Lab-IV

Devansh Shukla
I18PH021

1 Aim

Write and execute an octave program to simulate/solve motion of a particle in a plane in cartesian coordinates.

2 Theory

3 Program

3.1 Simple pendulum

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% MotionInAPlaneCart
%
% Program to solve/simulate motion of a particle in uniform circular motion
% in cartesian coordinates.
% Author: Devansh Shukla I18PH021
% 9th March, 2022
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

graphics_toolkit gnuplot
pkg load symbolic

% set the symbolic variables
syms x0 y0 t t0 R omega

% Equations for positions ‘x‘ and ‘y‘
x = x0 + R*cos(omega * (t-t0));
y = y0 + R*sin(omega * (t-t0));

% Equations for velocities
vx = diff(x, t);
vy = diff(y, t);

% Equations for accelerations
ax = diff(vx, t);
ay = diff(vy, t);


% for pretty output
fprintf("Position\n");
fprintf("x = "); pretty(x)
fprintf("y = "); pretty(y)
fprintf("\n")

fprintf("Velocity\n");
fprintf("vx = "); pretty(vx)
fprintf("vy = "); pretty(vy)
fprintf("\n")

fprintf("Acceleration\n");
fprintf("ax = "); pretty(ax)
fprintf("ay = "); pretty(ay)
fprintf("\n")

% Initial conditions
omega = input("Enter omega ");
R = input("Enter Radius ");
x0 = y0 = t0 = 0.0;

pos_x = pos_y = [];
vel_x = vel_y = [];
acc_x = acc_y = [];

% loop for numerically computing position, velocity and acceleration
idx = 1;
for t=0:0.1:5
   pos_x(idx) = eval(x);
   vel_x(idx) = eval(vx);
   acc_x(idx) = eval(ax);

   pos_y(idx) = eval(y);
   vel_y(idx) = eval(vy);
   acc_y(idx) = eval(ay);
   idx = idx + 1;
endfor

% plot position, velocity and acceleration
figure();
hold on;
grid on;
set(gcf, 'PaperSize', [6, 3]);
set(gca,'XMinorTick','on','YMinorTick','on')
```

```
plot(pos_x, pos_y, "linewidth", 2);
xlabel("x[m]");
ylabel("y[m]");
title("Trajectory");
xlim([-3, 3])
ylim([-3, 3])
set(gcf, 'renderer', 'painters');
print("-dpng", "motion_cart_traj.png");
hold off;

figure();
hold on;
grid on;
set(gcf, 'PaperSize', [6, 3]);
set(gca,'XMinorTick','on','YMinorTick','on')
plot(pos_x, "linewidth", 2);
plot(pos_y, "linewidth", 2);
xlabel("Time[s]");
ylabel("Position[m]");
title("Position");
ylim([-5, 5])
legend("x", "y")
legend boxoff
set(gcf, 'renderer', 'painters');
print("-dpng", "motion_cart_pos.png");
hold off;

figure();
hold on;
grid on;
set(gcf, 'PaperSize', [6, 3]);
set(gca,'XMinorTick','on','YMinorTick','on')
plot(vel_x, "linewidth", 2);
plot(vel_y, "linewidth", 2);
xlabel("Time[s]");
ylabel("Velocity[m/s]");
title("Velocity");
ylim([-5, 5])
legend("v_x", "v_y")
legend boxoff
set(gcf, 'renderer', 'painters');
print("-dpng", "motion_cart_vel.png");
hold off;

figure();
hold on;
grid on;
set(gcf, 'PaperSize', [6, 3]);
set(gca,'XMinorTick','on','YMinorTick','on')
plot(acc_x, "linewidth", 2);
plot(acc_y, "linewidth", 2);
xlabel("Time[s]");
ylabel("Acceleration[m/s/s]");
title("Acceleration");
legend("a_x", "a_y")
legend boxoff
set(gcf, 'renderer', 'painters');
print("-dpng", "motion_cart_acc.png");
hold off;
```

4 Results

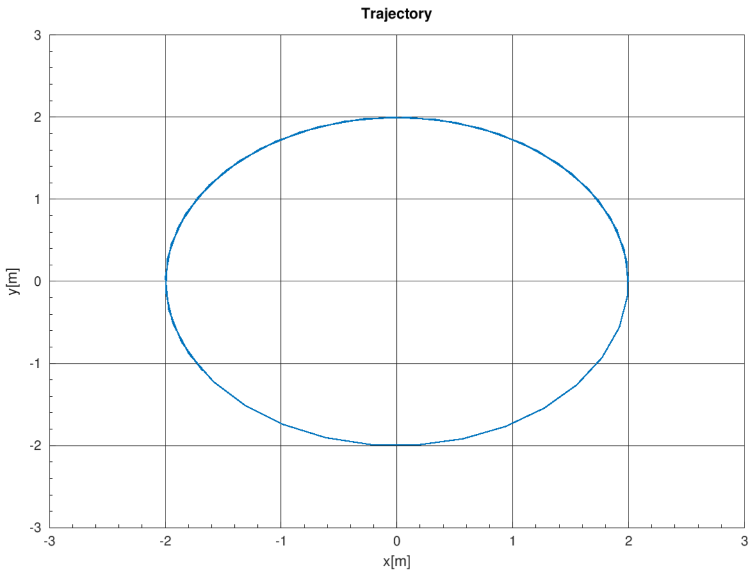
4.1 Terminal output

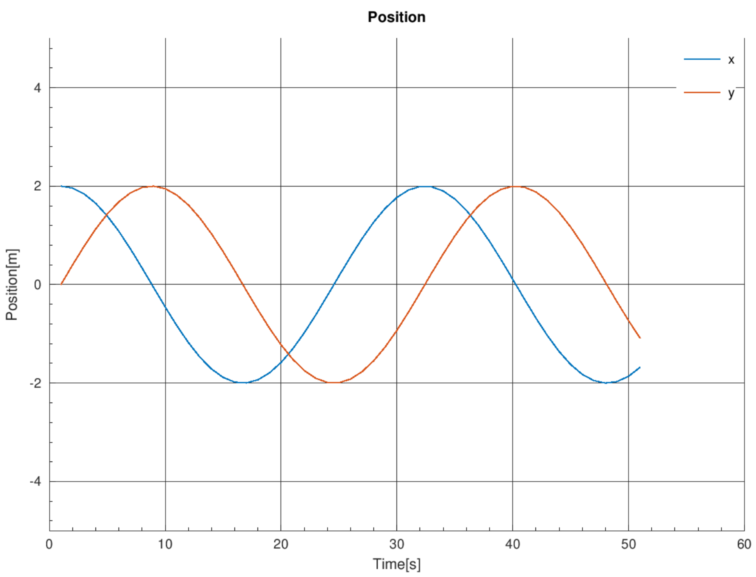
```
(escape) devansh@ds:~/GitHub/Vault/OctaveLab/Programs/outputs$ octave ../MotionOnAPlaneCart.m
Symbolic pkg v2.9.0: Python communication link active, SymPy v1.5.1.
Position
x =   R⋅cos(ω⋅(t - t₀)) + x₀
y =   R⋅sin(ω⋅(t - t₀)) + y₀

Velocity
vx =   -R⋅ω⋅sin(ω⋅(t - t₀))
vy =   R⋅ω⋅cos(ω⋅(t - t₀))

Acceleration
ax =       2
  -R⋅ω ⋅cos(ω⋅(t - t₀))
ay =       2
  -R⋅ω ⋅sin(ω⋅(t - t₀))

Enter omega 2
Enter Radius 2
```

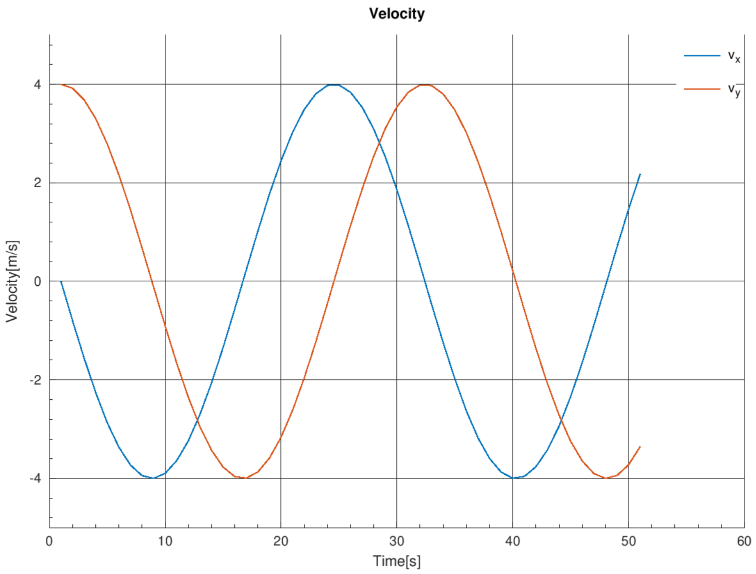
4.2 Plots

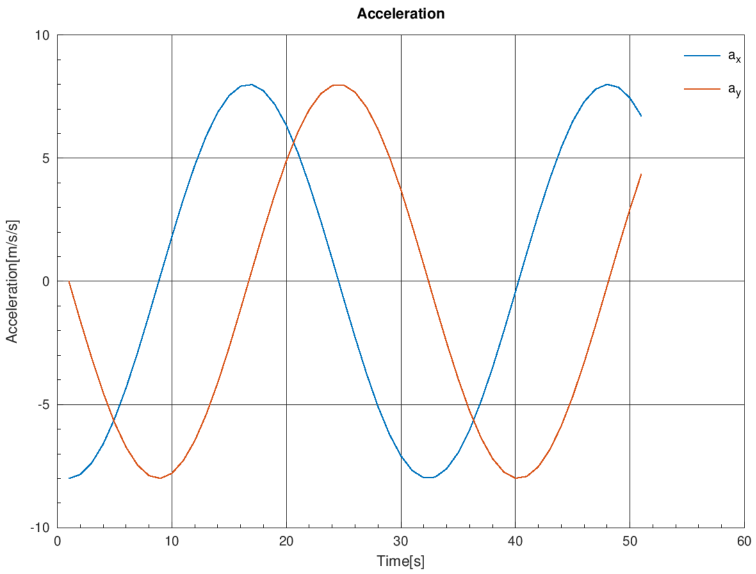
Initial parameters

- $R = 2.0\ m$
- $\omega = 2.0\ rad/s$
- $x_0 = y_0 = 0.0m$









5 Remarks

The programs can be used to trace and simulate the motion of any particle in a plane by defining the required parameters.