Fall 2021

APPLIED PHYSICS - LAB

Lab-7

- **Collisions**
- **Submission task**

[VPython documentation]
[VPython vector operations]
[Vpython basic mathematics]

Collisions:

In the last lab we learnt that for simulating an accelerated object we just need to update velocity along with the positions in the while loop. We also learnt how to control simulation execution and termination. For the second part; termination over a condition, we were actually detecting a collision, i.e. we were observing when the two objects are close enough to say that the collision has occurred. In this lab we will define collision on its own rather than as a while loop condition.

We will simulate a simple experiment where a ball is bouncing on a horizontal plane. First, we need space. This time the space is very unique. We have a plane (made using the box function) placed at -5 units below the origin and the ball (using sphere function) at 7 units above the origin.

```
1  GlowScript 3.1 VPython
2  from vpython import *
3
4  ground = box(pos=vector(0,-5,0),size=vector(10,0.2,10))
5  ball = sphere(pos=vector(0,5,0),radius=0.8)
6
```

Now, we let the ball move in free fall for some time.

```
6
7  v=vector(0,0,0)
8  a=vector(0,-9.8,0)
9  t,dt = 0,0.01
10
11 while t<2:
12   rate(30)
13   ball.pos = ball.pos + v*dt
14   v = v + a*dt
15   t = t + dt
16
```

You will notice that the ball goes right through the plane. Let us now create a collision. This means the following:

- For every updated value of the position we check the distance between the surface and the ball.
- The distance (dist_y) must not reduce from the sum of ball.radius and the height of the plane.
- We will employ `if` condition inside the loop.
- If the distance decreases from the above mentioned then we flip the direction of the velocity.

We can add the if condition inside the while loop;

```
13    ball.pos = ball.pos + v*dt
14    v = v + a*dt
15
16    # checking for collision in y
17    dist_y = abs(ground.pos.y - ball.pos.y)
18    if dist_y < (ball.radius + ground.size.y/2):
19      v.y = -v.y
20
21    t = t + dt
```

of course you can adjust the rate and time condition to run simulation for proper visualization.

Task ↦ What if we give this ball a horizontal velocity as well…? Try to simulate this.

But we could also have a collision with a wall as well. For this, first, let us create a wall;

```
wall = box(pos=vector(5,0,0),size=vector(0.2,10,10))
```

Remember that you now need to create a second variable, dist_x and use another if condition to reverse the horizontal part of the velocity.

```
# checking for collision in x
dist_x = _____________________
if _____________________________:
  ___________
```

Submission Task:

After this, start simulation with `v = vector(7,3,0)` and answer which of the following shapes will be followed after the collision with the wall (green curve in the figure below). Your submission will be the program that supports your answer.