Computer Model of Spring-Mass System, Part 3 of Springs Lab

QUESTIONS: Modeling the real world:

Q1: Can a computer program of a mass-spring system, based on the finite-time form of the momentum principle

$$\vec{p}_f = \vec{p}_i + \vec{F}_{\text{net}}\Delta t$$

and using your measured data on the mass and spring you used in a previous experiment, accurately predict the oscillation period you measured in your experiment?

Q2: What initial conditions must you specify in your program in order to get your virtual mass-spring system to oscillate in all three dimensions, instead of just staying in one plane?

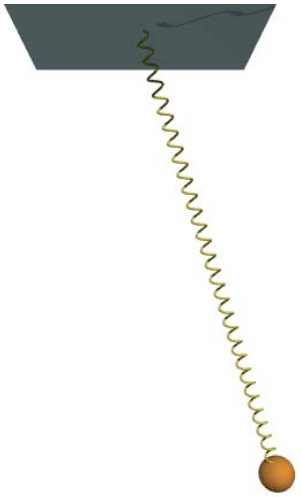
Q3: What are the limitations of your model? What real-world factors are left out?

Planning (on whiteboard)

Work with your partner(s) to write out answers to the following questions.

1. Force (freebody) diagram)

The picture shows a snapshot of a mass oscillating on a spring at a particular instant. The mass is moving at this instant.



Consider the mass to be the system.

1.a. List objects interacting with the system.

1.b. Draw a freebody diagram showing all significant forces acting on the system at the instant shown in the picture. **You will need to draw the freebody diagram (complete with force names!) in your lab report, using the drawing tools in *Word*.**

2. Outline the calculation to be done in the program

Your program will have several sections. As a group, lay out how the program will be organized. On your paper, indicate what will need to be in each section. Don't write actual code – just show what goes in each section.

```
## constants

## create objects

## initial values

## calculation loop - calculations which must be repeated over
and over
```

For example, here is an outline for the fancart program you wrote early in the semester:

```
## constants
    deltat
    mass of cart

## create objects
    track (a box)
    cart (a box)

## initial values
    velocity of cart (a vector)

## calculation loop - calculations which must be repeated over and
over
```

$\vec{p}_f = \vec{p}_i + \vec{F}_{\text{net}}\Delta t$ `## apply momentum principle`

$\vec{r}_{\text{f}} = \vec{r}_{\text{i}} + \vec{v}_{\text{avg}}\Delta t$ `## update position of cart`

CHECKPOINT VP1: Skeptic, make sure everyone understands what is going on

3. Program shell

On the class website (Files page) you will find the shell of a program to use for this project. You will need to add code to the shell to create a running program that allows you to answer the questions at the beginning of this handout. **WARNING:** On older versions of VPython, you may need to change "helix" to "cylinder" and eliminate parameters unique to the helix before the program will run.

3.1. Save the shell to your directory or the /Users/Shared directory.

3.2. READ the code in the shell. Be prepared to explain what each line does.

3.3. Put the data from your part 2 lab measurements into the program (mass, spring constant, relaxed length.) If you have not yet completed part 2, use m = 0.026 kg, k_s = 1.5 N/m, L_0 = 0.21 m.

3.4. Figure out what "pos" and "axis" mean for a helix object. You may need to consult the online reference manual (pull down the Help menu in IDLE, choose "visual", then choose "reference manual".)

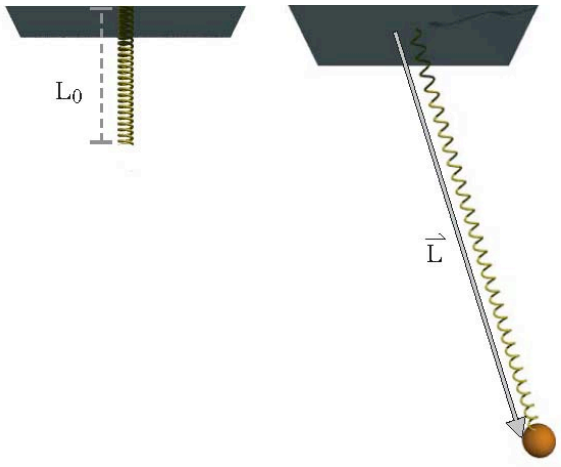
4. Calculating the spring force as a vector

Consider the diagrams on the next page. When the spring is stretched as shown on the right side of the diagram, the force on the ball due to the spring is directed along the spring, toward the point where the spring is attached to the ceiling.

We define a vector $\vec{L}$ pointing from the attachment point (`spring.pos`) to the center of the ball (`ball.pos`), as shown in the diagram. The direction of $\vec{F}_{\text{spring}} = -(k_s s)\hat{L}$ is given by the unit vector $\hat{L}$ (L-hat).

The stretch is a scalar value, which in this case is positive, since the length of the spring is currently longer than its relaxed length.

$$s = |\vec{L}| - L_0$$



For positive values of *s*, the spring force is in the direction of $-\hat{L}$ (toward the ceiling). So the force on the ball due to the spring is:

$$\vec{F}_{\text{spring}} = -(k_s s)\hat{L}$$

For negative values of *s*, the spring is compressed, and the spring force is in the direction of $\hat{L}$ (away from the ceiling). Since *s* is a negative number, the same equation describes the spring force:

$$\vec{F}_{\text{spring}} = -(k_s s)\hat{L}$$

4.1. Write the necessary code in your program to calculate the force on the ball due to the spring. (Does this calculation go inside the loop or before the loop? Why?)

4.2. Use this force in your calculation of the net force on the ball (**what other force must you include?**) *Make sure the expression for the net force works in more than one dimension.*

4.3. Apply the momentum principle

4.4 Update the position of the ball

4.5 Update the axis of the spring

4.6 Run your program. Does its behavior look reasonable?

CHECKPOINT VP2: Skeptic, make sure everyone understands what is going on. Verify that the program runs properly.

5. Using a graph to answer Question 1 (page 1):

Have your program produce a graph of the y-coordinate of the ball's position vs. time. Use this graph to determine the period of the oscillating system in your computer model. Here are some reminders on how to do this:

This statement at the beginning of your program imports the graphing module:

```
from visual.graph import *
```

In the ##objects section create a gcurve object, for plotting the position of your ball

```
ygraph = gcurve(color=color.yellow)
```

Inside the loop, after updating the momentum and position of the ball and the time, add the following statement:

```
ygraph.plot(pos=(t, ball.pos.y))
```

As you might expect, this plots a point on a graph at a location given by t on the horizontal axis, and the y component of the position of the ball on the vertical axis.

Questions to answer about the period (related to Question 1 on page 1):

- Why doesn't the graph cross zero?
- What is the period of the oscillations shown on the graph? (You will find it easier to read the period off the graph if you change the `while True:` statement to make the program quit after a small number of oscillations.)
- How does the period of your model system compare with the period you measured for your real system?
- What does the analytical solution for a spring-mass oscillator predict for the period?
- Should these numbers be the same? If they are not, why not?
- Make the mass 4 times bigger. What is the new period? Does this agree with theory? (Afterwards, reset the mass to its original value.)
- Make the spring stiffness 4 times bigger. What is the new period? Does this agree with theory? (Afterwards, reset the spring stiffness to its original value.)
- Make the amplitude twice as big. (How?) What is the new period? Does this agree with theory? (Afterwards, reset the amplitude to its original value.)

Write down the answers to the above questions, since you will need to include them in your lab report. Be prepared to relate them to other parts of the lab.

6. How can you make the system oscillate in all dimensions, instead of staying in one plane (Question 2 on page 1)? (Playing with a real spring may be helpful.)

The initial conditions for your model system are: 1) The initial position of the ball (which also determines the initial stretch of the spring) 2) The initial momentum of the ball

Find initial conditions that make your model system oscillate in all dimensions (that is, the motion of the ball is not confined to a single plane). It will be easier to tell what your system is doing if you add a trail (see below).

7. Comment out the graphing command and add a trail, so you can track the ball's motion in 3D

Add to ## objects

```
trail = curve(color=ball.color)
```

Add to the end of ## calculation loop

```
trail.append(pos=ball.pos)
```

Rotate the display to make sure the ball's motion is not confined to one plane. Play around with the different initial conditions.

8. List at least two real-world factors that have not been taken into account in your model. Under what circumstances (what kind of motion or initial conditions) would they be important? Include your answers in your lab report.

CHECKPOINT VP3: Skeptic, make sure everyone understands what is going on. Verify that the program runs properly.

9. The recorder should turn in your program to WebAssign. Make sure to read the directions given there.