

PHY307, Science and Computers I

Lab #7, Part 2, October 3, 2002 (try to finish today!)

Cultivating Colorful Continuous Closed Curves using Coordinates

Summary of what you will do:

You will study an alternate solar system, where there is a planet similar to Mars, but which is much more massive than our Mars.

What effect would this massive Mars have on the Earth's orbit? (Leading to changes in seasons or plunging into the Sun?) Is it conceivable for the planets to behave chaotically?

This lab will be preceded by a review of vectors, the forces that act on planets, and of part 1 of the lab.

LOAD YOUR MODULES

1. Log in and continue your report for Lab #7 – this is Part 2.
2. There are two modules available under Codes on the web page – save these modules as **gravity.py** and **fakesolarsystem.py** on your home drive. Copies of these codes are listed at the end of the lab.
 - a. **gravity.py** defines the forces of gravity and updates the position and curve that records the path of a planet. It has two functions. The function **gravaccel** computes the acceleration due to gravity on a planet at a given position due to a planet having another position and mass. The function **update** updates the position of a planet based upon its velocity. This function also updates the velocity by adding up all of the acceleration vectors that act on the planet. It tracks the position of the planet by appending points to the curve associated with the planet, at the position of the planet.
 - b. **fakesolarsystem.py** gives coordinates, velocities, masses, and visible sizes to the Sun, the Earth, and to Mars. Mars's position and velocity are not quite accurate and you will also vary its mass.
3. First, investigate the dynamics of a solar system that resembles ours. The **fakesolarsystem.py** module has zero mass for Mars, which is approximately true as far as the Earth is concerned. To get the simulation started, you need to write a program, saved as **theplanets.py**. This program will need to:
 - a. **import** the definitions from **gravity.py** and **fakesolarsystem.py**.
 - b. Set **scene.autoscale** to **0**.
 - c. Have a **while 1:** loop that, at a particular rate, executes the function **update** for all of the objects in the list **planets**.

Execute this program. PUT YOUR PROGRAM INTO YOUR REPORT.

RECORD HOW THE MOTION OF THE EARTH AND MARS APPEARS.

MAKE MARS MASSIVE

4. Increase the mass of Mars to 0.02 (the Sun has a mass of 1.0) and repeat your simulation again. NOTE: whenever you modify **fakesolarsystem.py**, make sure to save it so that your program **theplanets.py** will load the updated module. Try a mass of 0.05 and one or two other masses – RECORD HOW THE MASS OF MARS AFFECTS THE CHARACTERISTICS OF THE EARTH'S ORBIT. Include a few snapshots. DISCUSS what you see.
5. Finally, see if you can show how a massive Mars can induce chaos. To do this, you want to study the butterfly effect. How could you compare the motion of the Earth with that of a second Earth that had slightly different starting conditions? Suppose this second Earth (**earth2**) had an initial position of **(1.999999, 0.002, 0)**, but was identical to the **earth** in all other respects. Can you modify fakesolarsystem.py to add this second Earth (say with color green and track color white)? Once you do this, run **theplanets.py** again and RECORD WHAT YOU SEE FOR VARYING VALUES OF THE MASS OF MARS.
6. SUMMARIZE the results of today's work.

MODULES

gravity.py

```
## Oct. 1, 2002, A. Middleton
## gravity.py
## This module defines two functions - one calculates the
## acceleration due to gravity on a body at position ra due
## to an object at position rb with mass mb. The second function
## updates the position, velocity, and track of a planet. The
## planet must be initialized properly (see fakesolarsystem.py)
## and have a list of "others": these are the planets which exert
## a force on it.

from visual import *

forcetype = -3

dt = 0.01

## Acceleration in the direction separating objects a and b, with magnitude
## 1/distance**2. ("one over r squared" law.)
def gravaccel(ra, rb, mb):
    return(-(ra-rb)*mag(ra-rb)**forcetype*mb)

## Update position based on velocity. Velocity changes by acceleration,
## with acceleration given by the sum from all 'others'. Also update track.
def update(planet):
    planet.pos = planet.pos + dt * planet.vel
    totalaccel = vector(0,0,0)
    for other in planet.others:
        totalaccel = totalaccel + gravaccel(planet.pos, other.pos, other.mass)
    planet.vel = planet.vel + dt * totalaccel
    planet.track.append(pos=planet.pos)
```

fakesolarsystem.py

```
## Parameters for the model solar system
## Used by the main program theplanets.py
##
## Each planet needs to be defined AND AT THE END
## all planets must be put into the list 'planets'
##     - All objects that exert gravity need a mass.
##     - All objects subject to gravity need a curve attribute named
##       'track' for tracing the orbit, a velocity velocity named 'vel',
##       and list of objects that exert gravity on them ('others')

from visual import *

##### sun ### (fixed in space by assumption)
sun = sphere(pos=(0.,0.,0.), radius=0.2, color=color.yellow)
sun.mass = 1

##### bigmars ### (this object will affect Earth,
##### but is not affected by the Earth.)
##### note that the z-position is not zero - its orbit will be tilted
##### relative to the unperturbed orbit of the Earth. Set
##### starting position, velocity, mass, and color of track.
bigmars = sphere(radius=0.1, color=color.magenta, pos=(3,0,0.2))
bigmars.vel = vector(0,0.5,0)
bigmars.mass = 0.0
bigmars.others=[sun]          ### sun affects the bigmars
bigmars.track=curve(color=color.red)

earth = sphere(radius = 0.1, color=color.blue, pos = (2,0,0))
earth.vel = vector(0,0.6,0)
earth.others=[sun,bigmars]    ### bigmars and sun affect the earth
earth.track=curve(color=color.cyan)

## list of the planets - things with tracks and dynamics - sun is fixed
planets = [bigmars, earth]
```