Simulating Rocket Launch Trajectories with Various Thrust Profiles

Dev Saxena

November 9, 2024

1 Project Background

The aim of this project is to develop an automated pipeline that could be used efficiently in real-time systems to simulate rocket launch trajectories. This is no trivial task as the project involves complex numerical calculations when dealing with varying drag and thrust forces as they change over time. Many applications in the real world such as GNC(guidance, navigation, control) systems, risk analysis, and system diagnostics require the essential components of speed and automation in any simulation framework.

2 Simulation Description

From a high level point of view, this simulation algorithm shows how factors such as thrust, drag, and gravity affect the altitude and the velocity over time of the rocket. Each stage of the rocket's ascent can be computed in discrete measurements. The algorithm that makes this computation requires optimizing the time steps and other parameters we choose for accuracy and stability [1]. We lay out the algorithm by first defining the forces that act on the rocket, then defining the differential equations that allow us to observe change over time, and finally defining the numerical method we use to solve those differential equations.

2.1 Forces Acting on the Rocket

There are three main forces that act on the rocket in our simulation and they are defined in the following way:

$$\text{Thrust } = F_T = c \tag{1}$$

describes the upward force generated by the rocket in Newtons where c is a positive real number.

$$\text{Gravitational } = F_G = m \cdot g \tag{2}$$

describes the gravitational force (Newtons) acting on the rocket where m is the mass of the rocket in kg and g is the downward acceleration due to gravity in $\frac{m}{s^2}$.

$$\text{Drag } = F_D = \frac{1}{2}\rho v^2 C_D A \tag{3}$$

describes the downward drag force (Newtons) acting on the rocket [2] where

$$\begin{aligned} \rho &: \text{ air density in } \frac{kg}{m^3} \\ v &: \text{ current velocity of the rocket in } \frac{m}{s} \\ C_D &: \text{drag coefficient} \\ A &: \text{ rocket's cross-sectional area in } m^2 \end{aligned}$$

The thrust and gravitational force are pretty straightforward in their definitions. The drag force however, depends heavily on C_D and A. C_D is difficult to determine as there can be multiple sources of drag forces and testing of the rocket body in a wind tunnel is usually required to accurately determine drag forces. As we do not have access to such resources, we test our rockets with varying values of C_D that rely solely on how streamlined the body of our rocket may be. The drag depends on the size of the body which is defined by a reference area, A, which in this case will be the frontal area of the rocket orthogonal to the direction of the drag flow [2].

With all our forces defined, we first determine the net force acting on our rocket using equations (1), (2), and (3). Then, we use Newton's 2nd law of motion to define the acceleration of our rocket [3].

$$F_{\text{net}} = F_T - F_G - F_D$$

$$a = \frac{F_{\text{net}}}{m} \tag{4}$$

2.2 Defining the Differential Equations

The equations we have so far describing the rocket's acceleration are for a single point in time. However, the aim of this project is to obtain the altitude and velocity of the rocket as it moves through time given some initial values. For this, we use a system of ordinary differential equations [5] derived from equation (4) that will define the velocity and altitude of the rocket as it changes over time.

Let

$$\begin{aligned} h &: \text{ altitude or position of the rocket in meters} \\ v &: \text{ velocity of the rocket in } \frac{m}{s} \end{aligned}$$

Then, since the derivative of velocity with respect to time is acceleration,

$$\begin{aligned} \frac{dv}{dt} &= a \\ \frac{dv}{dt} &= \frac{F_{\text{net}}}{m} \\ \frac{dv}{dt} &= \frac{F_T - (mg) - (\frac{1}{2}\rho v^2 C_D A)}{m} \end{aligned}$$

And, since the derivative of position with respect to time is velocity,

$$\frac{dh}{dt} = v$$

This results in our system of ODEs:

$$\frac{dv}{dt} = \frac{F_T - (mg) - (\frac{1}{2}\rho v^2 C_D A)}{m} \tag{5}$$

$$\frac{dh}{dt} = v \tag{6}$$

2.3 Numerical Integration by RK4 Method

The system of ODEs given by (5) and (6) can't be solved analytically due to the presence of the drag equation (3) when $C_D > 0$. Therefore, we must rely on numerical integration techniques that move through time in discrete steps and integrate the system at each step. Although there are many different numerical methods that can solve this system from the Euler method to the trapezoidal method, we choose the fourth-order Runge Kutta (RK4) method. The RK4 method gives us convergence properties superior to methods of the first and second order [4]. The approximation error from the RK4 method also drops much faster than other methods.

When applied to our system of ODEs given by (5) and (6), RK4 approximates the change in velocity and altitude requiring only an initial condition to start. In a certain time step, dt, RK4 uses four slopes at sub-time steps within

dt to estimate how the rocket's velocity and altitude will change.

Let an arbitrary dt (time step) be given by the interval $[t_i, t_i + dt]$. Then, the first slope which calculates the change in altitude and velocity at the left end of the interval, t_i, using the initial values of altitude (h_{t_i}) and velocity (v_{t_i}) is given by

$$s_1^h = v_i \tag{7}$$

$$s_1^v = \frac{F_T - (mg) - (\frac{1}{2}\rho v_i^2 C_D A)}{m} \tag{8}$$

The second slope, which calculates change in h and v at the midpoint of the interval ($t_i + \frac{dt}{2}$) using the slope provided by s_1, is given by

$$s_2^h = h_{t_i} + \frac{dt}{2} \cdot s_1^h \tag{9}$$

$$s_2^v = v_{t_i} + \frac{dt}{2} \cdot s_1^v \tag{10}$$

The third slope also calculates the midpoint change in values of h and v but uses the values obtained in s_2 giving a better estimate at this point in the time interval.

$$s_3^h = h_{t_i} + \frac{dt}{2} \cdot s_2^h \tag{11}$$

$$s_3^v = v_{t_i} + \frac{dt}{2} \cdot s_3^v \tag{12}$$

Finally, the fourth slope calculates the change in values of h and v at the endpoint of the interval using the previous slope s_3

$$s_4^h = h_{t_i} + dt \cdot s_3^h \tag{13}$$

$$s_4^v = v_{t_i} + dt \cdot s_3^v \tag{14}$$

We can then write the final change in altitude and in velocity for the time interval $[t_i, t_i + dt]$ as

$$h_{t_i+1} = h_{t_i} + \frac{dt}{6}(s_1^h + 2s_2^h + 2s_3^h + s_4^h) \tag{15}$$

$$v_{t_i+1} = v_{t_i} + \frac{dt}{6}(s_1^v + 2s_2^v + 2s_3^v + s_4^v) \tag{16}$$

where h_{t_i+1} and v_{t_i+1} describe the altitude and velocity at the next time step. This numerical scheme is calculated over multiple time intervals to obtain the overall changes in altitude and velocity of the rocket.

3 Progress and Next Steps

Currently in this project, I have implemented a basic Python implementation of the simulation algorithm to obtain the rocket positions and velocities at different time steps. By breaking down the trajectory calculation into several discrete time steps, the RK4 method can be used to calculate the full trajectory of the rocket.

For my next steps, I will aim to refine the simulation to increase the accuracy and minimize errors coming from the numerical calculations. Additionally, I plan to complete the rest of the workflow to include automatic parameter adjustment based on input data, incorporating different thrust profiles, and graphically displaying the predicted trajectories of the rocket. If time permits, I would also like to explore using parallel processing libraries to make my computations faster, leading to a quicker real-time simulation system.

References

[1] Andøya Space Center. *Simulating a Rocket Launch*. `https://learn.andoyaspace.no/ebook/student-rocket-pre-study/rocket-dynamics-2/simulating-a-rocket-launch/`. 2024.

[2] NASA Glenn Research Center. *Drag Equation*. `https://www1.grc.nasa.gov/beginners-guide-to-aeronautics/drag-equation-2/`. 2024.

[3] The Physics Classroom. *Newton's Second Law*. `https://www.physicsclassroom.com/class/newtlaws/lesson-3/newton-s-second-law`. 2024.

[4] Timothy Sauer. *Numerical Analysis*. 3rd. Pearson, 2017, pp. 328–331.

[5] Massachusetts Institute of Technology. *Unified Engineering: Fluid Mechanics, Structural Mechanics, Materials and Dynamics*. `https://ocw.mit.edu`. `https://ocw.mit.edu/courses/16-01-unified-engineering-i-ii-iii-iv-fall-2005-spring-2006/af59df7c34ec80675d75c0bc2ac61e02_spl2.pdf`. 2005.