieve the Core Stage of the Ariane V could save upto $8.5 Million per flight. Now-a-days, SpaceX is actively working on (and have successfully demonstrated) propulsively landing and recovering their rocket boosters. Through this project, I attempt to design a PID controller in SIMULINK to propulsively land a rocket and I also simulate the same.

Keywords: Control System, PID Controller, SIMULINK, Rockets.

Nomenclature

T	Thrust exerted by the Engine
D	Drag force acting on the Rocket
M	Mass of the Rocket
J_y	Moment of Inertia of the Rocket
R_e	Radius of the Earth
g	Acceleration due to Earth's gravity
γ	Flight Path Angle
v	Velocity of the Rocket
x	Downrange Position of the Rocket
h	Altitude of the Rocket
t	Time

2. Introduction

There are predominantly two ways to land a rocket. One is along the lines of the Space Shuttle Program by NASA (1981 - 2011), where the rocket re-enters the atmosphere from outer space and lands like a traditional aeroplane, and the other one is by deploying subsequent stages of the rocket and landing the primary stage vertically on another launchpad, like what SpaceX does. Both methods depend on fuel onboard to provide thrust, guiding the vehicle to recovery. Here, we are aiming for a vertical launch and vertical landing, and will be using a Control System designed in Simulink to achieve the same.

2.1 Control Systems

Figure 1

(https://www.tutorialspoint.com/control_systems/images/block_diagram.jpg)

In a nutshell, a Control System is a system in which we attempt to control the output of the system by varying its input. There are two main types of control systems, namely Open Loop Control Systems and Closed Loop Control Systems. Here, we make use of a Closed Loop Control System to control the angle of thrust, hence guiding the rocket to a vertical descent. This type of control system is also commonly known as a Feedback Controlled Loop,

2.2 SIMULINK

Simulink is a MATLAB-based graphical programming environment for modeling, simulating and analyzing multi-domain dynamical systems. Its primary interface is a graphical block diagramming tool and a customizable set of block libraries. In this project, Simulink was used to model the Rocket system, tune the PID Controller and also perform rough animations using the 6DoF animation block.

3. Governing Equations

In general, the governing equations of a system describe how the values of the dependent variables change when one or more of the independent variables change. Our system requires the following equations to define it. For simplicity's sake, we define the system to have 3 Degrees of Freedom (DoF), restricting motion to just the Pitch, X and Z axes. The Rocket system is designed based on publicly available data regarding the Falcon rocket from Space X.

3.1 Engine Model

Here, we take inspiration from SpaceX’s Falcon series rocket’s engine. Reading the Merlin engine’s static burn test results, we get the following details.

	Mean Sea Level	Vacuum
Thrust	346961.28 N	394557.26 N
Specific Impulse	267 s	300 s
Mass Flow Rate	132.46 kg/s	134.06 kg/s
Burn Time	162.25 s	160.30 s
Fuel	21491.26 kg	-

Table 1

(Source : http://www.b14643.de/Spacerockets_2/United_States_1/Falcon-9/Merlin/index.htm)

To make our calculations simpler, we assume constant mass flow rate throughout the system duration, and that the rocket's operating conditions are identical to that observed at Mean Sea Level.

3.2 Rocket Model

Again, we make reference to the Falcon 9 series rockets and their launches to get the details of our rocket's model. They are as follows:

Dry Mass	1360.7 kg
Fuel Mass	21491.26 kg
Payload Mass	5000 kg
Length	15 m
Diameter	1.5 m
Location of Centre of Gravity	7 m (from the base)

Table 2

(Source : https://www.spacelaunchreport.com/falcon.html)

3.3 Equations of Motion

These equations govern the motion of the rocket. To account for the curvature of the Earth, polar coordinates are used. The origin of the coordinate system is centered at the center of the Earth. The final equations of motion are as follows:

$$\frac{dv}{dt} = \frac{T - D}{M} - g * sin(\gamma)$$

(Integrate acceleration from engine minus acceleration due to gravity to get velocity)

$$\frac{d\gamma}{dt} = \left(\frac{v}{R_e + h} - \frac{g}{v}\right) * cos(\gamma)$$

(Integrate normal acceleration equation to get Flight Path Angle)

$$\frac{dx}{dt} = \left(\frac{R_e}{R_e + h}\right) * v * cos(\gamma)$$

(Integrate horizontal component of velocity to get downrange position from center of Earth)

$$\frac{dh}{dt} = v * sin(\gamma)$$

(Integrate vertical component of velocity to get altitude from the center of the Earth)

$$\frac{d^2\theta}{dt^2} = \frac{T * d_2}{J_y}$$

(Integrate Thrust into Moment arm divided by Moment of Inertia to get Pitch Angle)

3.4 Atmosphere Model

Since the rocket experiences drag forces, an atmosphere model is required to calculate the same. We make use of NASA's Earth Atmosphere Model for the same. The atmosphere model is as follows:

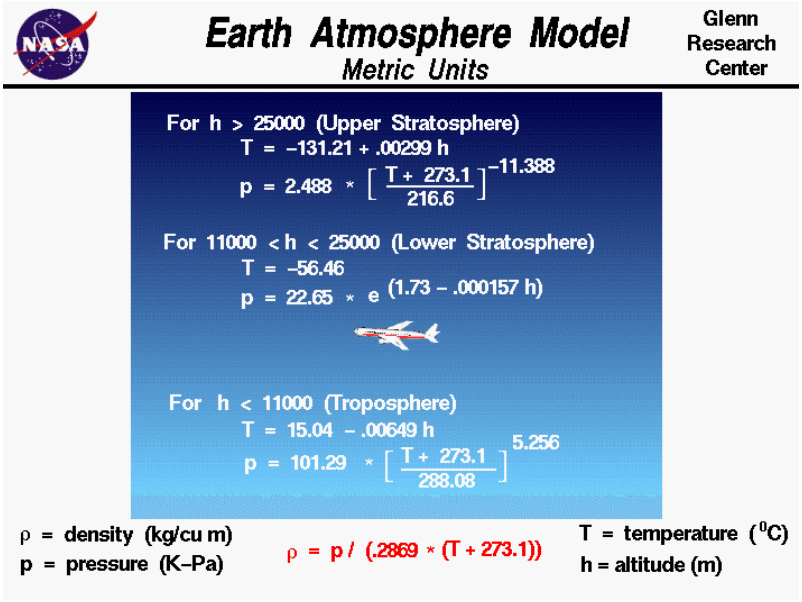


Figure 2, NASA's Earth Atmosphere Model

(https://www.grc.nasa.gov/www/k-12/airplane/atmosmet.html)

3.5 PID Controller Equations

A Proportional Integral Derivative (PID) controller is an intuitive controller that is suitable for Single Input Single Output systems.

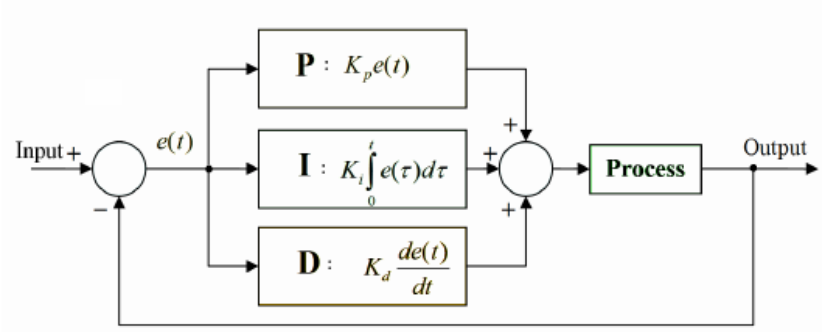


Figure 3

(https://www.researchgate.net/figure/Closed-loop-feedback-single-input-single-output-SISO-control-using-a-PID-controller-In_fig1_221919279)

From figure 3 we can see that the input - output relationship can be defined with the equation,

$$u(t) = K_p e(t) + K_i \int e(t)dt + K_d \frac{d}{dt} e(t)$$

Taking the Laplace Transform of this equation yields us the Transfer Function,

$$\frac{E(s)}{U(s)} = \frac{s}{K_d * s^2 + K_p * s + K_i}$$

Fine-tuning of *Kp*,*Ki* and *Kd* is performed in the continuous time domain, in this project. These parameters are designed based on the desired response to a step input. Well-known methods for parameter tuning are the Ziegler-Nichols rules and root locus method. For this project, I make use of Simulink's inbuilt PID controller tuning function.

4. Simulink Implementation

The Rocket's flight is broken up into three segments. They are "Powered Ascent", "Pitch Over" and "Powered Descent" respectively. In the "Powered Ascent" phase, the Rocket is launched from the launch pad. In the "Pitch Over" phase, the Rocket flips itself over, to prepare itself for landing. Finally, in the "Powered Descent" phase, the Rocket tries to land itself vertically. These phases are inspired from the Falcon 9's phases as well.

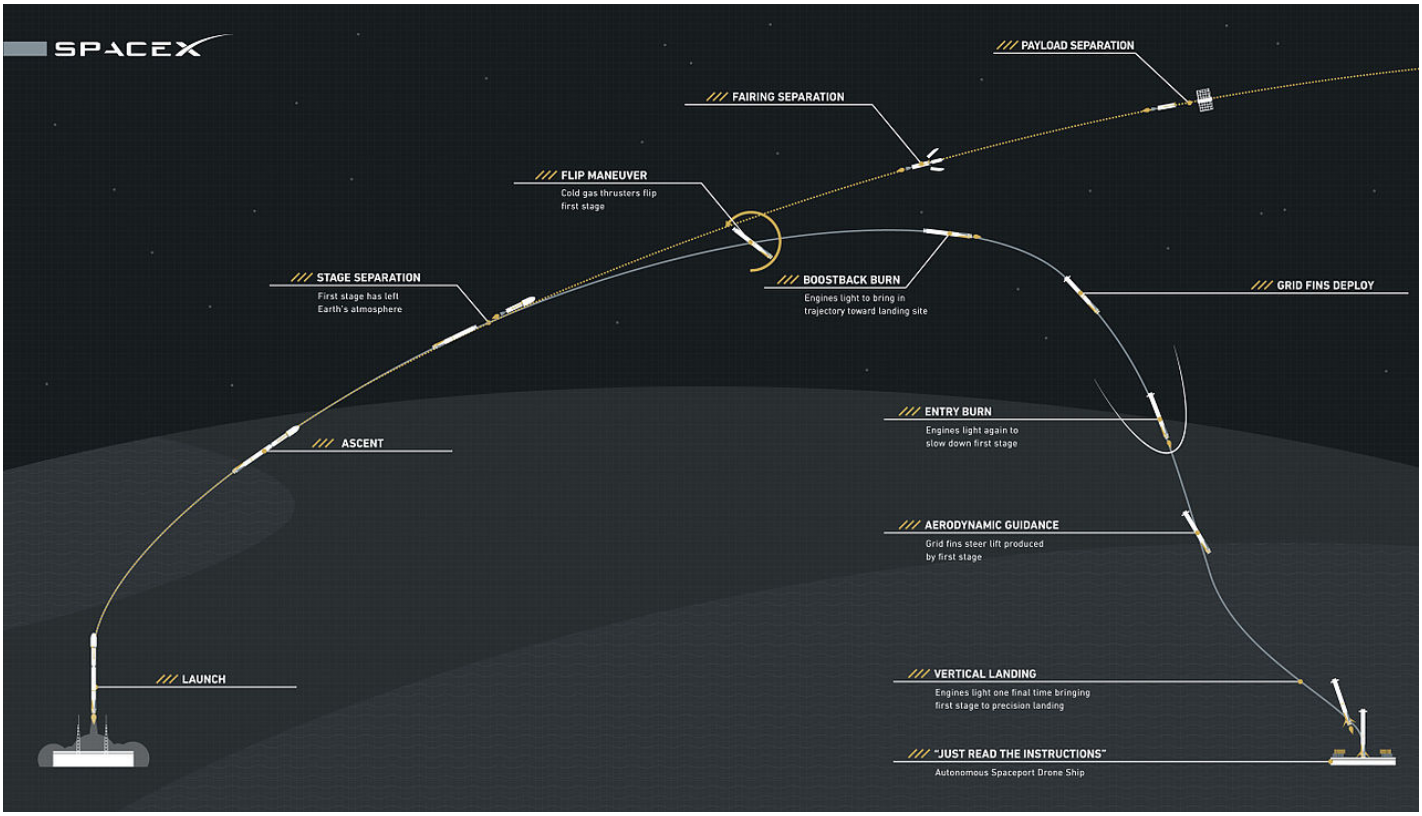


Figure 4, Adapted from (https://upload.wikimedia.org/wikipedia/commons/thumb/c/c1/Falcon_9_First_Stage_Reusability_Graphic.jpg/1280px-Falcon_9_First_Stage_Reusability_Graphic.jpg)

4.1 Initial State of the System

Most simulations require a set of initial conditions to simulate from, and this system is no different. Here, our initial conditions are defined as follows:

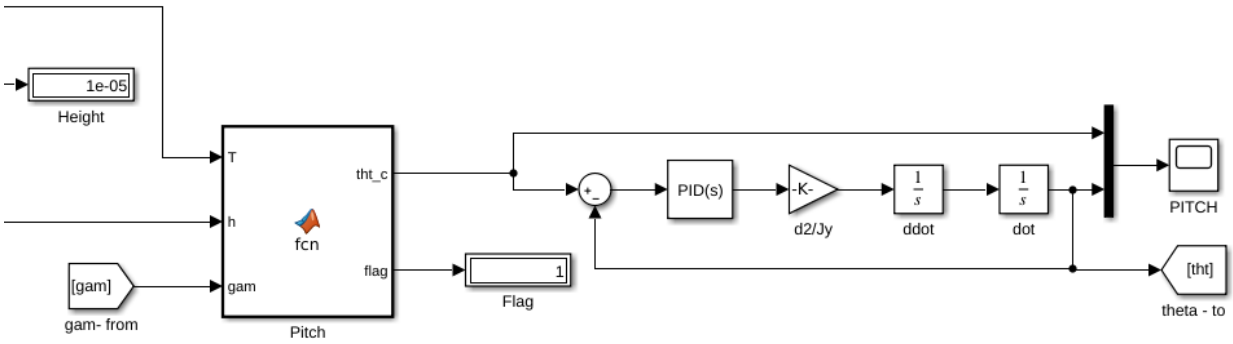
v_0	Initial velocity	0.001 m/s
x_0	Initial downrange position	0.001 m
h_0	Initial altitude	0.001 m
gam_0	Initial gamma	pi/2 rad
gam_in	Gamma step value	0.1
t_turn	Time required for Pitch Over	30 s

Note : Initial conditions are not exactly 0, but rather close to 0 to prevent numeric integrator errors

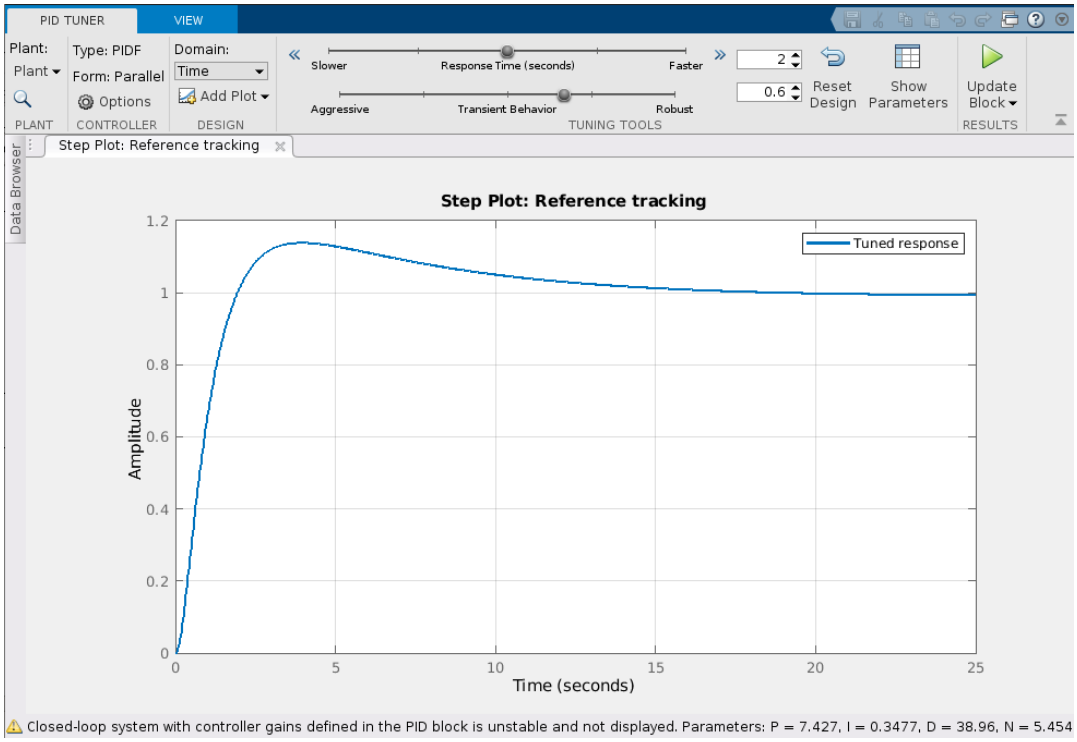
4.2 PID Controller Tuning

I performed PID Controller tuning by using the Simulink in-built tool. The results of the tuning are as follows.

a) PID Controller Blocks

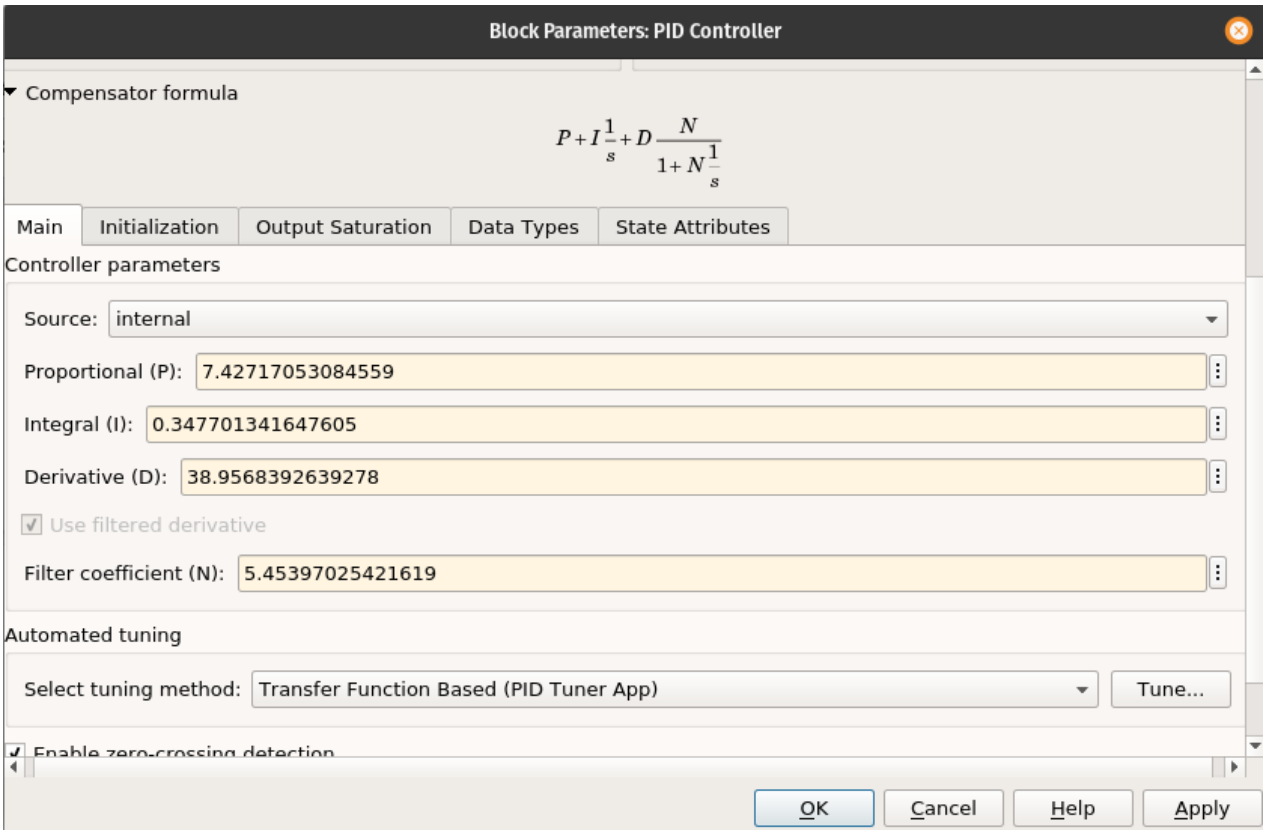


b) Tuning the Controller



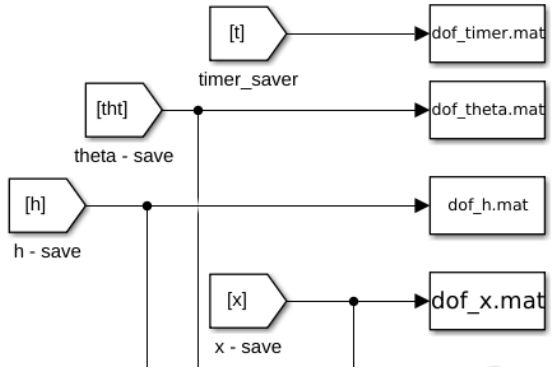
As we know, rocket dynamics is non-linear in nature. Hence, the system is flagged by Matlab as being unstable in nature here.

c) Final Values



4.3 Logging System State

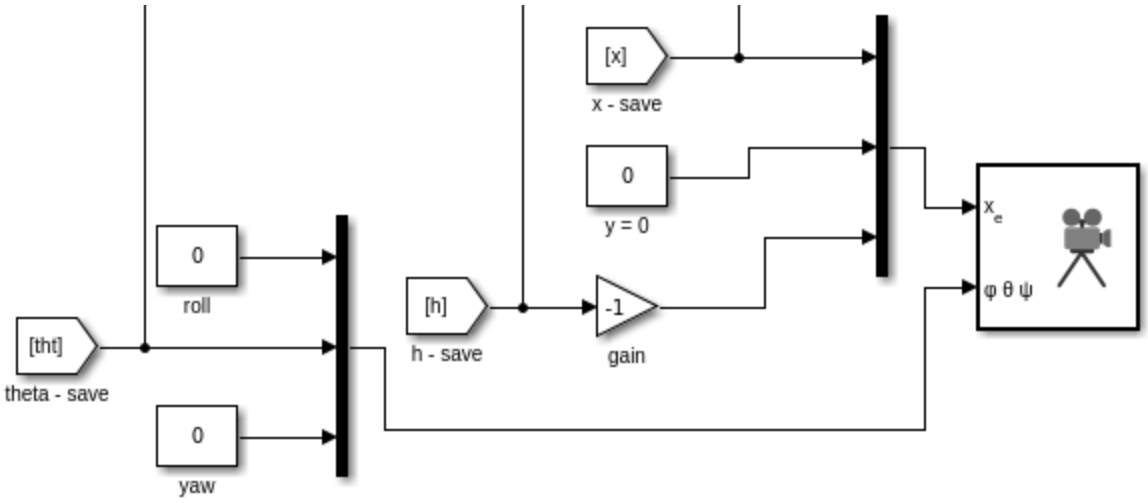
I logged the system's state at each timestep to produce custom visualisations. This was implemented as shown in the image below.



4.4 Visualisation with the 6DoF Animation Block

The system can be easily visualised by using the 6DoF animation block provided out of the box in Simulink. Although our system has only 3DoF, we can trick Simulink into thinking it is a 6DoF system by setting yaw, roll and y to equal 0. The block was rigged up as shown in the images below.

a) Rigging up the 6DoF block

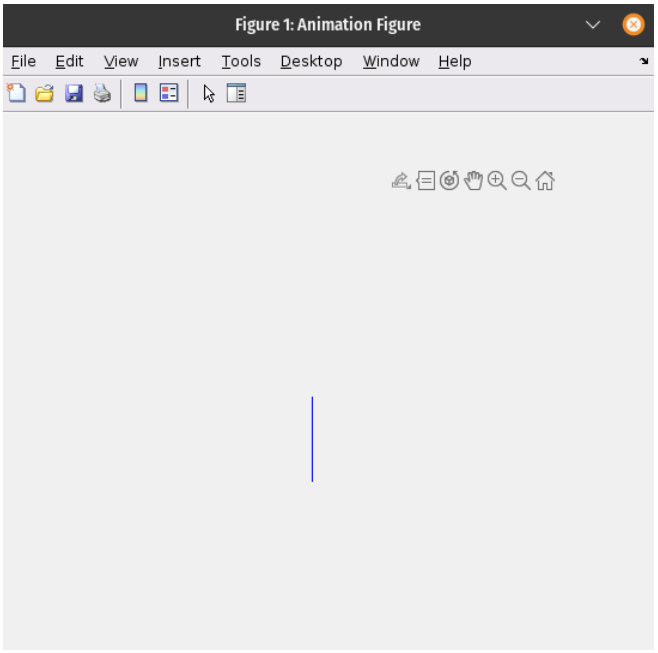


The images on the next page indicate the following stages of flight.

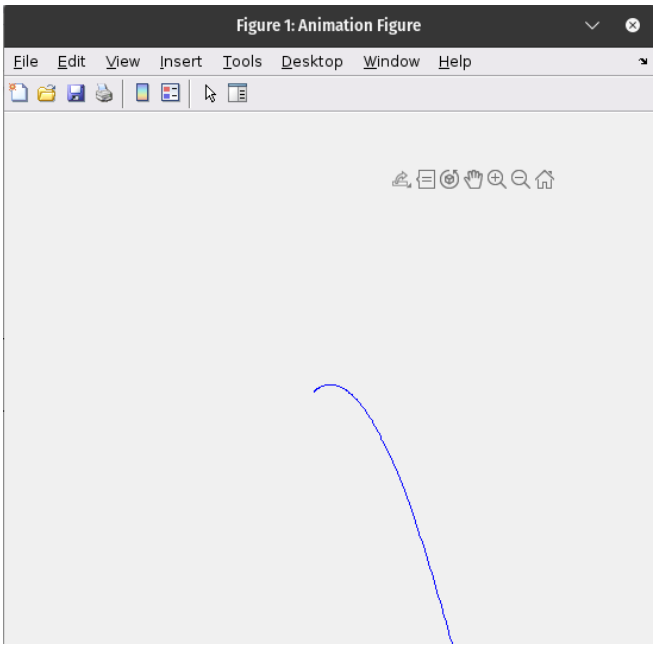
(1) Take off

(2) Pitch Over Maneuver

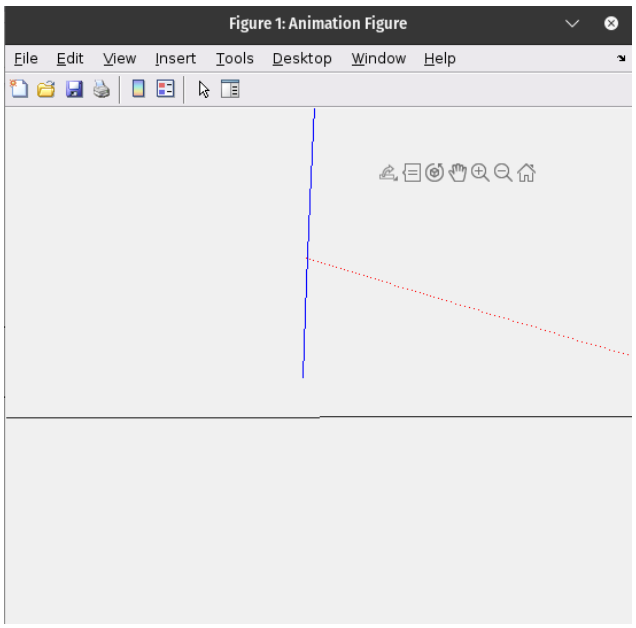
(3) Landing



(1)



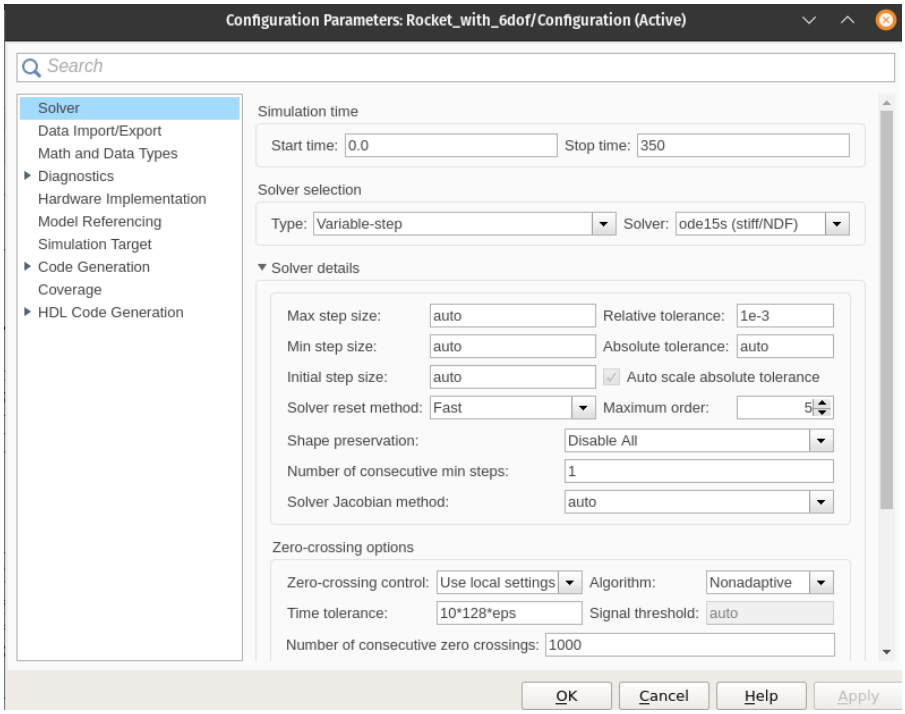
(2)



(3)

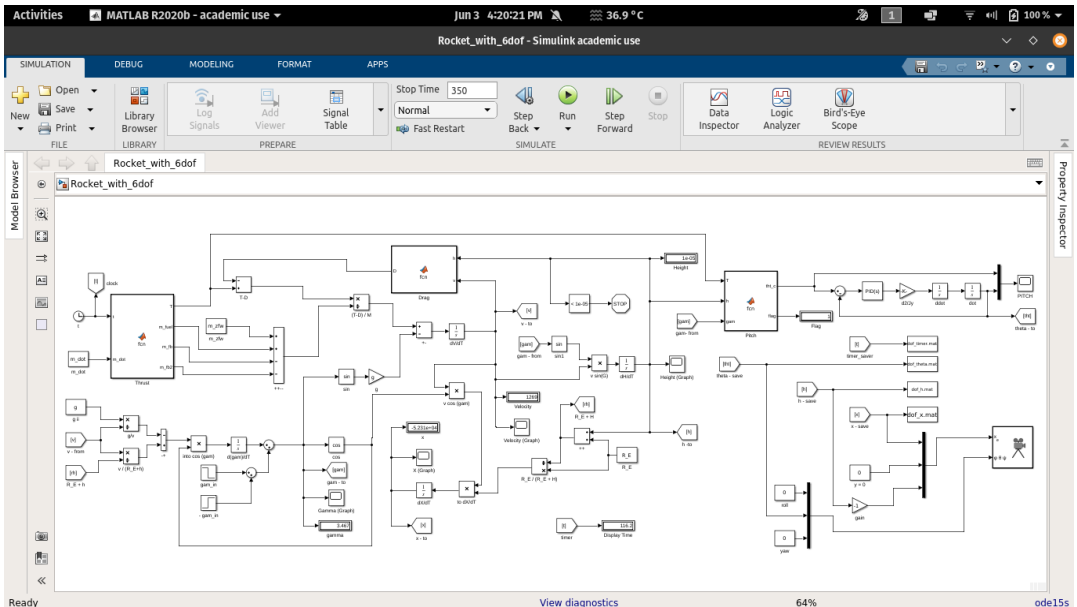
4.5 Integrator Settings

As we are simulating the system through means of a numeric integrator, it is crucial that we select the appropriate one. Since the system is very unstable, different integrators can produce very different results. The integrator settings can be seen in the image attached below.



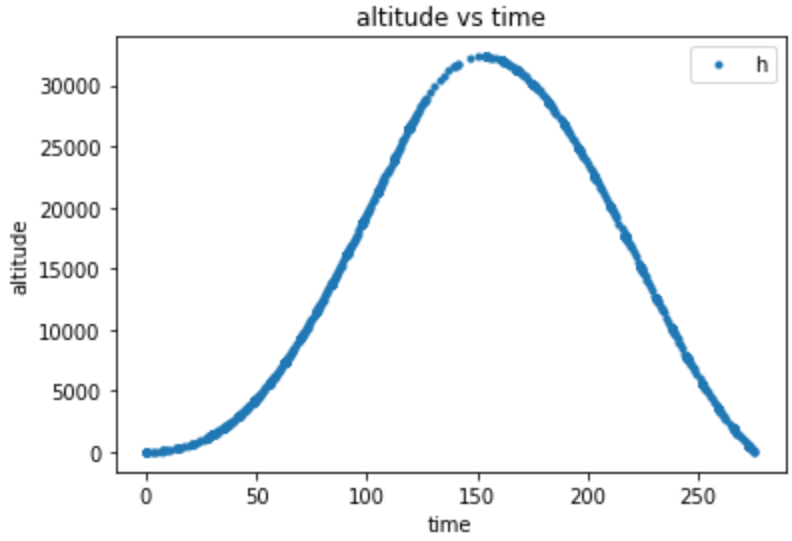
4.6 Final Simulink Model

After implementing the model in Simulink, this is what the system looks like.



5. Interpreting Simulation Logs in Python

After running the simulation, we load the log files into Python for further visualisations. The results of this can be seen below.

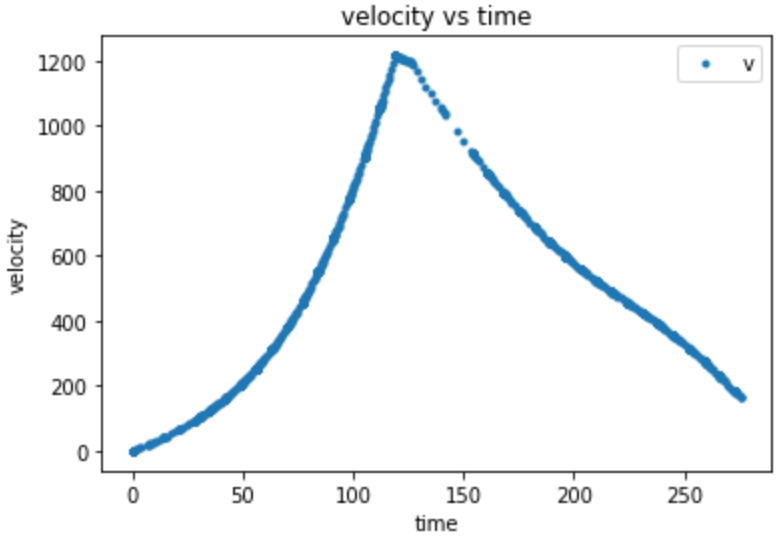


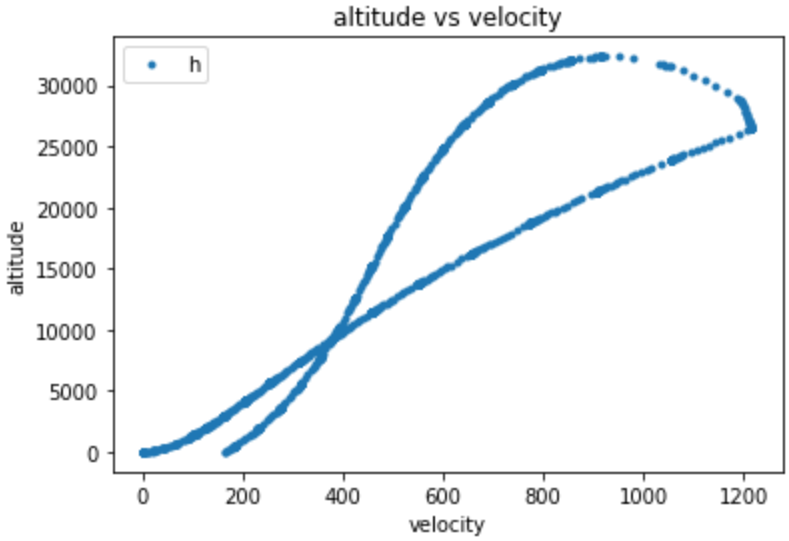
5.1 Altitude vs Time

From this graph, we can infer that the rocket ascended and descended smoothly across the time of flight.

5.2 Velocity vs Time

From this graph, we can see that the velocity of the rocket was decreasing towards the end of the simulation. So, by adjusting the landing fuel reserves, we might be able to bring the touchdown velocity even further ideally to the single digits.



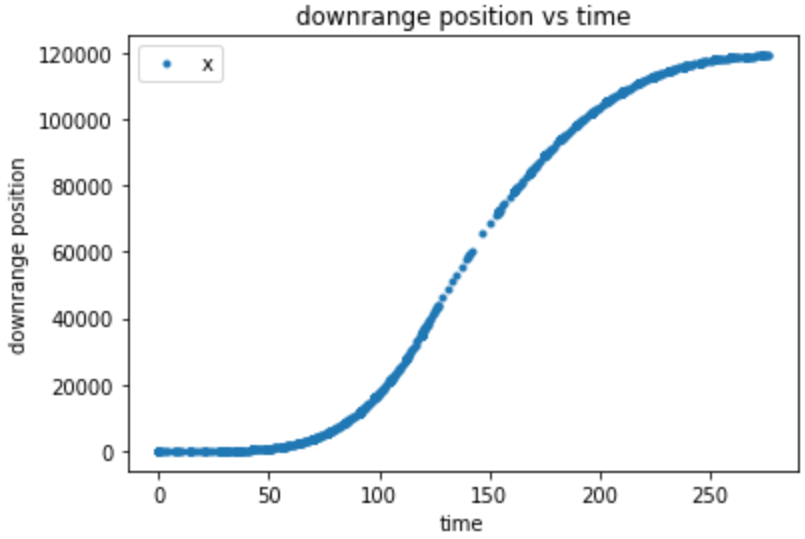


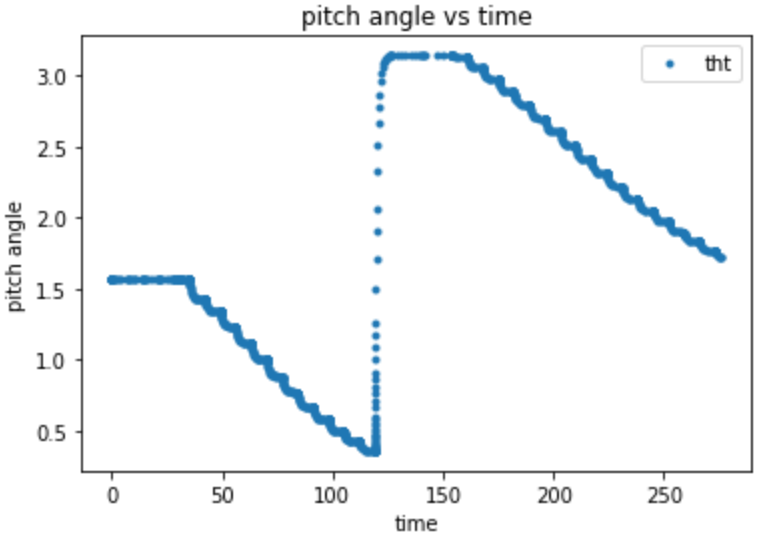
5.3 Velocity vs Altitude

From this graph, we can see that the velocity of the rocket was decreasing as it started approaching the ground, implying the PID controller was working nominaly.

5.4 Downrange Position vs Time

Here, we can see that the rocket was going downrange across the entire duration of the flight, and briefly (towards the middle of the flight) the rocket had a strong x-component of velocity.

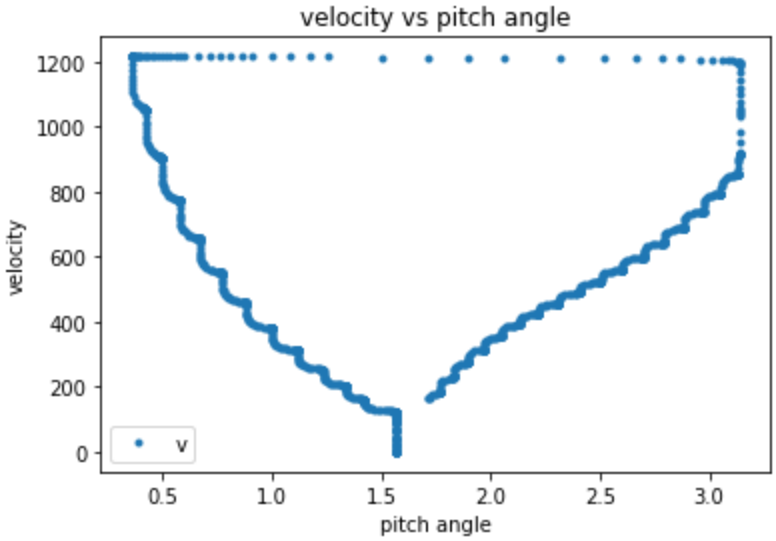




5.5 Pitch Angle Vs Time

From this graph, we can see that the pitch-over maneuver worked successfully, as is indicated by the pitch angle jumps from 0

to pi radians. We can also see that the PID controller was working nominally.



5.6 Velocity vs Pitch Angle

The left half of the plot is from the powered ascent phase, and the right half from the powered descent phase. From this plot, we can clearly see the pitch over maneuver the rocket underwent, and also see how the PID controller was working to get the rocket as stable vertically as possible.

5.7 Final Flight Conditions

```
[ ]  # Final Conditions
     df.iloc[-1, :]
```

```
t         275.713582
x      119190.740078
h           0.000010
v         162.685531
tht         1.716363
Name: 1142, dtype: float64
```

From this, we can see that:

- The rocket reached the ground successfully.
- The rocket landed with a pitch angle of about 98 degrees.
- The rocket landed with a velocity of 162.69 m/s.

From this, we can make the following inferences:

1. The PID controller works, as indicated by the pitch change.
2. The rocket is falling at a high velocity, indicating we should increase the fuel reserved for landing.

5.8 General Flight Overview

```
[ ] df.describe()
```

	t	x	h	v	tht
count	1143.000000	1143.000000	1143.000000	1143.000000	1143.000000
mean	129.225792	48489.548080	14266.084848	513.725390	1.630348
std	77.525223	48172.128544	10110.606428	318.726247	0.848382
min	0.000000	0.000956	0.000010	0.001000	0.356004
25%	63.086178	2222.140578	4651.201270	253.917541	0.881126
50%	111.174474	26757.517854	13380.862467	458.168605	1.570796
75%	198.689546	103224.729204	23748.858611	736.085821	2.313150
max	275.713582	119190.740078	32353.300626	1216.174272	3.141593

From the above table, we can make the following inferences:

1. The maximum altitude attained by the rocket was 32 km.
2. The maximum velocity of the rocket was 1216.17 m/s.
3. The maximum pitch angle of the rocket was pi radians (i.e., the rocket briefly was travelling horizontally)
4. The rocket travelled 119.19 km downrange from the launchpad.
5. The rocket was in motion for 275.71 seconds.

6. Conclusion

From the above graphs and simulations, we can make the following conclusions:

1. The PID controller works as intended, but can still be improved upon.
2. The pitch over maneuver worked as intended.
3. To recover the Rocket, we will be required to travel 120 km downrange.
4. The split up of fuel between the two stages can be modified to improve performance.

7. Future Work

Some potential avenues for future work on this project include

- Using Reinforcement Learning to identify better values for the PID controller.
- Using a more robust atmospheric model to calculate Drag.
- I assumed a constant mass flow rate and 100% engine efficiency for the rocket for this simulation. In the future, we could look into more realistic models for the same.
- Using FlightGear for visualisations instead of Simulink 6DoF animator.

8. References

Buursink, J. Launch cost reduction by the reuse of the core stage of Ariane 5. Journal of Reducing Space Mission Cost 1, 209–223 (1998)

Glen J. Culler and Burton D. Fried, Universal Gravity Turn Trajectories, Journal of Applied Physics 28, 672 (1957)

Evolution of SpaceX Merlin Engines

http://www.b14643.de/Spacerockets_2/United_States_1/Falcon-9/Merlin/index.htm

Date last accessed : 12th June 2021

Space Mission Launch Reports

https://www.spacelaunchreport.com/falcon9ft.html

Date last accessed : 12th June 2021

Standard Atmosphere Model

https://www.grc.nasa.gov/www/k-12/airplane/atmosmet.html

Date last accessed : 12th June 2021

PID Controller Tuning in Simulink

https://ctms.engin.umich.edu/CTMS/index.php?example=Introduction&section=ControlPID

Date last accessed : 12th June 2021

Falcon 9 Mission Reports and Data Sheet

https://sma.nasa.gov/LaunchVehicle/assets/space-launch-report-falcon-9-data-sheet.pdf

Date last accessed : 12th June 2021

Rocket Equations of Motion

https://fenix.tecnico.ulisboa.pt/downloadFile/281870113704944/Tese.pdf

Date last accessed : 12th June 2021

Vehicle Pitch Dynamics, Static and Dynamic Stability

http://mae-nas.eng.usu.edu/MAE_5900_Web/5900/USLI_2010/Flight_Mechanics/Section5.2.pdf

Date last accessed : 12th June 2021

9. Code Files

GitHub Repository

https://github.com/Naimish240/rocket-lander

Google CoLab Notebook

https://colab.research.google.com/drive/1DLmdj9AxHuTVn3pF3y2NuVd2Z18hUjxH?usp=sharing