THE UNIVERSITY OF CHICAGO
MATTER, SPACE, ENERGY, AND TIME
ASTR 12600 1

Alejandro Alonso

Final Project

Simulating a 2-D Gravity-Assist Maneuver

Professor: Paolo Privitera & Edward W. Kolb

SUBMITTED AT 5:30PM ON DECEMBER 2ND, 2023

Contents

List of Figures

1 Project Introduction

In this project, I wrote a Python program to simulate a two-dimensional gravity-assist maneuver. Gravity-assists are currently used to give spacecraft (typically probes) extra kinetic energy to reach their destination. An example of this is the case of the Voyager probes. In this scenario, partial slingshots were used to reach the outer Solar System.

The example that originally inspired this project is the climactic slingshot maneuver in the science-fiction novel/movie *The Martian*. This case involves a gravity assist maneuver around the Earth that is used to conserve fuel and allow the spacecraft (*The Hermes*) to turn back towards Mars, and then a subsequent gravity assist maneuver is used around Mars to once again conserve fuel for the final journey back to Earth.

1.1 Link to Code

The code repository is on GitHub, accessible here: https://github.com/AAWorks/gravity-assist. The main simulation code is in *main.ipynb* and the major components (and the bulk of logic) is broken up into various files in the *components* folder. Package requirements (currently just *matplotlib*) are in *requirements.txt*.

2 Simulation Breakdown

The simulation is done through a Jupyter notebook, with the bulk of the logic separated into object-specific files to allow for better modular construction of the program. The simulation is done step-by-step, using only equations covered in lectures (i.e. without the standard gravity assist formula). The simulation stops when either the spaceship crashes or the distance between the spaceship and the planet becomes greater than the starting distance. The simulation is in 2-D and makes a few easily-inferable assumptions (spaceship does not have any thrust, is spherical, planet does not move, no atmosphere, no other nearby planets, etc.) to make the project more doable within the time frame given.

2.1 Setup

The simulation is carried out primarily by two distinct class instances. The first is the spaceship (instance of the *Spaceship* class in *spaceship.py*) and the second is the planet (instance of the *Planet* class in *planet.py*). The spaceship keeps track of its current position, past positions, x-velocity, y-velocity, x-acceleration, y-acceleration, and starting parameters and contains much of the utility required for completing a step. The planet keeps track of its starting parameters and contains utility for calculating its gravitational force.

2.1.1 Starting Parameters

The simulation accepts the following starting parameters:

1. Mass of the planet (*M_planet*)
2. Mass of the spacecraft (*M_spacecraft*)

3. Initial y-velocity of the spacecraft (*V_spacecraft*)
4. Radius of planet (*R_planet*)
5. Starting x-distance from spaceship to planet (*planet_starting_x_dist*)
6. Starting gravity threshold (*threshold*)

2.1.2 Start Details

To start, both the spaceship and the planet are placed on a Cartesian grid where each tick represents a meter. The spaceship starts at (0, 0), has an initial x-velocity of 0, and an initial acceleration of 0 (no thrust). The planet's start position is based on the distance from the spaceship it has to be for it to exert a gravitational force of *threshold* Newtons on the spaceship (calculated through the gravitational force equation in lectures 6 & 7). The starting y-position is then set at whatever is needed to reach this calculated distance assuming an x-distance of *planet_starting_x_dist*.

2.2 Step (Bulk of Calculations)

The simulation is organized in "steps," each of which represents one second. Before each step is completed, the gravitational force the planet is exerting on the spacecraft is calculated (using the gravitational force equation from Lectures 6 & 7) and used to determine the spacecraft's x-acceleration and y-acceleration (using the equations of motion from Lectures 5 & 6). This is then used to adjust the spacecraft's current X-Velocity and Y-Velocity (with equations from Lecture 5). The actual step then uses these velocities (which are in meters per second) to adjust it's position on the "grid." A series of simulation-termination checks are performed, such as does this step cause the spaceship to crash or is the spaceship now further away then it started. If checks clear, then the spaceship's position is updated, and the calculations for the next step commence.

2.3 Outputs

If the aforementioned checks do not clear, then the simulation stops. Using the object's internal properties, the simulation outputs the following:

1. (Boolean) Whether or not the craft crashed
2. (Integer) The number of steps, or time in seconds, taken for the maneuver
3. (Floating-Point Number) The time taken for the maneuver in hours
4. (Floating-Point Number) The spaceship's absolute change in direction in degrees
5. (Floating-Point Number) The final velocity in meters per second
6. (Visualization) A graph of the spacecraft's trajectory in relation to the planet

As an aside, due to the way the simulation is constructed, each class could be easily modified (2-3 lines) to collect more data than currently utilized in the final outputs of the notebook. For example, all three of acceleration, velocity, and gravitational force could be easily kept track of and then plotted, averaged, or otherwise parsed vs. time. The most interesting plot I tested was the one for the spaceship's trajectory so for clarity that was the only plot I explicitly included.

2.4 Running the Sim

The simulation is run by executing each cell in order (cells are labeled to make use easier).

*Note: For all graphs, the x-range and y-range are manually set. If the starting parameters are adjusted, the ranges may need to be adjusted as well to properly visualize the simulation.

3 Simulation Example

3.1 Starting Parameters

1. Mass of the planet: 6.39×10^{23} kg (Mars)
2. Mass of the spacecraft: 5.05×10^{4} kg (Avg. spacecraft mass)
3. Initial y-velocity of the spacecraft: $1.36 \times 10^{3} \frac{\text{m}}{\text{s}}$
4. Radius of planet: 3.3895×10^{6} m (Mars)
5. Starting x-distance from spaceship to planet: $4 \times R\_planet$ m
6. Starting gravity threshold: 1×10^{3} N

3.2 Pre-Sim Visualization

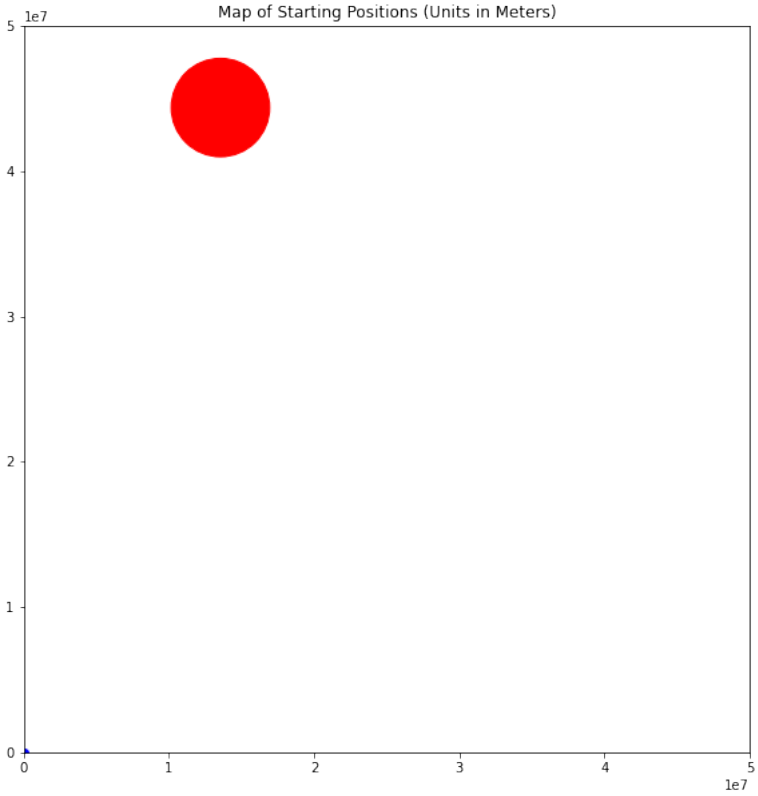

Figure 1: Example Sim: Starting Positions

3.3 Simulation Output

```
PRE-SIM
Starting Spaceship Values (kg, m/s): Spaceship(Mass=50500.0, Vy=1360.0, Vx=0, Pos=(0, 0))
Starting Planet Values (kg, m): Planet(Mass=6.39e+23, Radius=3389500.0, Origin=(13558000.0, 44384049.7757922))

SIM RESULTS
Spaceship Crashed: False
Time for Maneuver (s): 51024
Time for Maneuver (hr): 14.17
Heading Change (Degrees): 145.62180437873934
Final Velocity (m/s): 1360.0177021452496
```

Figure 2: Example Sim: Simulation Results

3.4 Post-Simulation Visualization

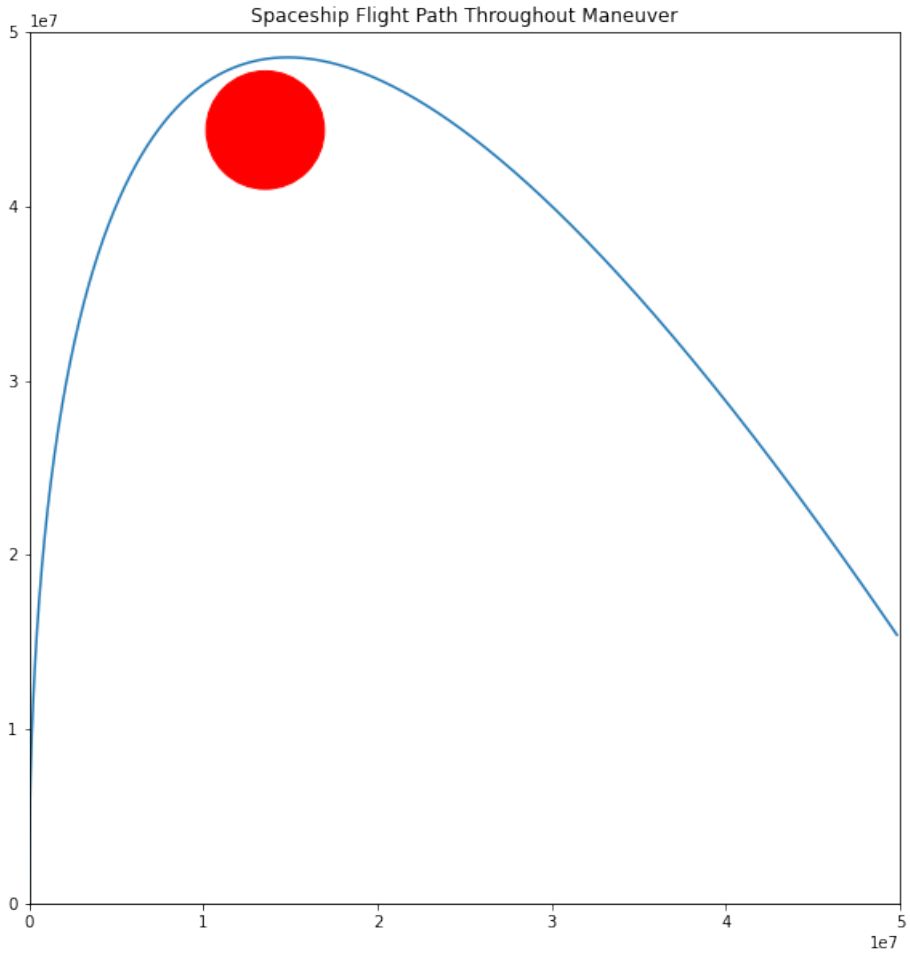

Figure 3: Example Sim: Final Spaceship Trajectory

4 References

1. Allain, Rhett. "How Does a 'Martian'-Style Gravity Assist Actually Work?" Wired, Conde Nast, 3 Jan. 2019, www.wired.com/story/how-does-a-gravity-assist-work/.

2. "Basics of Spaceflight: A Gravity Assist Primer - NASA Science." NASA, NASA, science.nasa.gov/learn/basics-of-space-flight/primer/. Accessed 2 Dec. 2024.

3. Privitera, Paolo, and Edward W. Kolb. "Lectures 5-7." ASTR 12600 Matter, Space, Energy, and Time. 2024, Chicago, University of Chicago.

4. Scott, Ridley, director. The Martian. Scott Free Productions, 2015.

5. Sharp, Tim. "How Big Is Mars?: Size of Planet Mars." Space.Com, Space, 2 Aug. 2012, www.space.com/16871-how-big-is-mars.html#:~:text=MarsóÀÊ%20mass%20is%206.42%20x,weigh%2038%20pounds%20on%20Mars.

Project Elements Checklist Form

This form must be completed and appended to your final project write-up or essay.

Synthesis of Lecture Material:
Please complete the table below, to summarize how your project synthesises material from at least 2 lectures. You may earn bonus points for including material from a 3rd lecture.

Lecture Number/Topic	Concepts included in the Project	Location in the Project/Essay/Write-Up
5	Newton's laws of motion, equations of motion	Project (spaceship.py file), Write-Up (Simulation Breakdown)
6	Newton's law of universal gravitation	Project (main.ipynb, spaceship.py, planet.py), Write-up (Sim breakdown)
7	Gravitational Slingshot	Main Project Concept, Project (all files), Write-Up (throughout)
7	Gravitational Constant	Project (constants.py, planet.py, main.ipynb)
(Optional Bonus)		

Quantitative analysis:

a) Briefly describe the quantitative analysis component of your project:

The quantitative analysis component is present in both the main project (a Python-based timestep-simulation program) and the write-up which includes some graphs generated by the program for clarity.

b) Where is the quantitative component(s) found, within your project, essay, and/or write-up?

Project and write-up.