A Spacecraft Voyage: Earth and Moon

Now that you've had some experience with generating full programs from scratch, we will sometimes provide starter programs that contain some of the necessary code but not all of it. This will save time and allow you to focus on modeling the physics rather than worrying about every little coding detail that must be included.

1 Explain and Predict Program Visualization

Copy the text from the file provided into an editor window, but **don't run the program yet.**

Go through each line of code in the program, making sure your group understands the purpose of each line and what it will do. On a whiteboard, draw a sketch of what you think will happen when the program runs. Make a note of what commands are present, as well as what's missing to make the program be a true simulation.

Once everyone in the group has agreed on what the program will do, run it and see how your predictions matched up.

2 Adding in the Physics

Obviously, the big thing that's missing is the force of gravity causing the craft to orbit the Earth.

Add all the commands necessary to calculate the force of gravity acting on the craft due to the Earth, as well as the commands necessary to update the craft's momentum or velocity as it's affected by gravity.

If you're having trouble, remember that you can go back and look at previous programs submitted in WebAssign. Last week you wrote a program to model the force of gravity, and in previous weeks you used animated programs that updated the object's momentum or velocity in each loop iteration.

3 Refining the Program

If your program is written correctly up to this point, you should see the craft orbiting the Earth. Now we will refine the program some to better evaluate it.

3.1 Crashing into the Earth

Right now, the spacecraft would just go right through the Earth if it flew toward it, since VPython doesn't assume that the spheres are solid.

Insert the following text into the `while` loop, after the position is updated. **Type them in, do not copy and paste (see below).**

```
if rmag < Earth.radius:
        break
```

Do not copy and paste these lines. There is a hidden symbol so that the pdf software would recognize the indentation, and it will give you an error if you just paste it into VPython. Make sure that the `if` line is part of the loop (indented) and the break line is indented relative to the rest of the loop. If you named the magnitude of your $\vec{r}$ vector something other than `rmag`, change this line of text accordingly.

Test it out. Change the initial velocity of the craft to ¡0, 0, 0¿. The craft should be pulled straight toward the Earth and then stop when it makes contact.

3.2 Momentum and Force Arrows

Next, insert arrows to draw the momentum of the craft and the gravitational force acting on it. This will help visualize what's happening in the simulation.

In the `Constants` section, add the following definitions:

```
pscale = 0.3
Fscale = 3e3
```

In the `Objects and Initial Values` section, add the following objects:

```
momentumarrow = arrow(color=color.red)
forcearrow = arrow(color=color.green)
```

Inside the `while` loop, after the craft's position is updated, add the following lines:

```
momentumarrow.pos = craft.pos
momentumarrow.axis = pcraft*pscale
forcearrow.pos = craft.pos
forcearrow.axis = Fgrav*Fscale
```

Note: If you called your force of gravity something besides `Fgrav`, modify the last line accordingly.

These commands create two arrows, then continuously update them so that they both move along with the craft as its position changes, with the red one being set to the craft's momentum and the green one being set to the force of gravity acting on the craft. The scale terms are just so that the arrows display at an appropriate size relative to the other objects.

3.3 Circular Motion

It was probably already clear that the craft was not in a circular orbit because the Earth was not in the center of the orbit. But with the momentum and force arrows, it's even more clear that this orbit is not circular.

For uniform circular motion, the craft would maintain a constant speed, so the magnitude of the momentum vector wouldn't change. The craft would also maintain a constant distance from the

Earth (since a circle has constant radius), so the magnitude of the force of gravity wouldn't change. Finally, in uniform circular motion, the momentum and force vectors would always be perpendicular to each other.

Try to determine what initial velocity value will produce a uniform circular orbit.

You can do this by trial and error (just change the velocity and see if your change makes it more or less circular) or by attempting to calculate what the speed should be. To do the calculation, recall that the net force during uniform circular motion is related to the speed by

$$\left|\vec{F}_{\mathrm{C}}\right| = \frac{mv^2}{r}$$

and the centripetal force is being provided by gravity

$$\left|\vec{F}_{\mathrm{grav}}\right| = \frac{Gm_1m_2}{r^2}$$