

## LAB 02-1: Modeling motion due to a constant net force

In this activity, you will learn how to use a computer to model motion with a constant net force. Specifically, you will model the motion of a fan cart on a track.

Before you begin, write down the following data from your previous experiment analyzing the motion of a fan cart in a video.

mass of cart	0.8 kg
$a_x$	0.19 m/s <sup>2</sup>
$\vec{F}_{\text{net on cart}}$	$\langle 0.15, 0, 0 \rangle$ N

### Writing your program

It is helpful to start with a previous program, as opposed to starting one from scratch.

#### Start with a previous program

1. Open your previous program that modeled the motion of a ball rolling on a track.
2. Save this file with a new name like `fan-cart.py`.
3. Edit this program to only shown a track. Give the track a length of 3.0 m.
4. Run your program to ensure that it works. It should ONLY show the track.
5. Add the line below to create a box that represents the fan cart that has a length of 0.1 m, a width of 0.05 m, and a height of 0.04 m. Give it an initial position so that it starts near the left end of the track.

```
cart = box(pos=vector(-1.4,0,0), size=(0.1,0.04,0.05), color=color.green)
```

6. Define constants and initial conditions: including the mass of the cart, its initial velocity, and its initial momentum. Let's assume that the cart starts from rest.

```
cart.m = 0.8
cart.v = vector(0,0,0)
cart.p = cart.m*cart.v
```

Each of these variables is a property of the object named `cart`. Thus, whenever you want to refer to the momentum of the cart, you must refer to `cart.p`, for example.

#### Define the time step

To make the cart move, we calculate its new position after each time interval  $\Delta t$  using the equation  $\vec{r}_f = \vec{r}_i + \vec{v}\Delta t$ . This equation is an approximation and is only valid for small  $\Delta t$ . So, choose a small time interval.

Since it only takes a few seconds for the cart to travel across the track, a reasonably small time interval would be one hundredth of a second.

7. For now, let's use 1 hundredth of a second as the time interval,  $dt$ . Define a variable  $dt$  for the time interval (i.e. time step).

```
dt=0.01
```

8. Also, let's define the total time  $t$  that represents the clock reading. We will add  $dt$  to  $t$  during each time interval in order to keep track of the total time. The clock starts out at  $t = 0$ .

t=0

That completes the first part of the program which tells the computer to:

- (a) Create the 3D objects.
- (b) Give the cart an initial position and velocity and momentum.
- (c) Define variable names for the clock reading  $t$  and the time interval  $dt$ .

**Create a “while” loop to continuously calculate the position of the object.**

We will now create a `while` loop. Each time the program runs through this loop, it will:

- (a) Calculate the change in the ball’s momentum and use it to calculate its final momentum.
  - (b) Calculate the final velocity.
  - (c) Calculate the change in the ball’s position and use it to calculate its final position.
  - (d) Calculate the new clock reading by adding the time interval to the current clock reading.
  - (e) Repeat.
9. On a new line, begin the while statement as shown below. Note that the condition of the while loop will cause the loop to stop when the cart reaches the right end of the track.

```
while cart.pos.x < 1.5:
```

10. In class, you learned that the final momentum of an object after a short time interval  $\Delta t$  is

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

Inside the loop, define the net force on the cart. Use the value measured in the video analysis experiment.

```
Fnet = vector(0.15,0,0)
```

11. Use the net force to update the momentum of the cart (i.e. calculate the final momentum of the cart).

```
cart.p = cart.p + Fnet*dt
```

12. Use the final momentum to calculate the final velocity of the cart.

```
cart.v = cart.p/cart.m
```

13. For a constant force,

$$\vec{r}_f = \vec{r}_i + \vec{v}_{\text{avg}} \Delta t$$

However, we are going to make the approximation:

$$\vec{v}_{\text{avg}} \approx \vec{v}_f$$

Thus,

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

14. Let's use this equation to calculate the new position of the cart after a time interval  $dt$ .

```
cart.pos = cart.pos + cart.v*dt
```

15. Now let's increment the clock reading  $t$

```
t=t+dt
```

16. Save and run the program.

17. The animation will likely run too fast. If so, use the `rate` statement to slow down the computer. Type the following line as the first line (indented of course) after the `while` statement.

```
rate(100);
```

18. Save and run the program.

Does the cart clearly accelerate? How do you know?

19. Adjust the force to be larger or smaller than the value measured in the lab, run the program, and view the resulting motion.

20. Now, keep the force the same value, and change the mass to larger values.

If the force is kept the same, how does increasing the mass of the cart affect its acceleration?

If the force is kept the same, how does decreasing the mass of the cart affect its acceleration?

## Graphing the cart's x-position and x-velocity as a function of time.

We are going to graph  $x$  vs.  $t$  and  $v_x$  vs.  $t$  and compare the graphs to what we obtained in the experiment.

1. On the third line of the program, after the first two import statements, add an `import` statement to import VPython's graph module which enables you to create graphs.

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

2. *Before the while loop*, type the following lines. Note that each statement should be on one line, though they wrap to multiple lines below.

```
xGraph=gdisplay(title="x vs. t", xtitle='t (s)',ytitle='x (m)', x=400, y=0, width=400, height=200)
```

```
xPlot=gcurve(color=color.cyan)
```

```
vGraph=gdisplay(title="v_x vs. t", xtitle='t (s)',ytitle='v_x (m/s)', x=400, y=0, width=400, height=200)
```

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

The lines above create two new graph windows (called a `gdisplay`). In each graph window, it creates a plot (i.e. a curve) to be graphed.

3. *Inside the while loop* at the end of the loop, add the following lines:

```
xPlot.plot(pos=(t, cart.pos.x))
```

```
vPlot.plot(pos=(t, cart.v.x))
```

These lines add data points to the two different plots (i.e. curves). In the `xPlot`, it adds a datapoint for  $(t,x)$  to the plotted curve. In the `vPlot`, it adds a datapoint for  $(t,v_x)$  to the plotted curve.

4. Run your program. You should see two new graph windows. Arrange them on your screen so that you can simultaneously view the graphs and the motion of the cart.
5. If the animation starts before you can arrange your windows, add the following line *just before the while loop*.

```
scene.mouse.getclick()
```

- 6.

Sketch the  $x$  vs.  $t$  graph below. How does it compare to what you found in the experiment with the fan cart?

- 7.

Sketch the  $v_x$  vs.  $t$  graph below. How does it compare to what you found in the experiment with the fan cart?

## Application

Answer the questions below about motion due to a constant net force.

1. Suppose that the cart starts from rest at the right side of the track, and the net force on the cart is in the  $-x$  direction. Predict what the graph of  $x$  vs.  $t$  and the graph of  $v_x$  vs.  $t$  will look like. Sketch your predictions below.

2. Place the cart on the right end of the track and make the net force on the cart equal to  $\langle -0.15, 0, 0 \rangle$  N. Run your simulation.

Sketch the graphs of  $x$  vs.  $t$  and  $v_x$  vs.  $t$  below. Compare them to what you predicted. Do they match or are they different? If they are different, study the graphs and see if they make sense.

3. In class, I did a demonstration where the initial velocity of the cart was opposite the force on the cart. As a result, the cart slowed down until it stopped and then sped up in the direction of the force. Set up your simulation so that the cart starts on the left end of the ramp at  $x = -1.4$  m and has an initial velocity of  $1.0$  m/s to the right. Set the net force on the cart to be  $\langle -0.15, 0, 0 \rangle$  N. Run your simulation.

Does the simulation match what you observed in the demonstration?

You will notice that the cart goes off the left end of the ramp. Change the condition in the `while` statement so that the simulation stops when the cart reaches the left end of the ramp. Again, run the simulation.

Sketch the graphs of  $x$  vs.  $t$  and  $v_x$  vs.  $t$  below.

At what instant in the simulation is the x-velocity of the cart zero?

At the instant when the x-velocity of the cart is zero, what is the net force on the cart?

Think carefully about your answer to the previous question. The air pushing against the turning fan is what exerts a force on the cart. If the fan turns at a constant rate, then the net force on the cart is constant. Is your answer to the previous question consistent with the fact that the net force on the cart is constant?

At the instant when the x-velocity of the cart is zero, what is the x-acceleration of the cart?

Acceleration is a result of the net force on the object. According to the Momentum Principle (Newton's second law), acceleration and net force are proportional. Are your answers to the previous two questions consistent with the fact that net force on the cart and the acceleration of the cart are proportional?

4. Now, with the cart starting at  $x = -1.4$  m and the net force on the cart equal to  $\langle -0.15, 0, 0 \rangle$  N, determine the velocity of the cart needed so that the turning point (the instant when the cart's x-velocity is zero) occurs at exactly (or at least very close to)  $x = 1.4$  m. You can do this by trial and error. It will be convenient to print the x-position and x-velocity of the cart by typing the following line at the end of the while loop:

```
print t, cart.pos.x, cart.v.x
```

This prints the clock reading, the x-position of the cart, and the x-velocity of the cart, separated by commas.

What is the initial velocity of the cart needed so that the cart's turning point is at  $x = +1.4$  m? What is the clock reading when the cart is at the turning point?

Now, apply theory to answer the previous question. Use the equations given to you in class. Compare the results from theory to the results from your computer model. How do they compare?