A Space Voyage: Part 1

Objectives

In this program you will model the motion of a spacecraft moving near the Earth. You will use your working program to explore the effect of the spacecraft's initial velocity on its trajectory. Before doing this activity you should have read Section 3.6 of the *Matter & Interactions 4e* textbook, which discusses the structure of a computational model that includes the gravitational force. This activity will require you to bring together ideas from previous activities in which you (1) wrote a program to calculate the gravitational force between two objects, and (2) used an arrow object and a scale factor to visualize a gravitational force.

After completing this activity you should be able to:

- Identify what quantities must be calculated inside a computational loop
- Write a program to model the motion of two gravitationally interacting objects
- Explain why the initial velocity of an object affects its trajectory as it moves near a more massive object
- Draw a diagram showing the directions of momentum and net force at different locations along an elliptical orbit

1 Explain and Predict

Study the following VPython program carefully. Make sure you understand the whole program, but don't run the program yet. Reading and explaining program code is an important part of learning to create and modify computational models.

```
from __future__ import division, print_function
from visual import *
scene.width = scene.height = 800

G = 6.7e-11
mEarth = 6e24
mcraft = 15e3
deltat = 60

Earth = sphere(pos=vector(0,0,0), radius=6.4e6, color=color.cyan)
craft = sphere(pos=vector(-10*Earth.radius, 0,0), radius=1e6,
               color=color.yellow, make_trail=True)
vcraft = vector(0,2e3,0)
pcraft = mcraft*vcraft
t = 0
scene.autoscale = False ##turn off automatic camera zoom

while t < 10*365*24*60*60:
    rate(100)
    craft.pos = craft.pos + (pcraft/mcraft)*deltat
    t = t+deltat
```

Without running the program, answer the following questions:

⇒ What is the physical system being modeled?

⇒ In the real world, how should this system behave? On the left side of your whiteboard or paper, draw a sketch showing how you think the objects should move in the real world.

⇒ Will the program as it is now written accurately model the real system?

⇒ On the right side of the whiteboard or paper, draw a sketch of how the objects created in the program will move on the screen, based on your interpretation of the code.

Check your work before continuing.

2 Modify and Extend the Model

⇒ Run the program.
⇒ How did your prediction compare to what you saw? Did something happen that you didn't expect to happen?
⇒ How should the program be changed so that it is a physically reasonable model of the system?
⇒ Modify the program. Run it and compare its behavior to the behavior you expect from the real system.

Check your work before continuing.

3 Visualize Momentum

In a previous activity, you used arrows to visualize gravitational force vectors. In this program, you will use an arrow to visualize the momentum of the spacecraft. Note that you want one arrow that moves with the spacecraft–you don't want to create many arrows. Therefore you will create the arrow before the loop, and change its `pos` and `axis` inside the loop, just after updating the `pos` of the spacecraft. You may review the video on scale factors if necessary: *VPython Instructional Videos: 5. Scale Factors* `http://vpython.org/video05.html`

⇒ Print the initial momentum of the spacecraft.
⇒ Use this value to decide on a value for a scale factor that will make the arrow representing momentum a reasonable length in the display.
⇒ Before the computational loop:
  ⇒ Add a line of code that assigns the symbolic name `sf` to the scale factor.
  ⇒ Create an arrow named `parr` to represent the craft's momentum.
⇒ Inside the loop, add two lines of code (after the position update) like this:

```
parr.pos = craft.pos
parr.axis = pcraft * sf
```

⇒ Once you have seen the entire orbit, you may have to adjust the scale factor.

4 Testing a Value of a Variable

An `if` statement can be used to instruct VPython to do something only in a particular situation. The action to take is indented after the `if` statement. For example, consider the following code fragment:

```
a = 3
if a < 4:
    sphere(color=color.green)
box(pos=vector(3,3,0), color=color.cyan)
```

⇒ Try the code above in a new program window. What does it do?
⇒ What happens when you replace `a = 3` with `a = 8`?

4.1 Detecting a Collision

If your spacecraft collides with the Earth, the program should stop.

⇒ Add code similar to the following inside your loop (using the name you defined for the relative position vector between the spacecraft and the center of the Earth):

```
if mag(r) < Earth.radius:
     break  ## exit from the loop
```

The `break` instruction tells VPython to get out of the loop and go to the first instruction after the loop (if there is one.) Because the `break` instruction is indented after the `if` statement, it will be executed only if the `if` test returns a value of `True`.

5 Direction of Momentum

Record your answers to the following questions:

⇒ For this elliptical orbit, what is the direction of the spacecraft's momentum vector? Tangential? Radial? Something else?
⇒ What happens to the momentum as the spacecraft moves away from the Earth?
⇒ As it moves toward the Earth?
⇒ Why? Explain these changes in momentum in terms of the Momentum Principle.

6 Effect of the Initial Velocity

⇒ Approximately, what minimum initial speed is required so that the spacecraft "escapes" and never comes back? You may have to zoom out to see whether the spacecraft shows signs of coming back. You may also have to extend the time in the while statement.
⇒ What initial speed is required to make a nearly circular orbit around the Earth? You may wish to zoom out to examine the orbit more closely.
⇒ How does increasing the initial speed affect the orbit? Explain this by considering the first few time steps.
⇒ How does decreasing the initial speed affect the orbit? Explain this by considering the first few time steps.

7 Adding an Arrow to Represent Force

⇒ Choose an initial speed that produces an elliptical orbit.
⇒ Add a second, different colored arrow representing the net force on the spacecraft. This arrow should also move with the craft. You'll need a different scale factor for this arrow.
⇒ Are the net force on the craft and the momentum of the craft in the same direction?
⇒ What is the relative direction of these arrows when the craft is slowing down?
⇒ Speeding up?
⇒ Draw a diagram showing the directions of the craft's momentum and the net force on the craft at 6 different locations along an elliptical orbit. At each location note whether the speed of the craft is increasing, decreasing, or momentarily not changing.