Table of Contents

Table of Figures

Chapter 1: Introduction

Fourteen-point masses lying on a flexible cord are connected through similar springs and dampers. The system is show in Figure 1. Except for the end-points all points fall under gravity while the springs and dampers provide a restoring force. The Free-Body Diagram (FDG) of a single point mass is shown in Figure 2.

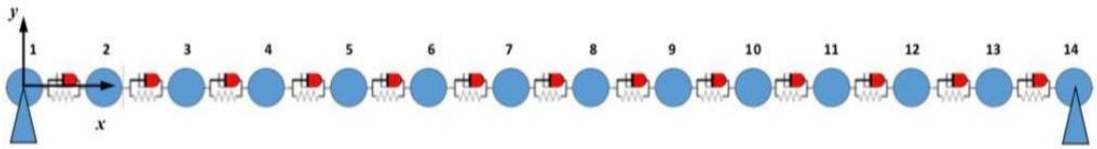

Figure 1 Mass-Spring-Damper (MSD) representation of problem

The purpose of this project is solving for each point mass its instantaneous position and velocity at 10 different times from rest time till the steady-state conditions. These tasks have been repeated for a variety of initial conditions. As per conditions of the project the solutions were found with the help of *Wolfram Mathematica.*

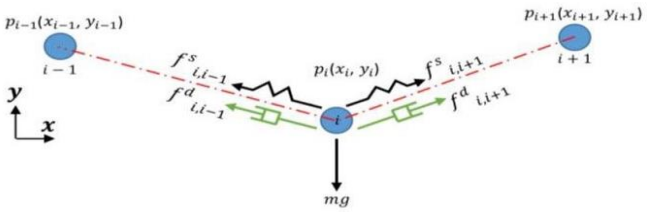

Figure 2 Free-Body Diagram (FDG) of point mass

The Figure 3 graphic provides an overview for the entire project. The report is structured to follow this graphic. Therefore Chapter 1 deals with the mathematical modelling of the problem. Chapter 2 discusses programming in Mathematica and Chapter 3 contains the displacement and velocity data as graphs.

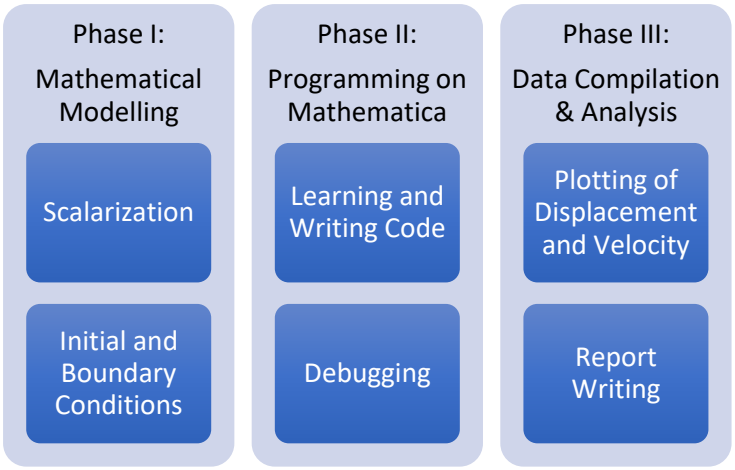

Figure 3 Structure Chart of Project

Chapter 2: Mathematical Modelling

The free body diagram was given in the problem statement along with the formula for the forces acting on the point mass.

$$\vec{f}^{s}_{i,i-1} = -k_s\left(\left\|\vec{p}_i - \vec{p}_{i-1}\right\| - l_r\right)\frac{\left(\vec{p}_i - \vec{p}_{i-1}\right)}{\left\|\vec{p}_i - \vec{p}_{i-1}\right\|} \tag{1}$$

$$\vec{f}^{s}_{i,i+1} = -k_s\left(\left\|\vec{p}_i - \vec{p}_{i+1}\right\| - l_r\right)\frac{\left(\vec{p}_i - \vec{p}_{i+1}\right)}{\left\|\vec{p}_i - \vec{p}_{i+1}\right\|} \tag{2}$$

$$\vec{f}^{d}_{i,i-1} = -k_d\left(\vec{u}_i - \vec{u}_{i-1}\right) \tag{3}$$

$$\vec{f}^{d}_{i,i+1} = -k_d\left(\vec{u}_i - \vec{u}_{i+1}\right) \tag{4}$$

where $\vec{f}^{s}_{i,i-1}$, $\vec{f}^{d}_{i,i-1}$, $\vec{f}^{s}_{i,i+1}$, and $\vec{f}^{d}_{i,i+1}$ are the spring and damper forces acting on point-mass i by its neighboring point-masses, and $\vec{p}_i$, $\vec{p}_{i-1}$, and $\vec{p}_{i+1}$ are the position vectors of point-masses i, $i-1$, and $i+1$, respectively. k_s and k_d are the spring and damping constants, respectively. The un-stretched length of the springs is shown with l_r. The velocity vectors for point-masses i, $i-1$, and $i+1$, are shown with $\vec{u}_i$, $\vec{u}_{i-1}$, and $\vec{u}_{i+1}$, respectively. The instantaneous distance between neighboring point-masses are

$$\left\|\vec{p}_i - \vec{p}_{i-1}\right\| = \sqrt{\left(x_i - x_{i-1}\right)^2 + \left(y_i - y_{i-1}\right)^2} \tag{5}$$

$$\left\|\vec{p}_i - \vec{p}_{i+1}\right\| = \sqrt{\left(x_i - x_{i+1}\right)^2 + \left(y_i - y_{i+1}\right)^2} \tag{6}$$

Therefore, the fractions on the right-hand side of Equations 1 and 2 are the unit vectors for the distance between the corresponding point-masses. The position and velocity of the point-mass can be obtained by solving Newton's 2nd law written for each point-mass:

$$\begin{aligned} m\vec{a}_i = &-k_s\left(\left\|\vec{p}_i - \vec{p}_{i-1}\right\| - l_r\right)\frac{\left(\vec{p}_i - \vec{p}_{i-1}\right)}{\left\|\vec{p}_i - \vec{p}_{i-1}\right\|} - k_s\left(\left\|\vec{p}_i - \vec{p}_{i+1}\right\| - l_r\right)\frac{\left(\vec{p}_i - \vec{p}_{i+1}\right)}{\left\|\vec{p}_i - \vec{p}_{i+1}\right\|} \\ &-k_d\left(\vec{u}_i - \vec{u}_{i-1}\right) - k_d\left(\vec{u}_i - \vec{u}_{i+1}\right) + m\vec{g} \end{aligned} \tag{7}$$

where $\vec{g}$ is the gravitational acceleration, $\vec{a}_i$ is the acceleration of the point-mass i, and m is the mass of each point-mass.

Equation (7) is a second order ordinary differential equations(ODE) of position. By solving this equation for a set of initial and boundary conditions we will find the solution to the tasks.

2-1 Scalarization

The given formulae are in vector notation. For ease of calculation they were reduced into their x- and y- components. Other reasons for scalarizing is that we see the system is symmetrical about the axis passing through the midpoint of the cord. Therefore, the x-forces about the mid-axis will

balance each other (though they would have some gross value). Therefore, our boundary conditions can be simplified as the x-coordinate of the point masses remain the same throughout the motion of the system. Equations (8) and (9) are now the system of ODEs when solved will give the solutions.

$$
\begin{aligned}
&m * x[i]'' \\
&= -\frac{k_s(-x[-1+i][t]+x[i][t])\left(-l_r+\sqrt{(-x[-1+i][t]+x[i][t])^2+(-y[-1+i][t]+y[i][t])^2}\right)}{\sqrt{(-x[-1+i][t]+x[i][t])^2+(-y[-1+i][t]+y[i][t])^2}} \\
&-\frac{k_s(-x[1+i][t]+x[i][t])\left(-l_r+\sqrt{(-x[1+i][t]+x[i][t])^2+(-y[1+i][t]+y[i][t])^2}\right)}{\sqrt{(-x[1+i][t]+x[i][t])^2+(-y[1+i][t]+y[i][t])^2}} \\
&-k_d(-x[-1+i]'[t]+x[i]'[t])-k_d(x[i]'[t]-x[1+i]'[t])
\end{aligned}
\tag{8}
$$

$$
\begin{aligned}
&m * y[i]'' \\
&= -\frac{k_s(-x[-1+i][t]+x[i][t])\left(-l_r+\sqrt{(-x[-1+i][t]+x[i][t])^2+(-y[-1+i][t]+y[i][t])^2}\right)}{\sqrt{(-x[-1+i][t]+x[i][t])^2+(-y[-1+i][t]+y[i][t])^2}} \\
&-\frac{k_s(-x[1+i][t]+x[i][t])\left(-l_r+\sqrt{(-x[1+i][t]+x[i][t])^2+(-y[1+i][t]+y[i][t])^2}\right)}{\sqrt{(-x[1+i][t]+x[i][t])^2+(-y[1+i][t]+y[i][t])^2}} \\
&-k_d(-y[-1+i]'[t]+y[i]'[t])-k_d(y[i]'[t]-y[1+i]'[t])
\end{aligned}
\tag{9}
$$

2-2 Initial Conditions

We are given three different set of initial conditions to model and solve for the system. The values of k_s/m, k_d/m, lr and m remain constant. m is arbitrarily set as 1 kg, we have $k_s = 100\ N/m$, $k_d = 10\ N.s/m$ and $lr = 1\ m$. Also, for all initial conditions, zero stretching in the cord is assumed.

2-2-1 Initial Conditions I

In the first task, the point masses begin from their rest position with zero initial velocity. They are simply modelled as:

$$
y[i][t] = 0\ , y[i]'[t] = 0, x[i][0] = \mathrm{lr} * (i-1), x[i]'[0] = 0 \quad for\ all\ i
\tag{10}
$$

2-2-2 Initial Conditions II

Now, the point masses have an upward parabolic initial velocity with 25 m/s^2 as the peak value.

The equation of the parabola with endpoints 0, 13 and peak value 25 was found to be:

$$y = -0.588694638695\,x^2 + 7.67266899767x \qquad (11)$$

Inserting the value of the different points gives us the initial conditions:

$$\{y[1]'[0] = 0,$$
$$y[2]'[0] = 7.083974359, y[3]'[0] = 12.99055944,$$
$$y[4]'[0] = 17.71975524, y[5]'[0] = 21.27156177,$$
$$y[6]'[0] = 23.64597902, y[7]'[0] = 24.84300699,$$
$$y[8]'[0] = 24.84300699, y[9]'[0] = 23.64597902,$$
$$y[10]'[0] = 21.27156177, y[11]'[0] = 17.71975524,$$
$$y[12]'[0] = 12.99055944, y[13]'[0] = 7.083974359,$$
$$y[14]'[0] = 0\}$$
$$y[i][0] == 0, x[i][0] = \mathrm{lr} * (i-1), x[i]'[0] = 0 \qquad for\ all\ i \qquad (12)$$

2-2-3 Initial Conditions III

In the last task, the cord is pulled from the middle to a height of 0.25 m. The two halves of the cord still show symmetry about the mid-axis. They make lines with equal but opposite slopes. One of those line equations are found and x is interchanged to find the initial y positions.

$$y = \frac{1}{26}x \qquad (13)$$

$$\left\{\begin{array}{c} y[1][0] = 0, \\ y[2][0] = 0.038462, y[3][0] = 0.076923, \\ y[4][0] = 0.115385, y[5][0] = 0.153846, \\ y[6][0] = 0.192308, y[7][0] = .230769, \\ y[8][0] == 0.230769, y[9][0] = 0.192308, \\ y[10][0] = 0.153846, y[11][0] == 0.115385, \\ y[12][0] = 0.076923, y[13][0] = 0.038462, \\ y[14][0] = 0 \end{array}\right\},$$

$$y[i]'[0] = 0, x[i][0] = \mathrm{lr} * (i-1),\ x[i]'[0] = 0 \qquad for\ all\ i \qquad (14)$$

2-3 Boundary Conditions

For all tasks, the endpoints do not change their position. This is due to the reaction force at the joints. Therefore, we can say that they experience no acceleration.

$$x[14]"[t] = 0, x[1]"[t] = 0, y[14]"[t] = 0, y[1]"[t] = 0 \qquad (15)$$

Chapter 3: Programming

In this chapter, we go over step-by-step the code used to solve the tasks.

3-1 Constant Declaration

All known constants are declared. These include number of time steps, Last point mass, gravitational acceleration, mass of a point mass, spring constant and damping constant.

$$\mathrm{t0} = 9; P = 14; \mathrm{lr} = 1; g = 9.8; m = 1; \mathrm{ks} = 100; \mathrm{kd} = 10; \quad (16)$$

3-2 System of differential equations

Our system of differential equations where entered in the matrix labelled “eqns”. The code is available in Appendix A.

3-3 Initial and Boundary Conditions:

Our modelled initial and boundary conditions discussed in the previous chapter are entered matrices “ic1”, “ic2”, “ic3” and “bc”. The code is available in Appendix A.

3-4 Solving using NDSolveValue

Mathematica has built-in functions for differential equations. “NDSolveValue” can solve for the known point-masses (1 till 14) and times (t0 till t9). The general input of NDSolveValue is of the form:

NDSolveValue[eqns,expr,{x,xmin,xmax},{y,ymin,ymax}]

Where, eqns are the differential equations to be solved by Mathematica, In matric form, include the initial and boundary conditions

expr is the expression the system has to solve.

{x,xmin,xmax},{y,ymin,ymax} represent the variable and their exterme values.

The displacement is solved as:

```
Y = NDSolveValue[{eqns, ic3, bc},  Table[y[i][t], {i, 1, P}], {t, 0, t0}];
 X = NDSolveValue [{eqns, ic3, bc}, Table[x[i][t], {i, 1, P}], {t, 0, t0}];
```

3-5 Plotting of X-Displacement and Y-Displacement with time

Using the following code:

```
{Plot[Y,{t,0,t0},PlotRange->All,AxesLabel->{"Time(s)","y-displacement (m)"}],
Plot[X,{t,0,t0},PlotRange->All,AxesLabel->{"Time(s)","x-displacement (m)"}]}
```

We plot the displacement versus time graphs.

3-6 X-Y Plane Plot

Now we will superimpose the x and y displacement. The following code will give the position of the point masses at 10 different times.

```
Table[Graphics[{PointSize[Medium], Red, Point[NDSolveValue[{eqns, ic3, bc},Table[{x[i][tn],
y[i][tn]}, {i, 1, P}], {t, 0, tn}]]},
  Frame -> True, FrameLabel -> {"Point Mass", "y-Displacement  (m)"},
  PlotLabel -> tn, PlotRange -> {{-.5, 13.5}, {-10.5, 10.5}}], {tn, 0,   t0, 1}]
```

3-7 Velocity Plot

Finally, We plot the velocity using:

```
VY = NDSolveValue[{eqns, ic, bc},   Table[y[i]'[t], {i, 1, P}], {t, 0, t0}];
VX =  NDSolveValue[{eqns, ic, bc}, Table[x[i]'[t], {i, 1, P}], {t, 0, t0}];
{Plot[VY, {t, 0, t0}, PlotRange -> All, AxesLabel -> {"t", "v"}],
 Plot[VX, {t, 0, t0}, PlotRange -> All, AxesLabel -> {"t", "u"}]}
```

Chapter 4: Data Compilation & Analysis

In this chapter, we will publish the results of the program for all initial conditions.

4-1 Initial Conditions I

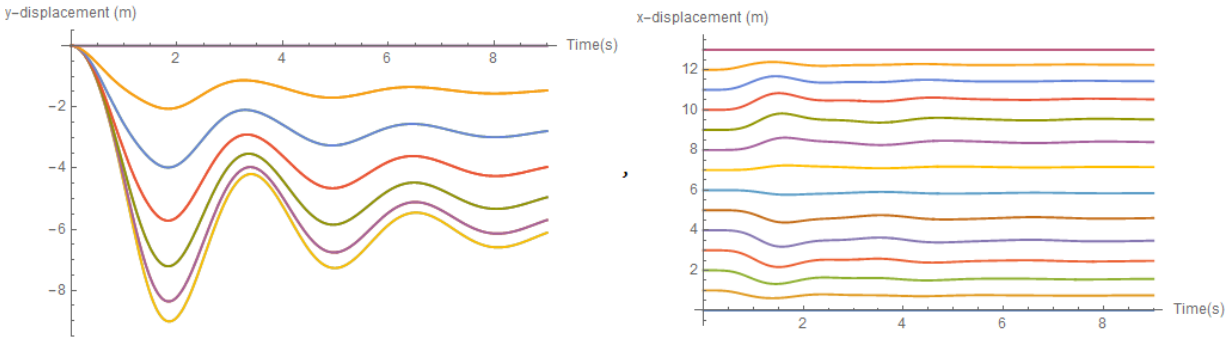

Figure 4 (a) Y-displacement vs Time [I.C. 1] (b) X-displacement vs Time [I.C. 1]

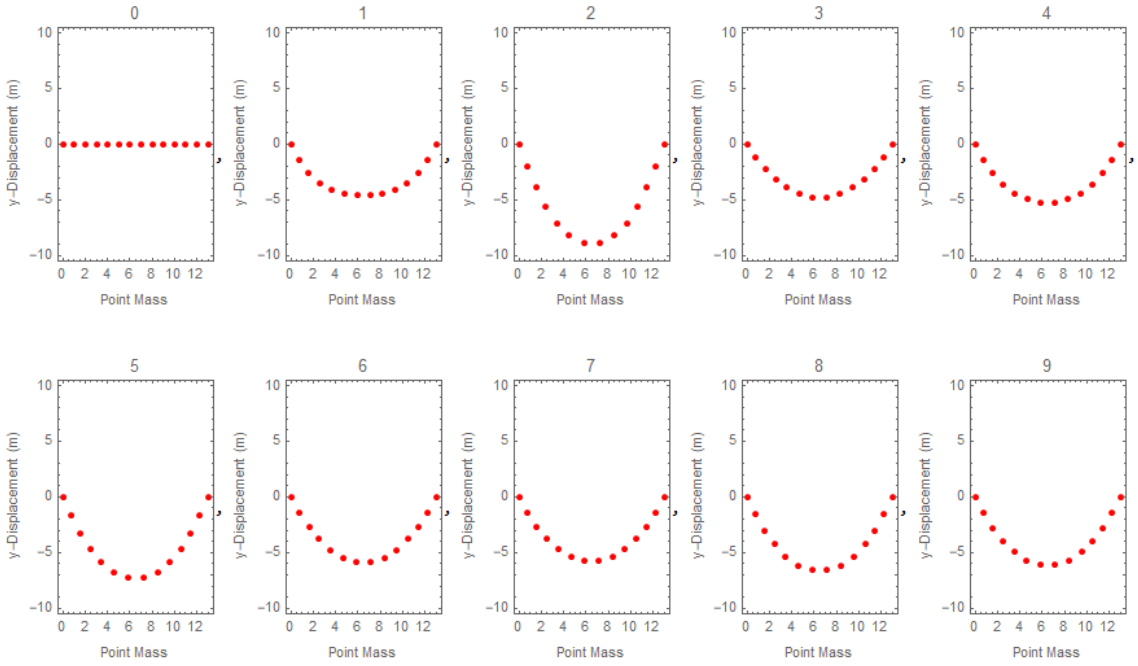

Figure 5 Cord Positions [I.C. 1]

From figure 4 we see that the cord experiences a damping sinusoidal motion. From figure 5, we see that position of the cord is nearly stabilized. Though not completely at rest as wee see from figure 6, we can approximate it to a steady state position as theoretically an infinite time is required for true steady state. Also, from figure the symmetric shape about x-axis indicates that the x-forces cancel each other.

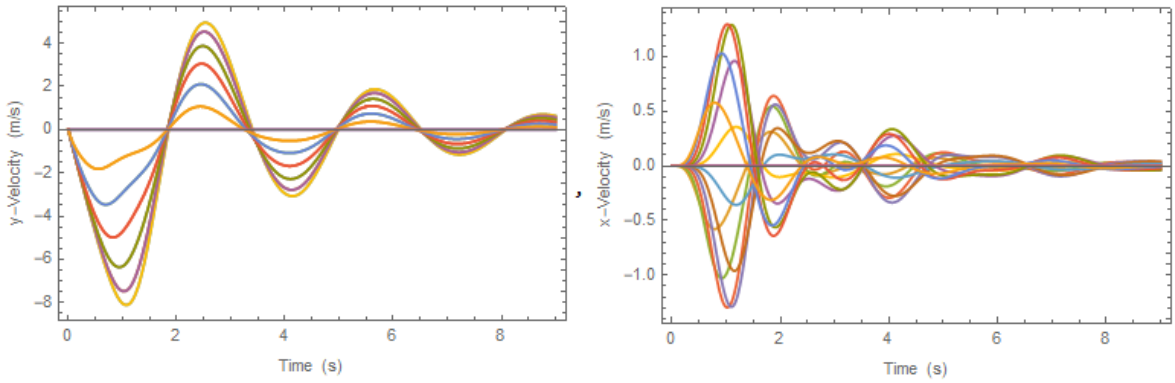

Figure 6(a) Y-Velocity vs Time [I.C. 1] (b) X-Velocity vs Time [I.C. 1]

4-2 Initial Conditions II

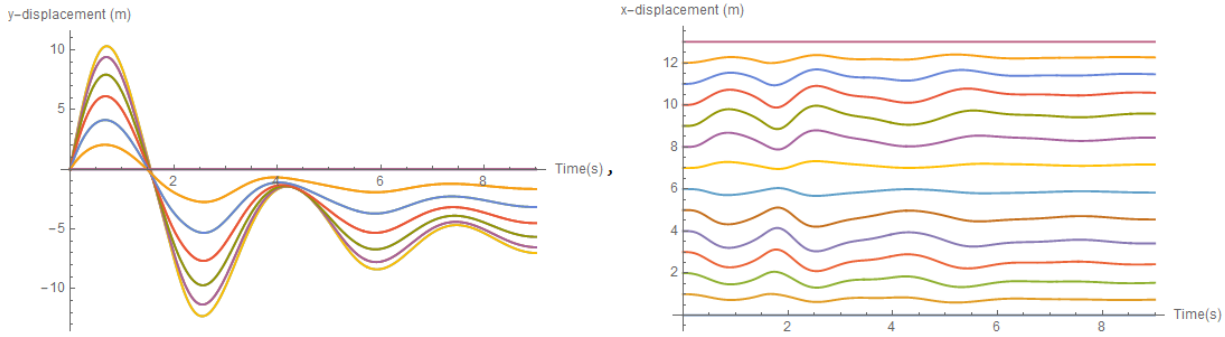

Figure 7 (a) Y-displacement vs Time [I.C. 2] (b) X-displacement vs Time [I.C. 2]

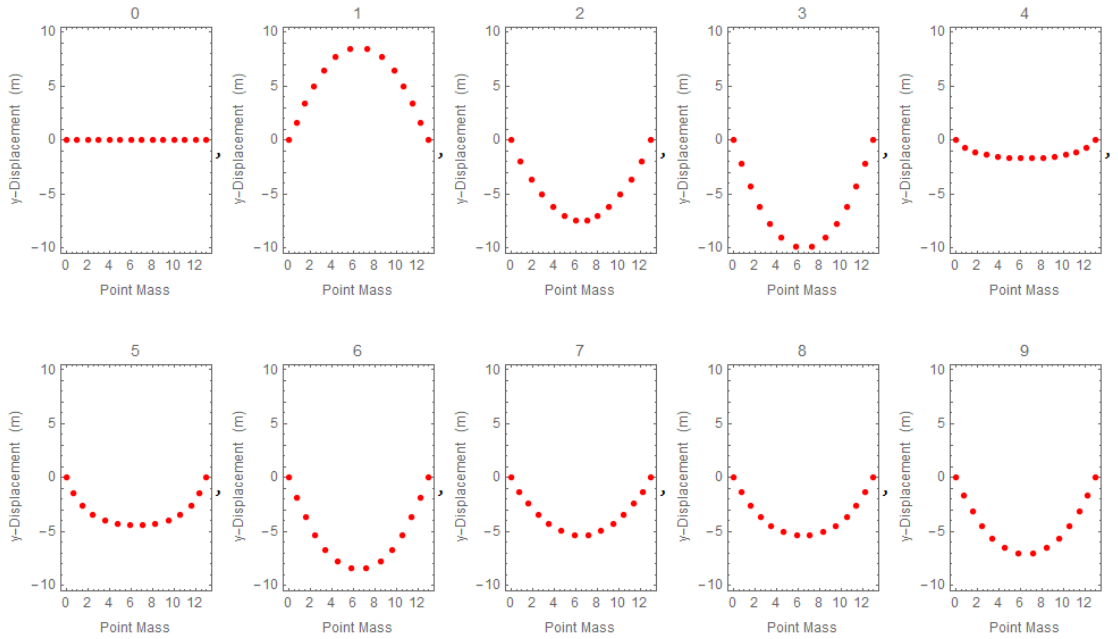

Figure 8 Cord Positions [I.C. 2]

From figure 7 and 8, we see that the initial upward velocity causes an upward displacement of nearly 10 meters. If not for damping a peak of 25 meters would have been reached by the cord. After the initial upward movement, gravity causes the masses to fall and repeated the previously observed damping sinusoidal movement.

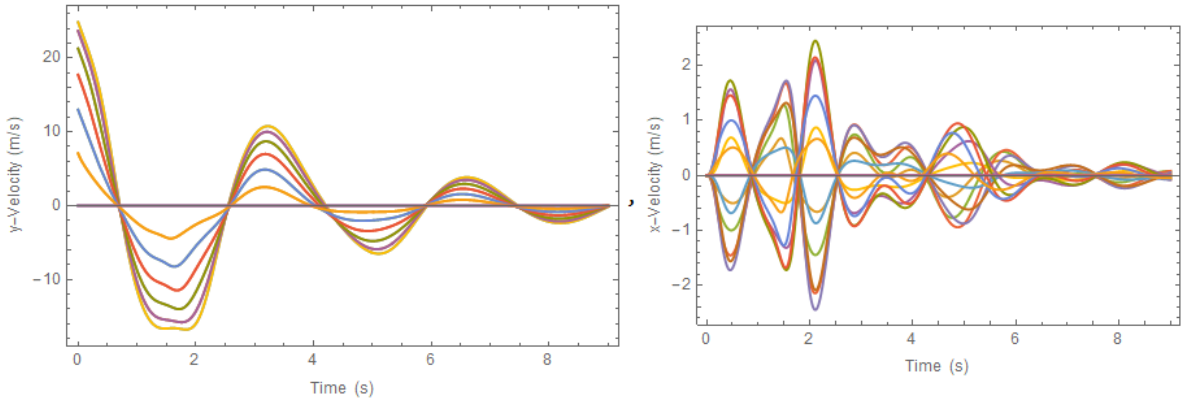

Figure 9 Y-Velocity vs Time [I.C. 2] (b) X-Velocity vs Time [I.C. 2]

4-3 Initial Conditions III

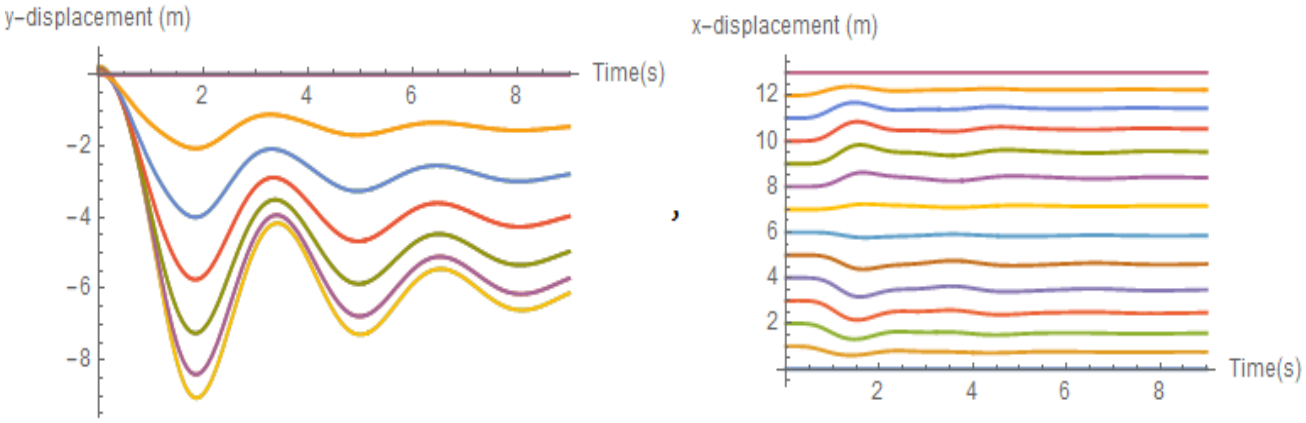

Figure 10 (a) Y-displacement vs Time [I.C. 3] (b) X-displacement vs Time [I.C. 3]

The initial displacement of 0.25 m is nominal compared to displacement caused by gravity. Therefore, the displacement and velocity follow nearly the same pattern as seen in 4-1. The initial position in figure 11 however is a clear indicator of the initial displacement.

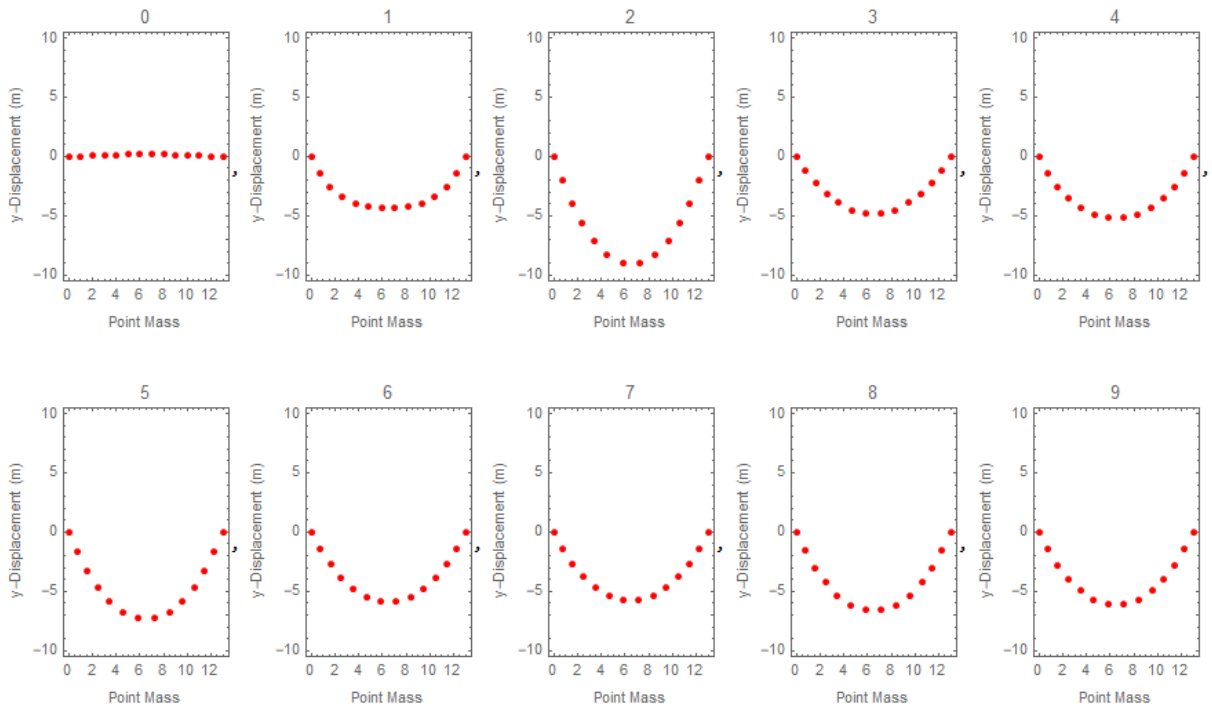

Figure 11 Cord Positions [I.C. 3]

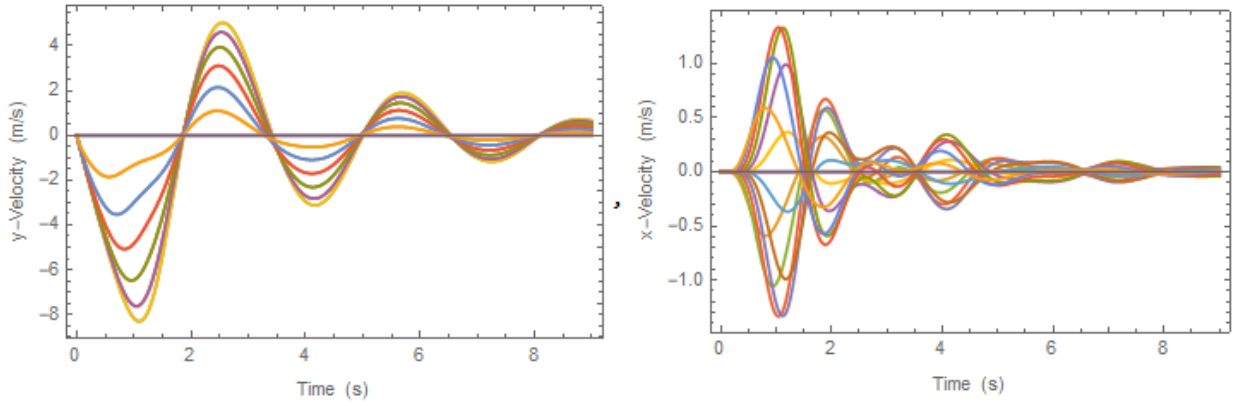

Figure 12 Y-Velocity vs Time [I.C. 3] (b) X-Velocity vs Time [I.C. 3]

4-4 Conclusion

The program gives an output that agrees with the expected theoretical output. It has adapted well to the different initial conditions inputted to it. Therefore, we term the project as a success.

Appendix: Mathematica Code

```
t0=9;P=14;lr=1;g=9.8;m=1;ks=100;kd=10;

eqns={Table[m*x[i]''[t]==-ks*(Sqrt[(x[i][t]-x[i-1][t])^2+(y[i][t]-y[i-1][t])^2]-lr)*(x[i][t]-x[i-
1][t])/Sqrt[(x[i][t]-x[i-1][t])^2+(y[i][t]-y[i-1][t])^2]-ks*(Sqrt[(x[i][t]-x[i+1][t])^2+(y[i][t]-
y[i+1][t])^2]-lr)*(x[i][t]-x[i+1][t])/Sqrt[(x[i][t]-x[i+1][t])^2+(y[i][t]-y[i+1][t])^2]-kd*(x[i]'[t]-
x[i-1]'[t])-kd*(x[i]'[t]-x[i+1]'[t]),{i,2,P-1}],Table[m*y[i]''[t]==-ks*(Sqrt[(x[i][t]-x[i-
1][t])^2+(y[i][t]-y[i-1][t])^2]-lr)*(y[i][t]-y[i-1][t])/Sqrt[(x[i][t]-x[i-1][t])^2+(y[i][t]-y[i-
1][t])^2]-ks*(Sqrt[(x[i][t]-x[i+1][t])^2+(y[i][t]-y[i+1][t])^2]-lr)*(y[i][t]-
y[i+1][t])/Sqrt[(x[i][t]-x[i+1][t])^2+(y[i][t]-y[i+1][t])^2]-kd*(y[i]'[t]-y[i-1]'[t])-kd*(y[i]'[t]-
y[i+1]'[t])-m*g,{i,2,P-1}]};

ic1={Table[y[i]'[0]==0,{i,1,P}],Table[y[i][0]==0,{i,1,P}],Table[x[i][0]==lr*(i-
1),{i,1,P}],Table[x[i]'[0]==0,{i,1,P}]};

ic2={{y[1]'[0]==0,y[2]'[0]==7.083974359,y[3]'[0]==12.99055944,y[4]'[0]==17.71975524,y
[5]'[0]==21.27156177,y[6]'[0]==23.64597902,y[7]'[0]==24.84300699,
y[8]'[0]==24.84300699,y[9]'[0]==23.64597902,y[10]'[0]==21.27156177,y[11]'[0]==17.71
975524,y[12]'[0]==12.99055944,y[13]'[0]==7.083974359,y[14]'[0]==0},Table[y[i][0]==0,{i
,1,P}],Table[x[i][0]==lr*(i-1),{i,1,P}],Table[x[i]'[0]==0,{i,1,P}]};

ic3={Table[y[i]'[0]==0,{i,1,P}],{y[1][0]==0,y[2][0]==0.0384620,y[3][0]==0.0769230,y[4][0]
==0.1153850,y[5][0]==0.1538460,y[6][0]==0.1923080,y[7][0]==.230769,
y[8][0]==0.230769,y[9][0]==0.1923080,y[10][0]==0.1538460,y[11][0]==0.1153850,y[12][
0]==0.0769230,y[13][0]==0.0384620,y[14][0]==0},Table[x[i][0]==lr*(i-
1),{i,1,P}],Table[x[i]'[0]==0,{i,1,P}]};

bc={x[P]''[t]==0,x[1]''[t]==0,y[P]''[t]==0,y[1]''[t]==0};

Y=NDSolveValue[{eqns,ic3,bc},Table[y[i][t],{i,1,P}],{t,0,t0}];X=NDSolveValue[{eqns,ic3,bc
},Table[x[i][t],{i,1,P}],{t,0,t0}];
{Plot[Y,{t,0,t0},PlotRange->All,AxesLabel->{"Time(s)","y-displacement
(m)"}],Plot[X,{t,0,t0},PlotRange->All,AxesLabel->{"Time(s)","x-displacement (m)"}]}


Table[Graphics[{PointSize[Medium], Red, Point[NDSolveValue[{eqns, ic2, bc}, Table[{x[i][tn], y[i][tn]}, {i,
→ True, FrameLabel → {"Point Mass", "y − Displacement (m)"}, PlotLabel → tn, PlotRange
→ {{−.5, 13.5}, {−10.5,10.5}}], {tn, 0, t0,1}]
```

```
VY=NDSolveValue[{eqns,ic2,bc},Table[y[i]'[t],{i,1,P}],{t,0,t0}];
VX=NDSolveValue[{eqns,ic2,bc},Table[x[i]'[t],{i,1,P}],{t,0,t0}];

{Plot[VY,{t,0,t0},Frame->True,FrameLabel->{"Time  (s)","y-Velocity  (m/s)"},PlotRange->All],

Plot[VX,{t,0,t0},PlotRange->All,Frame->True,FrameLabel->{"Time  (s)","x-Velocity  (m/s)"}]}
```