**CPE 2073**

Project: Lunar Lander Simulation
Due 6 pm, May 6

# Required Submission

- C source file
- Written report (2-4 pages)

You may work on the code individually or with a group of up to four persons. The C source file can be submitted as a group, but your report must be ***individually*** written and in .pdf (preferred), MS Word (.docx), or text format. Reports that have significant copying, even from the same group, will not be accepted. Indicate in your code (in comments) and in your report the other members of your group (if any).

# Report

## Format

Your report should be formatted in 12-point font, either Calibri (this font) or Times New Roman and should be single or 1.1-spaced (preferred). Minimum length is two full pages and maximum length is four pages. Graphic(s) are recommended but cannot cover more than 1/3 of the total report length.

The report part of the grade will consider the following:

- Complete and concise. The report should cover the topic accurately, completely, and without unnecessary verbiage. Focus on the technical aspects of the project.
- Appropriate use of graphics. Graphics (plots, images, etc.) are relevant and useful. Images are attributed properly if used from another source.
- Formatting of text. Text is properly broken into sections and paragraphs.

## Content

Your report should include the following at a minimum.

- Problem Statement. Explain the objective of the project is and why it’s important (not considering class credit or grade). A good problem statement will catch the reader’s interest and motivate them read the rest of the report.
- Describe your approach in solving the problem. Consider the physics, math, and coding.
- Show the results (plot, printouts, etc.).
- Describe challenges you had and how you overcame them.
- Brief conclusion about the project and what you learned.

# Background

This project simulates a greatly simplified version of the lunar lander from the Apollo 11 mission in 1969. The program should implement two modes:

1. “Game” mode, in which the user repeatedly enters a value representing the throttle on the rocket thrusters slowing the vehicle down as it descends.

2. Autopilot mode, in which a control function calculates the needed thrust automatically.

In both modes, the goal is to make a soft landing on the lunar surface before fuel runs out.

Apollo 11 lunar lander on the Moon. (Photo: NASA)

## Suggested Approach

For sake of simplicity, make the simulation run at one second intervals. Note: This is not one second of real time, but each step should represent one second of simulation time. This provides reasonable granularity without too much detail. The simulation will generally last no more than 150 steps (i.e. simulated seconds). The math required is very basic physics and is in discrete time, not continuous time.

The premise is this: The vehicle is descending straight down from given initial altitude and velocity. Negative values for velocity and acceleration indicate downward motion, while positive values indicate upward motion. The user or the control function (depending on mode: “game mode” or “autopilot”) can choose any amount of thrust, up to a maximum of 45 kN (kiloNewtons). The gravity of the moon causes a constant acceleration of -1.6 $m/s^2$ on the vehicle. A soft landing is one where the velocity at touchdown is one meter per second or less.

For the game mode, it is suggested that the thrust input be in kN and the value be converted to Newtons by multiplying by 1000. You’ll find that you need to run full thrust for several seconds, so entering 45 over and over instead of 45000 is much easier. The autopilot control function will most easily be implemented in Newtons, not kN, however.

Each kilogram of fuel produces 3000 N of force for one second, so the chosen thrust is divided by 3000 to determine how much fuel was burned in that one-second step. If the chosen thrust is greater than

3000 times the amount of fuel remaining, the thrust needs to be limited to consume only the remaining fuel and no more thrust can be produced after that.

As each simulated second passes, the current altitude, acceleration, velocity, mass, and fuel remaining need to be updated with new values reflecting the effects of the chosen thrust. These values should be printed to the screen and an output file in .csv format for later plotting. You can use the following pseudocode to update those values:

```
// Calculate acceleration
acceleration = (thrust/mass + gravity) * timeStep

// Update mass and fuel
mass -= thrust / 3000
fuel -= thrust / 3000

// Update velocity and height
velocity += acceleration * timeStep
altitude += velocity * timeStep
```

NOTE: Since timeStep = 1 in the above pseudocode, it can be omitted entirely.

The autopilot control function should take current altitude, mass, and velocity as parameters and return a thrust value that is designed to land the vehicle safely.

## Requirements

Create a structure that represents the current state, i.e. altitude, velocity, acceleration, mass, and fuel. Use this structure throughout your code.

You should include separate functions for the following tasks:

- Checking that the desired thrust does not exceed the maximum (45 kN) and is limited by the remaining fuel. This function should return the actual thrust allowed.
- Updating all of the state values (altitude, velocity, acceleration, mass, and fuel) with each step.
- The autopilot control: Determining a thrust based on the current state (this function is not called in game mode).
- Main function: Asks the user which mode (game mode or autopilot mode), opens the file for writing, runs the simulation loop, prints the state to the screen and to the file.

## Physical Details

| | |
|---|---|
| $H_0$ (initial altitude): | 15000 m |
| $v_0$ (initial velocity): | -325 m/s |
| Initial fuel remaining: | 1800 kg |
| $m_0$ (initial mass): | 9000 kg (including fuel) |
| $g_m$ (Lunar gravity): | -1.6 m/s$^2$ |
| Allowable thrust: | 0 - 45 kN |

Fuel consumption: (*T* / 3000 m/s) kg/s

Soft landing: $-1 \leq v_f \leq 0 \text{ m/s}$

*H* represents the current altitude (e.g. height), *v* is the current velocity, and *T* is the thrust.

## Suggested control formulas for autopilot

Try $H_f$ = 5 and $v_f$ = -0.9. You are free to experiment with other values and control formulas.

For higher altitudes ($H > H_f$):

$$a_d = \frac{\left(v^2 - v_f^2\right)}{2\left(H - H_f\right)}$$

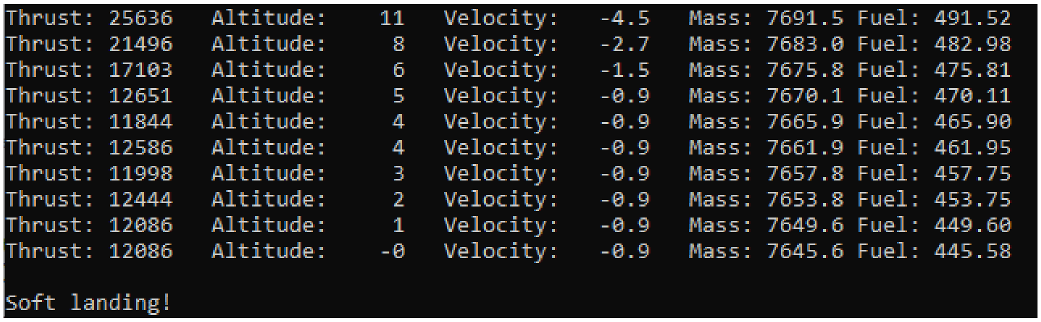
$$T = m(a_d - g_m)$$

Near landing ($H < H_f$):

$$T = -mg_m\left(\frac{V}{V_f}\right)$$

Note: $a_d$ is just a temporary variable representing the “desired” acceleration. The actual acceleration is calculated from the actual thrust.

## Example Output (in autopilot mode)



```
Thrust: 25636   Altitude:    11   Velocity:   -4.5   Mass: 7691.5 Fuel: 491.52
Thrust: 21496   Altitude:     8   Velocity:   -2.7   Mass: 7683.0 Fuel: 482.98
Thrust: 17103   Altitude:     6   Velocity:   -1.5   Mass: 7675.8 Fuel: 475.81
Thrust: 12651   Altitude:     5   Velocity:   -0.9   Mass: 7670.1 Fuel: 470.11
Thrust: 11844   Altitude:     4   Velocity:   -0.9   Mass: 7665.9 Fuel: 465.90
Thrust: 12586   Altitude:     4   Velocity:   -0.9   Mass: 7661.9 Fuel: 461.95
Thrust: 11998   Altitude:     3   Velocity:   -0.9   Mass: 7657.8 Fuel: 457.75
Thrust: 12444   Altitude:     2   Velocity:   -0.9   Mass: 7653.8 Fuel: 453.75
Thrust: 12086   Altitude:     1   Velocity:   -0.9   Mass: 7649.6 Fuel: 449.60
Thrust: 12086   Altitude:    -0   Velocity:   -0.9   Mass: 7645.6 Fuel: 445.58

Soft landing!
```