FLY ME TO THE MOON!

MODELING "LUNAR LANDER" WITH OPTIMAL CONTROL

PATRICK BEAL, TIARA EDDINGTON, TJ HART, ADAM WARD, MADELYN VINES

OVERVIEW

For our project we modeled the classic Atari game entitled "Lunar Lander" (see Figure 1). The task of this game is to guide a lunar lander from a given starting position and horizontal velocity in the atmosphere to the surface of the moon by controlling the rotation of the lander and the amount of thrust applied. The goal is to land slowly, right-side up, and to minimize total fuel usage. If the lander is moving too quickly in either the horizontal or vertical direction when it touches the ground, the lander will crash and break, and the game is lost. When the lander runs out of fuel, you can no longer control the thrust and will therefore crash. The problem that we seek to solve is minimizing both the fuel expended and the time taken to land while directing the lunar lander gently to the surface of the moon.

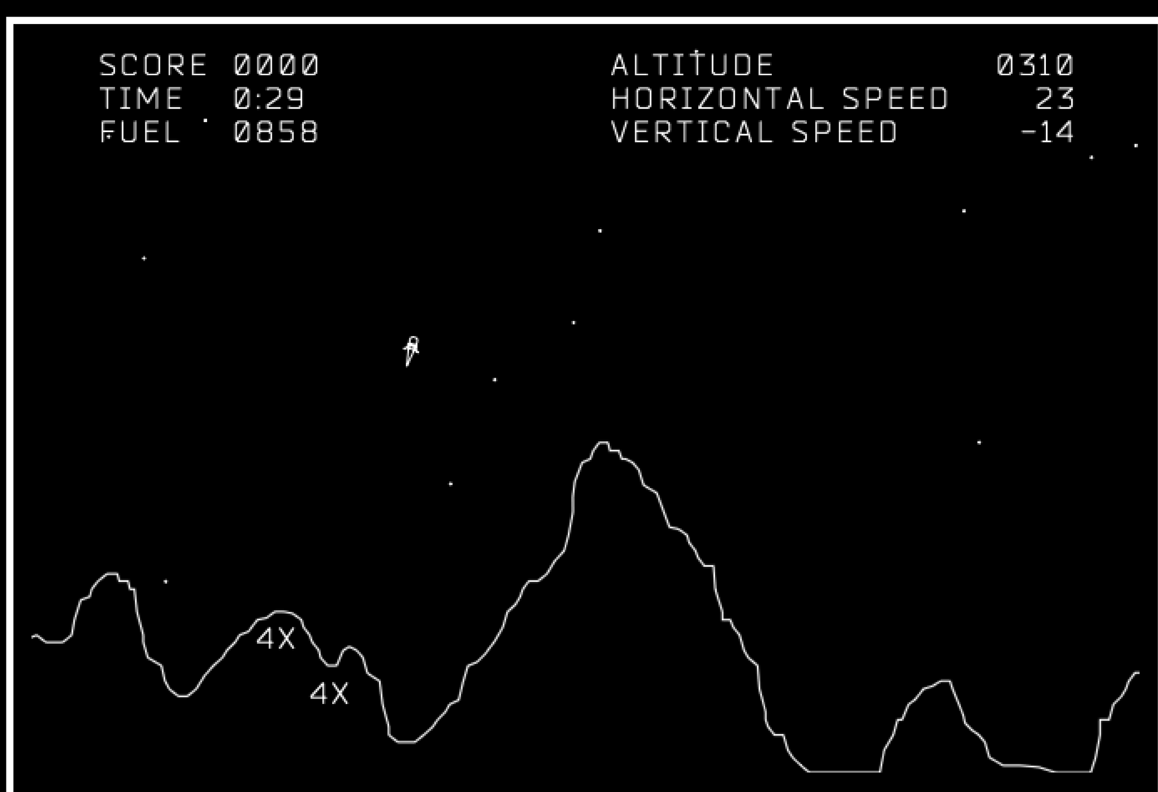

Figure 1: Lunar Lander Gameplay

THE PROBLEM

Assumptions

The game is two-dimensional, so only movement in the x (horizontal) and y (vertical) directions is allowed. During the game, the lander is controlled by rotating the lander and then either engaging the thrust or not. To make the numerical solution more tractable, we make the simplifying assumption that the lander applies thrust directly horizontally and vertically through the controls u_x and u_y. We can then reconstruct the rotation angle and magnitude of the acceleration after the solution is computed, as shown in the third image below. Moreover, we let $x_1 = x$, $x_2 = \dot{x}$, $y_1 = y$, and $y_2 = \dot{y}$ so that we have a first-order system of ODEs.

We minimize the fuel expended by seeking to minimize the magnitude of the control, $\|\mathbf{u}\|_2^2 = u_x^2 + u_y^2$. This is because the amount of fuel used is proportional to the magnitude of the thrust. To minimize the time taken to land, we let the final time t_f be free and include it in our cost functional. Next, to ensure that the landing is smooth, we minimize the magnitude of the final velocity using the penalty function $\phi(x_1(t_f), x_2(t_f), y_1(t_f), y_2(t_f)) = x_2(t_f)^2 + y_2(t_f)^2$. Also, in order to enforce the constraint that the rocket does not hit the ground before the final time t_f, we subtract $\min(0, y(t))$ from $\|\mathbf{u}\|_2^2$ inside the integral. Thus if $y(t)$ becomes negative for any $t < t_f$, an extra cost will be imposed.

Equations

Putting this all together, we obtain the following cost functional:

$$J[u] = \int_0^{t_f} \left[\alpha\|\mathbf{u}\|_2^2 - \nu \min(0, y_1(t))\right] dt + \gamma\, t_f + \beta\phi(t_f).$$

where α, β, γ, and ν are all positive constants corresponding to the cost of acceleration, final velocity, final time, and the lander hitting the ground, respectively. Using g as a gravitational constant, the state equations corresponding to this system are

$$\begin{aligned} \dot{x}_1 &= x_2, & x_1(0) &= x_0 \\ \dot{x}_2 &= u_x, & x_2(0) &= v_0 \\ \dot{y}_1 &= y_2, & y_1(0) &= y_0 \\ \dot{y}_2 &= u_y - g, & y_2(0) &= 0, \end{aligned}$$

The boundary conditions come from the fact that the initial height, horizontal position, and horizontal velocity are fixed, while the initial vertical velocity should be zero. We also enforce the boundary condition $y_1(t_f) = 0$, which assumes the lander will land on flat ground. The Hamiltonian for the system is

$$H = p_1 x_2 + p_2 u_x + p_3 y_2 + p_4(u_y - g) - \alpha\|\mathbf{u}\|_2^2 + \nu \min(0, y_1(t)).$$

Next, we apply Pontryagin's Maximum Principle to obtain the necessary conditions for the optimal control.
The condition that $\dot{\mathbf{p}} = -\frac{DH}{D\mathbf{x}}$ gives the costate equations

$$\begin{aligned} \dot{p}_1 &= 0 & \dot{p}_2 &= -p_1 \\ \dot{p}_3 &= \nu h(-y_1) & \dot{p}_4 &= -p_3 \end{aligned}$$

where h is the Heaviside function. The condition $\frac{DH}{D\mathbf{u}} = 0$ gives

$$p_2 - 2\alpha u_x = 0 \qquad p_4 - 2\alpha u_y = 0.$$

Next, because x_1, x_2, and y_2 are all free at t_f, the conditions that $\mathbf{p}(0) = \frac{D\phi}{D\mathbf{x}}$ and $\mathbf{p}(t_f) = -\frac{D\phi}{D\mathbf{x}(t_f)}$ give the equations

$$\begin{aligned} p_1(t_f) &= 0 & p_2(t_f) &= -2\beta x_2(t_f) \\ p_3(0), p_3(t_f) &= \text{free} & p_4(t_f) &= -2\beta y_2(t_f). \end{aligned}$$

Note that we do not have any endpoint conditions on p_3 because y_1 is fixed at both endpoints. Finally, because we are optimizing over the final time t_f as well, we have the condition $H(\tilde{t}_f) = \frac{\partial\phi}{\partial t}(\tilde{t}_f)$, giving

$$H(\tilde{t}_f) = \gamma.$$

Using these conditions, we formulate a numerical model to estimate solutions to the optimal control using `solve_bvp`.

EARLY RESULTS & ANALYSIS

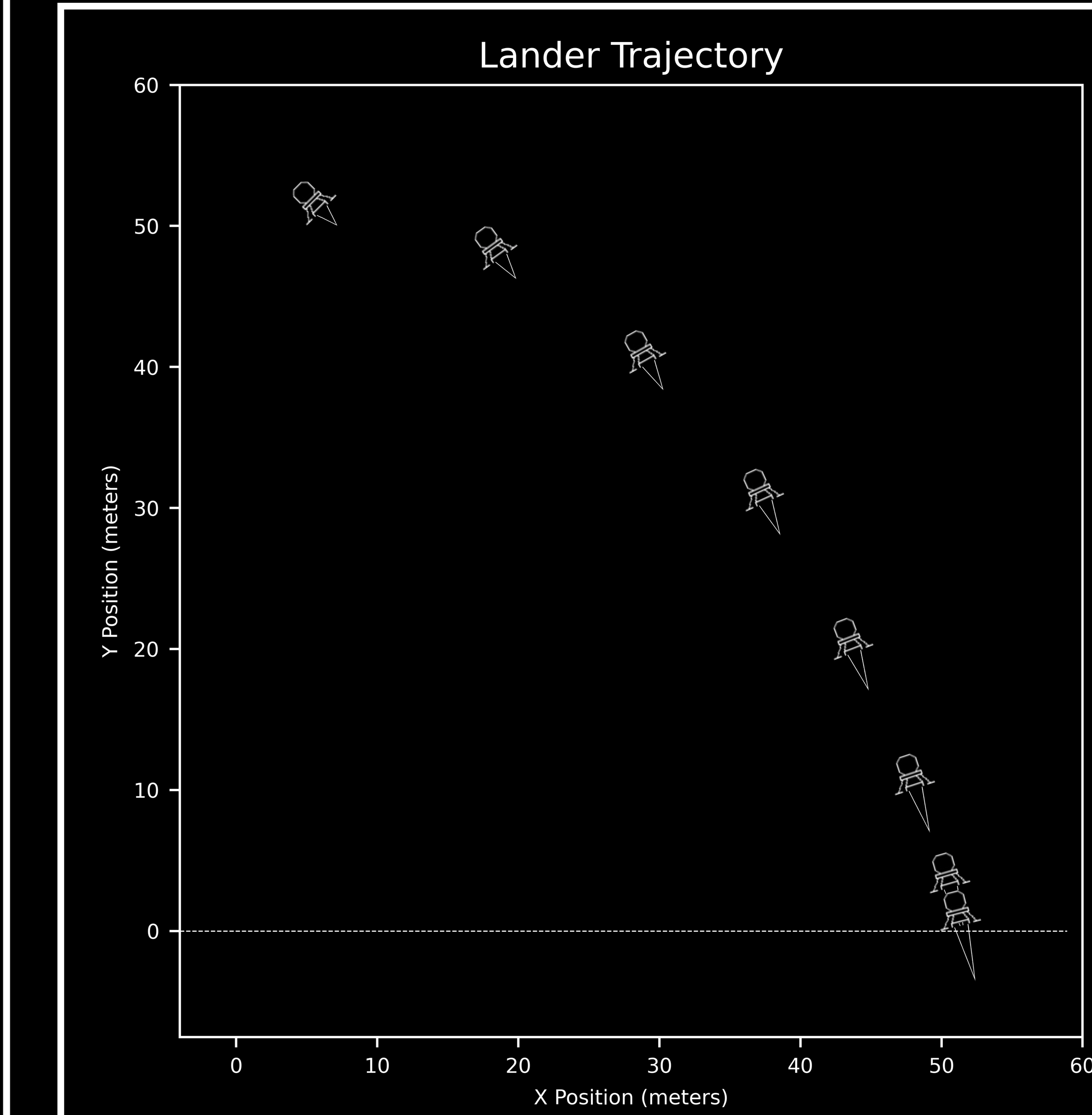

Figure 2: Optimal Trajectory

Our modeling produces numerical solutions that match both our intuition and the actual gameplay of "Lunar Lander." An example of a successful lander trajectory is given in Figure 2. We found that the horizontal velocity starts at x_0 and decreases steadily to zero, while the vertical velocity starts at zero, decreases as the rocket moves downwards, and then goes back up toward zero for a soft landing. We also see that the angle of the lander starts more steep and moves toward upright over time, and the the magnitude of the acceleration increases to slow the rocket down before it lands. See Figure 3 for plots of these results.

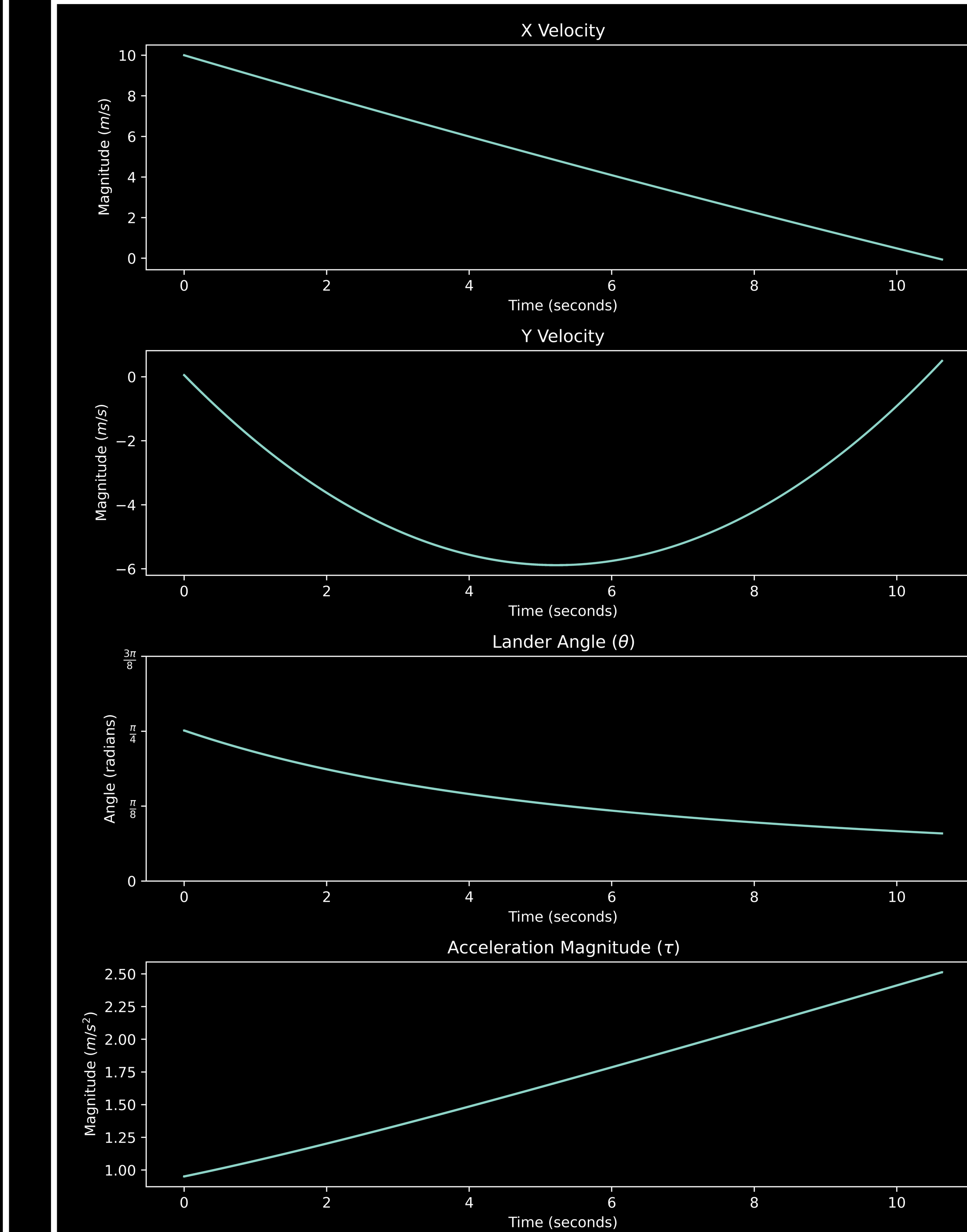

Figure 3: Optimal Velocity and Controls

CONCLUSIONS AND NEXT STEPS

In the future, we will consider a non-uniform lunar surface, as shown in the first image below, where the lander is rewarded for landing in flat areas and certain areas with point bonuses. Also, to mimic the actual gameplay, we plan to implement an inequality constraint on the lander's final angle so the craft lands upright. It would also be interesting to adjust our functional to account for obstacles, both on the ground and in the air, that the lander must dodge, such as moving asteroids or rocky areas that are impossible to land on.

References

[1] ilan. Lunar lander [b4x] - full game (source included), 2025. Accessed: 2025-04-07. URL: https://www.b4x.com/android/forum/threads/lunar-lander-b4x-full-game-source-included.137875/.

[2] T. Jarvis and J. Whitehead. *Foundations of Applied Mathematics, Vol. 4: Model Dynamics and Control.*

[3] S. Lee-Delisle. Lunar lander. URL: http://moonlander.seb.ly.

[4] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. https://doi.org/10.1038/s41592-019-0686-2 doi:10.1038/s41592-019-0686-2.