A Spacecraft Voyage, Part 1: Spacecraft and Earth

**OBJECTIVES**

In this program you will model the motion of a spacecraft. You will use your working program to explore the effect of the spacecraft's initial velocity on its trajectory.

**TIME**

You should finish this activity in 45 minutes or less.

**1** **Explain and Predict Program Visualization**

 A minimal working program has been provided below.

**DO NOT RUN THE PROGRAM YET. Read through the program together. Make sure everyone in the group understands the function of *each* program statement. Reading and explaining program code is an important part of learning to create and modify computational models.**

from \_\_future\_\_ import division

from visual import \*

scene.width =800

scene.height = 800

#CONSTANTS

G = 6.7e-11

mEarth = 6e24

mcraft = 15e3

deltat = 60

#OBJECTS AND INITIAL VALUES

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)

trail = curve(color=craft.color) ## for leaving a trail behind the craft

vcraft = vector(0,2e3,0)

pcraft = mcraft\*vcraft

t = 0

#CALCULATION LOOP: ALL REPEATED CALCULATIONS GO INSIDE THE LOOP

while t < 10\*365\*24\*60\*60:

 rate(100) ## slow down motion to make animation look nicer

 craft.pos = craft.pos + (pcraft/mcraft)\*deltat

 trail.append(pos=craft.pos) ## this leaves a trail behind the spacecraft

 t = t+deltat

After reading *every* line of 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 the whiteboard, 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? **DO NOT RUN THE PROGRAM YET.** Study the program again. On the right side of the whiteboard, draw a sketch of how the objects created in the program will move on the screen, based on your interpretation of the code.

• Run your program *after* everyone agrees on both predictions. Discuss how closely your prediction of what the program would do matches what you see in the visual output.

**2** **Run the program**

• How did your prediction compare to what you saw? Did something happen that you didn't expect to happen?

**3** **Modify the program**

• **What calculations are needed to express the interactions between the spacecraft and the Earth?** Recall the previous discussion of iterative prediction of motion. All repeated calculations go inside the loop in the program.

 - Calculate the (vector) forces acting on the system and their sum, .

 - Update the momentum of the system: .

 - Update the position: .

 - Repeat.

• **Modify the program with these calculations. Where should you place these new lines of code in the program?** The previous two programs you have written may be helpful resources.

• **Use symbolic names defined in the program. You may need to add other symbolic names in your calculations.**

• **Finally, run the program. Is the behavior what you expected? If not, check that your code correctly expresses the relevant physics, and that you have placed repeated calculations inside the while loop.**

**4** **Visualize the momentum vector with an arrow**

In a previous program, you used arrows to visualize gravitational force vectors. In this program, you will use an arrow to visualize the momentum of the spacecraft.

You will create an **arrow** before the **while** loop and update its position and axis inside the loop, as the spacecraft's momentum changes.

**It is very important NOT to create the arrow inside the loop, because then you would have many arrows rather than one. Just as you create ONE sphere to represent the spacecraft, and then inside the loop you continually modify the sphere's position, so also you create ONE arrow to represent the momentum, and then inside the loop you continually modify the arrow's position and axis.**

You need to know the approximate magnitude of the momentum in order to be able to scale the arrow to fit into the scene, so just before the loop, print the momentum vector:

**print('p=', pcraft)**

In the **#CONSTANTS** section of your program, add this scale factor

**pscale = ??**

Also insert the following statement **in the #OBJECTS section of your program, NOT inside the loop**, because that would create a very large number of arrows:

**parr = arrow(color=color.green)**

This statement creates an arrow object with default pos and axis attributes.

• **Comment out any print statements that are inside your loop, because they slow the program down.**

• **Inside your loop, update the pos attribute of the parr arrow object to be at the center of the spacecraft, and on a separate line update its axis attribute to represent the current vector value of the momentum of the spacecraft (multiplied by pscale).**

parr.pos = ?

parr.axis = ?

• **You may have to adjust the scale factor once you have seen the full orbit.**

**5** **Answer questions about changes in the spacecraft's momentum**

• **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.**

• **Compare your answers to those of another group.**

**6** **Answer questions about the effect of initial velocity on the motion**

• **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.**

• **Why does changing the initial velocity have an effect on the orbit?**

**7** **Optional: 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 distance between the spacecraft and the center of the Earth):

  **if rmag  Earth.radius:**

 **break**

This code tells VPython to get out of the loop if the spacecraft touches the Earth.

**8** **Report your work on your blog**

Include answers to each of the questions plus a video of your ship in motion about the earth.