Group Members: ____

Computer models of motion: Iterative calculations

OBJECTIVES

In this activity you will learn how to:

- Create 3D box objects
- Update the position of an object iteratively (repeatedly) to animate its motion
- Update the momentum and position of an object iteratively (repeatedly) to predict its motion

TIME

You should plan to finish this activity in 50 minutes or less.

I. Setup statements

In this task, we are going to use VPython to model a moving cart on a 2.0 m track.

1. In a browser, go to the webpage **www.glowscript.org** to get to the Glowscript home page.
2. In the right hand corner click "**Sign In**" and login using your existing Google account from Lab 1.
3. You are now on the Glowscript program development page where you can either run programs that you have previously written or create new programs. We want to click on the "**Create New Program**".
4. In the pop-up display give the program the name "**Cart**"

a. Creating objects

5. We use a white box to represent the track. Type the following command and run the program.
   track = box(pos=vector(0,-.05, 0), size=vector (2.0, 0.05, .10), color=color.white)

6. We will now use a red box whose center is at the origin to represent a moving cart located at the origin. Type the following command and run the program.
   cart=box(pos=vector(0,0,0), **size=vector**(0.1, 0.05, 0.06), color=color.red)

Test Your Knowledge

Reposition the cart so its left end is aligned with the left end of the track.
Run the program and once you are sure it is working have your instructor sign off on your work.

____
Instructor

b. Initial conditions

To model the motion of an object over time, we need to know the value of two vector:

- initial position; and
- initial velocity.

You have already given the cart an initial position. Now you need to give it an initial velocity, **velcart**. If you push the cart with your hand, the initial velocity is the velocity of the cart just *after* it leaves your hand.

7. Give the cart a velocity of 0.5 m/s in the +x direction by adding the following command

   velcart = vector(0.50,0,0)   **#sets the carts velocity at 0.5 m/s in the +x direction**

The # symbol tells the computer that everything type after the # is a comment and should be ignored. Using comments in your code makes it easier for people to follow your program. You can also use comments to find errors in your code by commenting out lines that you believe may contain an error.

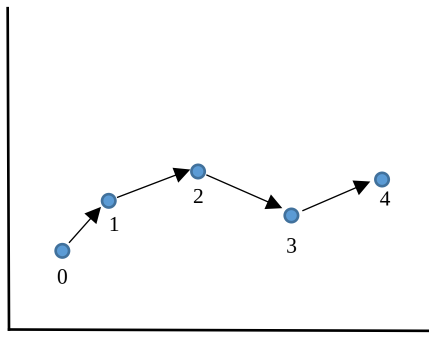
c. Time step and total elapsed time

One way we can describe the general motion of an object is what is called a local formalism of motion. In this approach, the object's next position vector after a short time step Δt is calculated using the object's initial position vector and its initial velocity vector. Mathematically, this can be stated as:

$$\vec{r}_f = \vec{r}_i + \vec{v}_i \, \Delta t$$

This equation comes from the definition of the average velocity and is thus only correct if the velocity is constant. However, if Δt is very small then the formula will approximate the new location of the object even if the velocity of the object is not constant.

For most problems, we will want to describe the motion of an object over a time period, t, that is much longer than our equation is valid. To get around this issue, we will break up our time period into thousands or even millions of small time periods, Δt, and calculate how the object moves from point to point as shown in the figure below. This is known as an iterative approach. We start initially at location zero at time, $0 = 0 * \Delta t$ with an initial velocity depicted by the arrow. This is called the "zeroth iteration". After a time lapse of $\Delta t = 1 * \Delta t$, the object has moved to location 1. This is called the "1st iteration". We now want to calculate the object's position at time $2\Delta t$ which is depicted as location 2. To do this, we need to not only know the object's position at location 1, but also its velocity at location 1. Thus, our local formalism of motion must also have an equation for determining how the velocity of the object changes with time. This is what acceleration tells us.



The object's velocity a short time, Δt, later can be determined using the object's initial velocity and initial acceleration using the equation:

$$\vec{v}_f = \vec{v}_i + \vec{a}_i \, \Delta t$$

Looking at this equation, you probably notice that we will only be able to solve it if are told how the acceleration changes with time. One case where this is occurs is when the acceleration is constant. Later in the course, we will cover the general method for obtaining the acceleration of an object based upon the forces acting upon an object which is called Newton's 2nd Law.

8. Let us assume that our cart has constant velocity so our acceleration is zero. Add the following command to your program:

```
accelcart = vector(0,0,0)          #cart's acceleration is set to zero
```

9. We now we set the initial time to zero, and set our step size, i.e. Δt, to 0.01. Add the following commands and run your program.

```
t = 0                              #initial time is 0 s
deltat = 0.01                      # time step size
```

d. Repeated calculations: Loops

In a computer program a sequence of instructions that are to be repeated is called a loop. The kind of loop we will use in VPython starts with a "while" statement. Instructions inside the loop are indented. Glowscript will indent automatically after you type a colon.

10. As an example of a loop, enter the following commands and run the program.

```
while t < 0.2:
        print (“The time is now “, t)
        t = t + deltat
print (“after the loop”)
```

The statement:

```
t = t + deltat
```

may look like a mathematical error. However, in a program, the "=" sign has a different meaning than it does in a mathematical equation. The right hand side of the statement tells Python to read the old value of **t**, and add the value of **deltat** to it.
The left hand side of the statement tells Python to store this new value into the variable **t**.

Note: Putting print statements in your code is another useful technique for finding logic errors when you are writing programs.

Answer the following questions:
What makes the loop stop?
Why is the first printed time 0?
Why is the last time 0.19 and not 0.2?
How can you get the program to print values from 0 through 0.3? (Try it.)

e. Constant velocity

We now want to model the constant velocity cart iteratively using a loop.

11. Modify the loop in your program to make it look like the following and run your program:

```
while t < 2:
        print("time = ", t, "  position = ", cart.pos, "  velocity = ",velcart)
        cart.pos=cart.pos + velcart*deltat                              #computes new position
        velcart = velcart + accelcart*deltat                            #computes new velocity
        t = t + deltat
print(“The End”)
```

12. Modify the while statement in your program so that it runs just long enough for the cart to travel 2 meters.

Slowing down the animation

When you run the program, you should see the cart at its final point. The program is executed so rapidly that the entire motion occurs faster than we can see, because a "virtual time" in the program elapses much faster than real time does. **We can slow down the animation rate by adding a “rate” statement.**

13. Add the following line inside your loop (indented) and run the program:

```
rate(100)
```

Every time the computer executes the loop, when it reads “**rate(100)**”, it pauses long enough to ensure the loop will take 1/100th of a second. Therefore, the computer will only execute the loop 100 times per second.

You should see the cart travel to the right at a constant velocity, ending up 2 meters from its starting location.

Note: The cart going beyond the edge of the track isn’t a good simulation of what really happens, but it’s what we told the computer to do. There are no “built-in” physical behaviors, like gravitational force, in VPython. Right now, all we’ve done is tell the program to make the cart move in a straight line. If we wanted the cart to fall off the edge, we would have to enter statements into the program to tell the computer how to do this.

Test Your Knowledge

Modify your code so that the cart starts at the right end of the track and moves to the left till it just reaches the end of the track. Make sure that you code works and then have your instructor check your work.

____

Instructor

Out of Lab Project

Write VPython programs that simulate the dropping of a red ball of radius 0.5 m to ground which is located at y=0 under various conditions given below. In your programs, the ground can be modeled as the top of a white box with a size=vector(10,0.5,1).

A. Program 1 called Drag1.py
   - Dropped ball is falling only under the force of gravity so its acceleration is a constant -9.8 m/s^2 in the y-direction.
   - Make an animation showing a dropped ball falling a distance of 10 meters where it just touches the ground.
   - The time step for your iterations is 2 ms
   - Use rate(2000) to slow your animation
   - For each iteration, print to the screen the time, y-position, and velocity in the y-direction of the ball. Print the data in three columns with a few spaces between them. Print only the numerical values and the spaces. Do not print any other text as it will make things harder for your other tasks.

B. Program 2 called Drag2.py
   - Dropped ball is now also experiencing a changing drag force that depends on the objects velocity. The object's acceleration in terms of its velocity is now given by:
     $\vec{a} = -2\vec{v} - 9.8\hat{\jmath}$
   - All other requirements and conditions are the same as for program 1.

C. Once you have a program working, run your program and collect your data on the falling ball.
   In Excel, create the following three tabs:
   1. First tab contains you data (column A is time, Column B is y-position, Column C is the y-velocity)
   2. Velocity vs time graph (smooth curve – no markers)
   3. Position vs time graph (smooth curve – no markers)

   An easy way to get your data into Excel so you can graph it is to use the program Notepad as follows:
   - Copy the output data from your program's output screen and paste it into Notepad
   - Save the data in Notepad as a text file
   - Open a blank spreadsheet in Excel
   - Go to the Data menu and select the "From Text" option
   - Enter the name of your text file
   - Select "Delimited" for your file option and choose your characters (space, commas, etc)

   A delimited file is one that uses special characters to tell when one piece of data ends and another piece of data starts. We are telling Excel that spaces are being used in the file to separate the time, position, and velocity data for the ball so that it puts the data into three different columns. Once you have the data on a sheet in Excel can use it to create your graphs.

D. Send your instructor an email prior to lecture on the day of your next lab with the following:
   1. Heading is the names of the members of your group
   2. Attach your computer program files as email attachments so your instructor can run them
   3. Attach your Excel files containing your data and graphs as email attachments.

4. In the main body of the email, give the final speed that the ball with air drag achieved according to your computer program. Explain how to calculate the terminal velocity of the ball according to theory and show if the theory matches the simulation (we will say that to within $< 0.5\%$ is a match).