Term Project:

Evolving Soft Robots (Phase B)

Submitted by: Yaoyao Xu (UNI: yx2725)

Ruoyao Zhao (UNI: rz2535)

Course number: MECSE4510

Instructor: Hod Lipston

Submitted time: 11/16/2021

Grace Hour Used: 1 h

Grace Hour Left: 98 h

**Results**

**Chart, radar chart, scatter chart

Description automatically generated**

Figure 1: Screenshot of Single Evolutionary Cube

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Longest Distance in 3s (m) | Evaluations times | Speed (m/s) | Speed (Diameter/s) | Spring Evaluations (times/s) |
| GA (Roulette Wheel with Fix Rate) | 13.05 | 2500 | 4.35 | 43.5 | 1000 |
| 20.50 | 2500 | 6.83 | 68.3 | 1000 |
| 13.00 | 2500 | 4.33 | 43.3 | 1000 |

Single evolutionary cube:

Two evolutionary cubes:

**Methods**

The main purpose of Assignment 3 Phase B is to find the longest distance a physics simulator can travel, which is a cube moving and bouncing on flat ground by using the Genetic Algorithm. The environmental conditions for integration step are set to be acceleration of gravity (g) = [0, 0, -9.81] 𝑚/𝑠^2, time step (dt) = 1e-3 with a spring evaluation of 1000 per second, mass of single mass point (m) = 0.1 kg, stiffness of ground = 1000 𝑁/𝑚, the friction coefficient of ground = 0.5.

The cube is formed by 12 springs with each rest length to be 0.1 m. The sine function is applied to each spring to simulate the force causing the extension and compression of the spring. This sine function could be expressed as 𝐿=𝑎+𝑏𝑠𝑖𝑛(𝜔𝑡+𝑐), where 𝑎 represents the rest length of that spring, and L represents the spring length after extension or compression. There are three parameters b, c, and k which could be changed to optimize this model. The ranges of those parameters are b (from -0.01 to 0.01), c (from 0 to 1), and the spring constant k (from 500 to 1000).

­­In order to get the motion of the entire cube, we calculate the motion of 8 masses separately. The 28 spring forces and the displacements caused by sine function are projected to x, y, and z directions. When each mass touched the ground, the ground is regarded as a spring with a k to be 1000 N/m, and the friction coefficient is 0.5.

When evolve the motion of the cube, b and c from sine function and spring coefficient are treated as the parameters in GA. The population was set to be 50, and the number of evaluations is set to be 2500. GA was run 3 times, we used roulette wheel as the selection method, in which the better members have more chances to survive. The fitness function will be the distance the robot traveled. A fixed number of chromosomes would be selected and put into crossover and mutation. The crossover will randomly exchange one point of parent to create a child. The mutation will randomly choose one point in the gene and change it. The data of the fastest robot will be stored. The program will finally find 3 sets of “best values for b, c and k” and take the average of them to get the best-representing data.

**Conclusion**

From Figure 2 and Figure 4, the learning curve shows the average longest distance the robot can travel is around 15.51 m, while the trend seems not to converge, which means it could be longer than what it has currently. The curve does not converge because the number of populations, iterations, and total runs are too small. However, due to the limit of performance for MATLAB and the personal computer hardware environment, it took more than 5 and a half hours for the program to finish all the calculations. Also, when test the motion of the cube, time step ‘dt’ and range of the spring coefficient ‘k’ will make a massive difference in the speed. Which is due to the way of motion calculation. A large spring coefficient and a long time-step will cause a large error in the calculation. To reduce the error, the time step is set to 0.001s, which makes the error smaller nearly 100 times. However, due to the limit of MATLAB, a smaller time step is impossible for the code.

**Performance Plots**

The learning curve is shown in figure below,

Chart

Description automatically generated

Figure 2: Learning Curve in 3 Runs

The dot plot is shown in figure below,

Chart

Description automatically generated

Figure 3: Dot Plot

The following figure shows the convergence plot,

Chart

Description automatically generated

Figure 4: Convergence Plot

**Appendix**

GA

%% Initialization:

tic % Start itering time

clearvars

close all

clc

n = 28;

w = 3\*pi;

%% Genetic Algorithm

run = 3; % runtimes

iter = 2500;%evolve times

popnum = 50; % number of population

path\_new = NaN(iter, n+1);

kept = NaN(iter, 1);

tdistance = NaN(popnum, 1);

Tdistance = NaN(popnum,iter);

alldata = NaN(iter,run);

bck=zeros(popnum,28\*3);

for j = 1:run

% Generate initial populations randomly:

pop = NaN(popnum, n);

for i=1:popnum

%create b c k

b = rand(28,1)\*(.02)-0.01; % -0.5 ~ 0.5

c = rand(28,1); %

k = rand(28,1)\*(1000-500)+500;

bck(i,:) = [b;c;k];

end

for N = 1:iter

%tic

for i = 1:popnum

tdistance(i) = Motion(w,bck(i,:));

end

Tdistance(:,N) = tdistance; % Store all the distances

[path\_best,idx\_best] = max(tdistance); % Find the longest path and its position in

fit = tdistance; % value of fitness

% store the best value:

kept(N) = path\_best;

bck2=zeros(15,28\*3);

bck3=zeros(15,28\*3);

for i = 1:15

bck1 = select(bck,fit); % Roulette Wheel Selection

bck2(i,:) = Cross(bck1); % Crossover

nm= randperm(popnum,1);

bck3(i,:) = Mutate(bck(nm,:)); % Mutation

end

[~,idx] = sort(tdistance,'descend');

bck = [bck(idx(1:20),:);bck2;bck3]; % Get good gene from parents

%toc

disp(N)

end

alldata(:,j) = kept;%record the distance

save ('data.mat')

end

toc

curve plotting

clear all

clc

load( 'data.mat') %% Learning Curve

x=2500; avel= mean(alldata,2); % calculated the mean

errorl=std(alldata,0,2)/4^0.5; % plot learning curve

X=[1:2500];

plot(X,avel)

hold on

errorbar(((2500/10):(2500/10):2500),avel((2500/10):(2500/10):2500),errorl((2500/10):(2500/10):2500),'o')

legend('GA','Error Bar')

%title('Learning Curve','FontSize',14)

ylabel('Distance(m)','FontSize',14)

xlabel('Number of Evaluation','FontSize',14)

%% Dot Plot

xx=[]; Dot=[];

Tdistance=Tdistance';

for i= 1:20 %% set x for dot plot

xx=[xx 1:2500];

Dot=[Dot;Tdistance(:,i)];

end

figure(2)

scatter(xx,Dot,2,'g') %dot plot

hold on

plot(X,alldata(:,3))%best fitness

legend('Dot','Best Fitness')

%title('Dot plot of the Best 40% ','FontSize',14)

ylabel('Distance(m)','FontSize',14)

xlabel('Number of Evaluation','FontSize',14)

hold off %% Convergence %Set compare point

for i =1:2500

g1=Tdistance(i,:);

number1=length(g1(g1>=2));

prop1(i)=number1/50;

end

for i =1:2500

g2=Tdistance(i,:);

number2=length(g2(g2>=5));

prop2(i)=number2/50;

end

for i =1:2500

g3=Tdistance(i,:);

number3=length(g3(g3>=8));

prop3(i)=number3/50;

end % Plot

figure(3)

plot([1:2500],prop1,[1:2500],prop2,[1:2500],prop3)

legend('Distance>2','Distance>5','Distance>8')

%title('Convergence from GA','FontSize',14)

ylabel('Proportion','FontSize',14)

xlabel('Evaluation','FontSize',14)

calculate the motion distance

function tdistance=Motion(w,bck)

global g

g = -9.81; % in m/s^2 %%%%%%

global dt

dt = .001;%%%%%%%%%%

global T;

T = 0;

kg = 1000 ;% ground k

u = 0.5;% ground friction

%load bck

b = bck(1:28);

c = bck(29:56);

k = bck(57:84);

m = .8;% mass

%origin length

l = .1;

L0 = .1;

%cube

Vo = [0 0 0; 0 L0 0; L0 L0 0; L0 0 0; 0 0 L0; 0 L0 L0; L0 L0 L0; L0 0 L0;];

V = [Vo(:,1)+.1,Vo(:,2)+.1,Vo(:,3)+.1];

%initial condition

posx = V(:,1);

posy = V(:,2);

posz = V(:,3);

vx = zeros(length(posx),1);%

vy = zeros(length(posx),1);

vz = zeros(length(posx),1);

agx = zeros(length(posx),1);%

agy = zeros(length(posx),1);

agz = zeros(length(posx),1);

mcposx = NaN(1,3/dt);

mcposy = NaN(1,3/dt);

mcposz = NaN(1,3/dt);

%

n = 0;

while n<=3000

n = n+1;

l0 = -b.\*sin(w\*T+c) + b.\*sin(w\*(T+dt)+c);%%calculate l0

V = [posx,posy,posz];%cube

[Fspring,mo] = spring(l,V,k,l0);%get spring force and l0in xyz

mox = mo(:,1);

moy = mo(:,2);

moz = mo(:,3);

%calculate mmotion

mtx=vx\*dt + mox;

mty=vy\*dt + moy;

mtz=vz\*dt + moz;

% get position

posx = posx + mtx; %%

posy = posy + mty;

posz = posz + mtz;

for i=1: length(posx) %% when touch ground

if posz(i) <= 0

agz(i)=-kg\*posz(i)/m;

if mtx(i)^2+mty(i)^2 ==0

agx(i)=0;

agy(i)=0;

else

agx(i)=kg\*posz(i)/m\*u\*(mtx(i)/sqrt(mtx(i)^2+mty(i)^2));

agy(i)=kg\*posz(i)/m\*u\*(mty(i)/sqrt(mtx(i)^2+mty(i)^2));

end

else

agz(i)=0;

agx(i)=0;

agy(i)=0;

end

end

%spring force

Fspringx = Fspring(:,1); %%

Fspringy = Fspring(:,2);

Fspringz = Fspring(:,3);

%spring acceleration

aspringx = Fspringx./m;

aspringy = Fspringy./m;

aspringz = Fspringz./m;

% speed

vx = vx + aspringx\*dt+agx\*dt;

vy = vy + aspringy\*dt+agy\*dt;

vz = vz + aspringz\*dt+g\*dt+agz\*dt;%%

% record position

Distancex(:,n) = posx;

Distancey(:,n) = posy;

Distancez(:,n) = posz;

T = T+dt;

end

tdistance = sqrt(mean(posx)^2 + mean(posy)^2);

end

calculate the spring force and moving direction

function [Fspring,moveL0] = spring(l,v,k,L0)

%l is orginal length of cube, v is current cube point position, k is spring constant

% to increase speed, loop is not used

%% debug

% clc

% clear

% l = 1;

% ll= 0.9;

% k=ones(1,28);

% v=[ 0 0 0;

% 0 ll 0;

% ll ll 0;

% ll 0 0;

% 0 0 ll;

% 0 ll ll;

% ll ll ll;

% ll 0 ll;];

% L0=ones(1,28)\*0.1;

%% get sine motion and force for spring and seprate into xyz direction

l1=l;

l2=l\*(2^0.5);

l3=l\*(3^0.5);

% getdirection of spring x y and z

lx1=(sum((v(1,:)-v(4,:)).^2))^0.5;

x14=(v(1,1)-v(4,1));

y14=(v(1,2)-v(4,2));

z14=(v(1,3)-v(4,3));

lx2=(sum((v(2,:)-v(3,:)).^2))^0.5;

x23=(v(2,1)-v(3,1));

y23=(v(2,2)-v(3,2));

z23=(v(2,3)-v(3,3));

lx3=(sum((v(5,:)-v(8,:)).^2))^0.5;

x58=(v(5,1)-v(8,1));

y58=(v(5,2)-v(8,2));

z58=(v(5,3)-v(8,3));

lx4=(sum((v(6,:)-v(7,:)).^2))^0.5;

x67=(v(6,1)-v(7,1));

y67=(v(6,2)-v(7,2));

z67=(v(6,3)-v(7,3));

ly1=(sum((v(1,:)-v(2,:)).^2))^0.5;

x12=(v(1,1)-v(2,1));

y12=(v(1,2)-v(2,2));

z12=(v(1,3)-v(2,3));

ly2=(sum((v(3,:)-v(4,:)).^2))^0.5;

x34=(v(3,1)-v(4,1));

y34=(v(3,2)-v(4,2));

z34=(v(3,3)-v(4,3));

ly3=(sum((v(5,:)-v(6,:)).^2))^0.5;

x56=(v(5,1)-v(6,1));

y56=(v(5,2)-v(6,2));

z56=(v(5,3)-v(6,3));

ly4=(sum((v(7,:)-v(8,:)).^2))^0.5;

x78=(v(7,1)-v(8,1));

y78=(v(7,2)-v(8,2));

z78=(v(7,3)-v(8,3));

lz1=(sum((v(1,:)-v(5,:)).^2))^0.5;

x15=(v(1,1)-v(5,1));

y15=(v(1,2)-v(5,2));

z15=(v(1,3)-v(5,3));

lz2=(sum((v(2,:)-v(6,:)).^2))^0.5;

x26=(v(2,1)-v(6,1));

y26=(v(2,2)-v(6,2));

z26=(v(2,3)-v(6,3));

lz3=(sum((v(3,:)-v(7,:)).^2))^0.5;

x37=(v(3,1)-v(7,1));

y37=(v(3,2)-v(7,2));

z37=(v(3,3)-v(7,3));

lz4=(sum((v(4,:)-v(8,:)).^2))^0.5;

x48=(v(4,1)-v(8,1));

y48=(v(4,2)-v(8,2));

z48=(v(4,3)-v(8,3));

% get force and sine motion of spring

fx14=(l1-lx1)\*k(1);

fx14x=fx14/lx1\*x14;

fx14y=fx14/lx1\*y14;

fx14z=fx14/lx1\*z14;

mx14x=L0(1)/lx1\*x14;

mx14y=L0(1)/lx1\*y14;

mx14z=L0(1)/lx1\*z14;

fx23=(l1-lx2)\*k(2);

fx23x=fx23/lx2\*x23;

fx23y=fx23/lx2\*y23;

fx23z=fx23/lx2\*z23;

mx23x=L0(2)/lx2\*x23;

mx23y=L0(2)/lx2\*y23;

mx23z=L0(2)/lx2\*z23;

fx58=(l1-lx3)\*k(3);

fx58x=fx58/lx3\*x58;

fx58y=fx58/lx3\*y58;

fx58z=fx58/lx3\*z58;

mx58x=L0(3)/lx3\*x58;

mx58y=L0(3)/lx3\*y58;

mx58z=L0(3)/lx3\*z58;

fx67=(l1-lx4)\*k(4);

fx67x=fx67/lx4\*x67;

fx67y=fx67/lx4\*y67;

fx67z=fx67/lx4\*z67;

mx67x=L0(4)/lx4\*x67;

mx67y=L0(4)/lx4\*y67;

mx67z=L0(4)/lx4\*z67;

fy12=(l1-ly1)\*k(5);

fy12x=fy12/ly1\*x12;

fy12y=fy12/ly1\*y12;

fy12z=fy12/ly1\*z12;

my12x=L0(5)/ly1\*x12;

my12y=L0(5)/ly1\*y12;

my12z=L0(5)/ly1\*z12;

fy34=(l1-ly2)\*k(6);

fy34x=fy34/ly2\*x34;

fy34y=fy34/ly2\*y34;

fy34z=fy34/ly2\*z34;

my34x=L0(6)/ly2\*x34;

my34y=L0(6)/ly2\*y34;

my34z=L0(6)/ly2\*z34;

fy56=(l1-ly3)\*k(7);

fy56x=fy56/ly3\*x56;

fy56y=fy56/ly3\*y56;

fy56z=fy56/ly3\*z56;

my56x=L0(7)/ly3\*x56;

my56y=L0(7)/ly3\*y56;

my56z=L0(7)/ly3\*z56;

fy78=(l1-ly4)\*k(8);

fy78x=fy78/ly4\*x78;

fy78y=fy78/ly4\*y78;

fy78z=fy78/ly4\*z78;

my78x=L0(8)/ly4\*x78;

my78y=L0(8)/ly4\*y78;

my78z=L0(8)/ly4\*z78;

fz15=(l1-lz1)\*k(9);

fz15x=fz15/lz1\*x15;

fz15y=fz15/lz1\*y15;

fz15z=fz15/lz1\*z15;

mz15x=L0(9)/lz1\*x15;

mz15y=L0(9)/lz1\*y15;

mz15z=L0(9)/lz1\*z15;

fz26=(l1-lz2)\*k(10);

fz26x=fz26/lz2\*x26;

fz26y=fz26/lz2\*y26;

fz26z=fz26/lz2\*z26;

mz26x=L0(10)/lz2\*x26;

mz26y=L0(10)/lz2\*y26;

mz26z=L0(10)/lz2\*z26;

fz37=(l1-lz3)\*k(11);

fz37x=fz37/lz3\*x37;

fz37y=fz37/lz3\*y37;

fz37z=fz37/lz3\*z37;

mz37x=L0(11)/lz3\*x37;

mz37y=L0(11)/lz3\*y37;

mz37z=L0(11)/lz3\*z37;

fz48=(l1-lz4)\*k(12);

fz48x=fz48/lz4\*x48;

fz48y=fz48/lz4\*y48;

fz48z=fz48/lz4\*z48;

mz48x=L0(12)/lz4\*x48;

mz48y=L0(12)/lz4\*y48;

mz48z=L0(12)/lz4\*z48;

% getdirection of spring xy yz and xz

lxy1=(sum((v(1,:)-v(3,:)).^2))^0.5;

x13=(v(1,1)-v(3,1));

y13=(v(1,2)-v(3,2));

z13=(v(1,3)-v(3,3));

lxy2=(sum((v(2,:)-v(4,:)).^2))^0.5;

x24=(v(2,1)-v(4,1));

y24=(v(2,2)-v(4,2));

z24=(v(2,3)-v(4,3));

lxy3=(sum((v(5,:)-v(7,:)).^2))^0.5;

x57=(v(5,1)-v(7,1));

y57=(v(5,2)-v(7,2));

z57=(v(5,3)-v(7,3));

lxy4=(sum((v(6,:)-v(8,:)).^2))^0.5;

x68=(v(6,1)-v(8,1));

y68=(v(6,2)-v(8,2));

z68=(v(6,3)-v(8,3));

lxz1=(sum((v(1,:)-v(8,:)).^2))^0.5;

x18=(v(1,1)-v(8,1));

y18=(v(1,2)-v(8,2));

z18=(v(1,3)-v(8,3));

lxz2=(sum((v(4,:)-v(5,:)).^2))^0.5;

x45=(v(4,1)-v(5,1));

y45=(v(4,2)-v(5,2));

z45=(v(4,3)-v(5,3));

lxz3=(sum((v(2,:)-v(7,:)).^2))^0.5;

x27=(v(2,1)-v(7,1));

y27=(v(2,2)-v(7,2));

z27=(v(2,3)-v(7,3));

lxz4=(sum((v(3,:)-v(6,:)).^2))^0.5;

x36=(v(3,1)-v(6,1));

y36=(v(3,2)-v(6,2));

z36=(v(3,3)-v(6,3));

lyz1=(sum((v(1,:)-v(6,:)).^2))^0.5;

x16=(v(1,1)-v(6,1));

y16=(v(1,2)-v(6,2));

z16=(v(1,3)-v(6,3));

lyz2=(sum((v(2,:)-v(5,:)).^2))^0.5;

x25=(v(2,1)-v(5,1));

y25=(v(2,2)-v(5,2));

z25=(v(2,3)-v(5,3));

lyz3=(sum((v(3,:)-v(8,:)).^2))^0.5;

x38=(v(3,1)-v(8,1));

y38=(v(3,2)-v(8,2));

z38=(v(3,3)-v(8,3));

lyz4=(sum((v(4,:)-v(7,:)).^2))^0.5;

x47=(v(4,1)-v(7,1));

y47=(v(4,2)-v(7,2));

z47=(v(4,3)-v(7,3));

% get force and sine motion of spring

fxy13=(l2-lxy1)\*k(13);

fxy13x=fxy13/lxy1\*x13;

fxy13y=fxy13/lxy1\*y13;

fxy13z=fxy13/lxy1\*z13;

mxy13x=L0(13)/lxy1\*x13;

mxy13y=L0(13)/lxy1\*y13;

mxy13z=L0(13)/lxy1\*z13;

fxy24=(l2-lxy2)\*k(14);

fxy24x=fxy24/lxy2\*x24;

fxy24y=fxy24/lxy2\*y24;

fxy24z=fxy24/lxy2\*z24;

mxy24x=L0(14)/lxy2\*x24;

mxy24y=L0(14)/lxy2\*y24;

mxy24z=L0(14)/lxy2\*z24;

fxy57=(l2-lxy3)\*k(15);

fxy57x=fxy57/lxy3\*x57;

fxy57y=fxy57/lxy3\*y57;

fxy57z=fxy57/lxy3\*z57;

mxy57x=L0(15)/lxy3\*x57;

mxy57y=L0(15)/lxy3\*y57;

mxy57z=L0(15)/lxy3\*z57;

fxy68=(l2-lxy4)\*k(16);

fxy68x=fxy68/lxy4\*x68;

fxy68y=fxy68/lxy4\*y68;

fxy68z=fxy68/lxy4\*z68;

mxy68x=L0(16)/lxy4\*x68;

mxy68y=L0(16)/lxy4\*y68;

mxy68z=L0(16)/lxy4\*z68;

fxz18=(l2-lxz1)\*k(17);

fxz18x=fxz18/lxz1\*x18;

fxz18y=fxz18/lxz1\*y18;

fxz18z=fxz18/lxz1\*z18;

mxz18x=L0(17)/lxz1\*x18;

mxz18y=L0(17)/lxz1\*y18;

mxz18z=L0(17)/lxz1\*z18;

fxz45=(l2-lxz2)\*k(18);

fxz45x=fxz45/lxz2\*x45;

fxz45y=fxz45/lxz2\*y45;

fxz45z=fxz45/lxz2\*z45;

mxz45x=L0(18)/lxz2\*x45;

mxz45y=L0(18)/lxz2\*y45;

mxz45z=L0(18)/lxz2\*z45;

fxz27=(l2-lxz3)\*k(19);

fxz27x=fxz27/lxz3\*x27;

fxz27y=fxz27/lxz3\*y27;

fxz27z=fxz27/lxz3\*z27;

mxz27x=L0(19)/lxz3\*x27;

mxz27y=L0(19)/lxz3\*y27;

mxz27z=L0(19)/lxz3\*z27;

fxz36=(l2-lxz4)\*k(20);

fxz36x=fxz36/lxz4\*x36;

fxz36y=fxz36/lxz4\*y36;

fxz36z=fxz36/lxz4\*z36;

mxz36x=L0(20)/lxz4\*x36;

mxz36y=L0(20)/lxz4\*y36;

mxz36z=L0(20)/lxz4\*z36;

fyz16=(l2-lyz1)\*k(21);

fyz16x=fyz16/lyz1\*x16;

fyz16y=fyz16/lyz1\*y16;

fyz16z=fyz16/lyz1\*z16;

myz16x=L0(21)/lyz1\*x16;

myz16y=L0(21)/lyz1\*y16;

myz16z=L0(21)/lyz1\*z16;

fyz25=(l2-lyz2)\*k(22);

fyz25x=fyz25/lyz2\*x25;

fyz25y=fyz25/lyz2\*y25;

fyz25z=fyz25/lyz2\*z25;

myz25x=L0(22)/lyz2\*x25;

myz25y=L0(22)/lyz2\*y25;

myz25z=L0(22)/lyz2\*z25;

fyz38=(l2-lyz3)\*k(23);

fyz38x=fyz38/lyz3\*x38;

fyz38y=fyz38/lyz3\*y38;

fyz38z=fyz38/lyz3\*z38;

myz38x=L0(23)/lyz3\*x38;

myz38y=L0(23)/lyz3\*y38;

myz38z=L0(23)/lyz3\*z38;

fyz47=(l2-lyz4)\*k(24);

fyz47x=fyz47/lyz4\*x47;

fyz47y=fyz47/lyz4\*y47;

fyz47z=fyz47/lyz4\*z47;

myz47x=L0(24)/lyz4\*x47;

myz47y=L0(24)/lyz4\*y47;

myz47z=L0(24)/lyz4\*z47;

% getdirection of spring xyz

lxyz1=(sum((v(1,:)-v(7,:)).^2))^0.5;

x17=(v(1,1)-v(7,1));

y17=(v(1,2)-v(7,2));

z17=(v(1,3)-v(7,3));

lxyz2=(sum((v(3,:)-v(5,:)).^2))^0.5;

x35=(v(3,1)-v(5,1));

y35=(v(3,2)-v(5,2));

z35=(v(3,3)-v(5,3));

lxyz3=(sum((v(2,:)-v(8,:)).^2))^0.5;

x28=(v(2,1)-v(8,1));

y28=(v(2,2)-v(8,2));

z28=(v(2,3)-v(8,3));

lxyz4=(sum((v(4,:)-v(6,:)).^2))^0.5;

x46=(v(4,1)-v(6,1));

y46=(v(4,2)-v(6,2));

z46=(v(4,3)-v(6,3));

% get force and sine motion of spring

fxyz17=(l3-lxyz1)\*k(25);

fxyz17x=fxyz17/lxyz1\*x17;

fxyz17y=fxyz17/lxyz1\*y17;

fxyz17z=fxyz17/lxyz1\*z17;

mxyz17x=L0(25)/lxyz1\*x17;

mxyz17y=L0(25)/lxyz1\*y17;

mxyz17z=L0(25)/lxyz1\*z17;

fxyz35=(l3-lxyz2)\*k(26);

fxyz35x=fxyz35/lxyz2\*x35;

fxyz35y=fxyz35/lxyz2\*y35;

fxyz35z=fxyz35/lxyz2\*z35;

mxyz35x=L0(26)/lxyz2\*x35;

mxyz35y=L0(26)/lxyz2\*y35;

mxyz35z=L0(26)/lxyz2\*z35;

fxyz28=(l3-lxyz3)\*k(27);

fxyz28x=fxyz28/lxyz3\*x28;

fxyz28y=fxyz28/lxyz3\*y28;

fxyz28z=fxyz28/lxyz3\*z28;

mxyz28x=L0(27)/lxyz3\*x28;

mxyz28y=L0(27)/lxyz3\*y28;

mxyz28z=L0(27)/lxyz3\*z28;

fxyz46=(l3-lxyz4)\*k(28);

fxyz46x=fxyz46/lxyz4\*x46;

fxyz46y=fxyz46/lxyz4\*y46;

fxyz46z=fxyz46/lxyz4\*z46;

mxyz46x=L0(28)/lxyz4\*x46;

mxyz46y=L0(28)/lxyz4\*y46;

mxyz46z=L0(28)/lxyz4\*z46;

%% put commbine force and motion into 8 mass point in xyz direction

Fspring=zeros(8,3);

Fspring(1,1)=(fx14x+fy12x+fz15x+fxy13x+fxz18x+fyz16x+fxyz17x);

Fspring(1,2)=(fx14y+fy12y+fz15y+fxy13y+fxz18y+fyz16y+fxyz17y);

Fspring(1,3)=(fx14z+fy12z+fz15z+fxy13z+fxz18z+fyz16z+fxyz17z);

Fspring(2,1)=(fx23x-fy12x+fz26x+fxy24x+fxz27x+fyz25x+fxyz28x);

Fspring(2,2)=(fx23y-fy12y+fz26y+fxy24y+fxz27y+fyz25y+fxyz28y);

Fspring(2,3)=(fx23z-fy12z+fz26z+fxy24z+fxz27z+fyz25z+fxyz28z);

Fspring(3,1)=(-fx23x+fy34x+fz37x-fxy13x+fxz36x+fyz38x+fxyz35x);

Fspring(3,2)=(-fx23y+fy34y+fz37y-fxy13y+fxz36y+fyz38y+fxyz35y);

Fspring(3,3)=(-fx23z+fy34z+fz37z-fxy13z+fxz36z+fyz38z+fxyz35z);

Fspring(4,1)=(-fx14x-fy34x+fz48x-fxy24x+fxz45x+fyz47x+fxyz46x);

Fspring(4,2)=(-fx14y-fy34y+fz48y-fxy24y+fxz45y+fyz47y+fxyz46y);

Fspring(4,3)=(-fx14z-fy34z+fz48z-fxy24z+fxz45z+fyz47z+fxyz46z);

Fspring(5,1)=(fx58x+fy56x-fz15x+fxy57x-fxz45x-fyz25x-fxyz35x);

Fspring(5,2)=(fx58y+fy56y-fz15y+fxy57y-fxz45y-fyz25y-fxyz35y);

Fspring(5,3)=(fx58z+fy56z-fz15z+fxy57z-fxz45z-fyz25z-fxyz35z);

Fspring(6,1)=(fx67x-fy56x-fz26x+fxy68x-fxz36x-fyz16x-fxyz46x);

Fspring(6,2)=(fx67y-fy56y-fz26y+fxy68y-fxz36y-fyz16y-fxyz46y);

Fspring(6,3)=(fx67z-fy56z-fz26z+fxy68z-fxz36z-fyz16z-fxyz46z);

Fspring(7,1)=(-fx67x+fy78x-fz37x-fxy57x-fxz27x-fyz47x-fxyz17x);

Fspring(7,2)=(-fx67y+fy78y-fz37y-fxy57y-fxz27y-fyz47y-fxyz17y);

Fspring(7,3)=(-fx67z+fy78z-fz37z-fxy57z-fxz27z-fyz47z-fxyz17z);

Fspring(8,1)=(-fx58x-fy78x-fz48x-fxy68x-fxz18x-fyz38x-fxyz28x);

Fspring(8,2)=(-fx58y-fy78y-fz48y-fxy68y-fxz18y-fyz38y-fxyz28y);

Fspring(8,3)=(-fx58z-fy78z-fz48z-fxy68z-fxz18z-fyz38z-fxyz28z);

moveL0=zeros(8,3);

moveL0(1,1)=(mx14x+my12x+mz15x+mxy13x+mxz18x+myz16x+mxyz17x)/7;

moveL0(1,2)=(mx14y+my12y+mz15y+mxy13y+mxz18y+myz16y+mxyz17y)/7;

moveL0(1,3)=(mx14z+my12z+mz15z+mxy13z+mxz18z+myz16z+mxyz17z)/7;

moveL0(2,1)=(mx23x-my12x+mz26x+mxy24x+mxz27x+myz25x+mxyz28x)/7;

moveL0(2,2)=(mx23y-my12y+mz26y+mxy24y+mxz27y+myz25y+mxyz28y)/7;

moveL0(2,3)=(mx23z-my12z+mz26z+mxy24z+mxz27z+myz25z+mxyz28z)/7;

moveL0(3,1)=(-mx23x+my34x+mz37x-mxy13x+mxz36x+myz38x+mxyz35x)/7;

moveL0(3,2)=(-mx23y+my34y+mz37y-mxy13y+mxz36y+myz38y+mxyz35y)/7;

moveL0(3,3)=(-mx23z+my34z+mz37z-mxy13z+mxz36z+myz38z+mxyz35z)/7;

moveL0(4,1)=(-mx14x-my34x+mz48x-mxy24x+mxz45x+myz47x+mxyz46x)/7;

moveL0(4,2)=(-mx14y-my34y+mz48y-mxy24y+mxz45y+myz47y+mxyz46y)/7;

moveL0(4,3)=(-mx14z-my34z+mz48z-mxy24z+mxz45z+myz47z+mxyz46z)/7;

moveL0(5,1)=(mx58x+my56x-mz15x+mxy57x-mxz45x-myz25x-mxyz35x)/7;

moveL0(5,2)=(mx58y+my56y-mz15y+mxy57y-mxz45y-myz25y-mxyz35y)/7;

moveL0(5,3)=(mx58z+my56z-mz15z+mxy57z-mxz45z-myz25z-mxyz35z)/7;

moveL0(6,1)=(mx67x-my56x-mz26x+mxy68x-mxz36x-myz16x-mxyz46x)/7;

moveL0(6,2)=(mx67y-my56y-mz26y+mxy68y-mxz36y-myz16y-mxyz46y)/7;

moveL0(6,3)=(mx67z-my56z-mz26z+mxy68z-mxz36z-myz16z-mxyz46z)/7;

moveL0(7,1)=(-mx67x+my78x-mz37x-mxy57x-mxz27x-myz47x-mxyz17x)/7;

moveL0(7,2)=(-mx67y+my78y-mz37y-mxy57y-mxz27y-myz47y-mxyz17y)/7;

moveL0(7,3)=(-mx67z+my78z-mz37z-mxy57z-mxz27z-myz47z-mxyz17z)/7;

moveL0(8,1)=(-mx58x-my78x-mz48x-mxy68x-mxz18x-myz38x-mxyz28x)/7;

moveL0(8,2)=(-mx58y-my78y-mz48y-mxy68y-mxz18y-myz38y-mxyz28y)/7;

moveL0(8,3)=(-mx58z-my78z-mz48z-mxy68z-mxz18z-myz38z-mxyz28z)/7;

end

Mutate

function output = Mutate(a)

% choose point for b c k

randnum1 = randperm(length(a)/3,1);

randnum2 = randperm(length(a)/3,1)+28;

randnum3 = randperm(length(a)/3,1)+56;

%mutate

a(randnum1(1)) = rand\*(.02)-0.01;

a(randnum2(1)) = rand(1);

a(randnum3(1)) = rand\*(500)+500;

output = a;

end

Cross over

function output = Cross(chrom1)

a = chrom1(1,:);%parent1

b = chrom1(2,:);%parent2

%choosee point for b c k

randnum1 = randperm(length(a)/3,2);

randnum2 = randperm(length(a)/3,2)+28;

randnum3 = randperm(length(a)/3,2)+56;

%cross the point to create child

a(randnum1(1)) = b(randnum1(2));

a(randnum2(1)) = b(randnum2(2));

a(randnum3(1)) = b(randnum3(2));

output = a;

end

Create video for one cube

clc

clear

load( 'data.mat')

bck=bck(1,:);

%% insert the motion part

% video part is just directly changed from motion

global g

g = -9.81; % in m/s^2

global dt

dt = .001;

global T;

T = 0;

kg = 1000 ;% ground k

u = 0.5;% friction

%load bck

b = bck(1:28);

c = bck(29:56);

k = bck(57:84);

% mass

m = .8;

%origin length

l = .1;

L0 = .1;

%cube

Vo = [0 0 0; 0 L0 0; L0 L0 0; L0 0 0; 0 0 L0; 0 L0 L0; L0 L0 L0; L0 0 L0;];

V = [Vo(:,1)+.1,Vo(:,2)+.1,Vo(:,3)+.1];

%initial condition

posx = V(:,1);

posy = V(:,2);

posz = V(:,3);

vx = zeros(length(posx),1);%

vy = zeros(length(posx),1);

vz = zeros(length(posx),1);

agx = zeros(length(posx),1);%

agy = zeros(length(posx),1);

agz = zeros(length(posx),1);

mcposx = NaN(1,3/dt);

mcposy = NaN(1,3/dt);

mcposz = NaN(1,3/dt);

%

n = 0;

while n<=3000

n = n+1;

l0 = -b.\*sin(w\*T+c) + b.\*sin(w\*(T+dt)+c);%%calculate l0

V = [posx,posy,posz]; %cube

[Fspring,mo] = spring(l,V,k,l0);%get spring force and l0in xyz

mox = mo(:,1);

moy = mo(:,2);

moz = mo(:,3);

%calculate mmotion

mtx=vx\*dt + mox;

mty=vy\*dt + moy;

mtz=vz\*dt + moz;

% get position

posx = posx + mtx;

posy = posy + mty;

posz = posz + mtz;

for i=1: length(posx) %% when touch ground

if posz(i) <= 0

agz(i)=-kg\*posz(i)/m;

if mtx(i)^2+mty(i)^2 ==0

agx(i)=0;

agy(i)=0;

else

agx(i)=kg\*posz(i)/m\*u\*(mtx(i)/sqrt(mtx(i)^2+mty(i)^2));

agy(i)=kg\*posz(i)/m\*u\*(mty(i)/sqrt(mtx(i)^2+mty(i)^2));

end

else

agz(i)=0;

agx(i)=0;

agy(i)=0;

end

end

%spring force

Fspringx = Fspring(:,1); %%

Fspringy = Fspring(:,2);

Fspringz = Fspring(:,3);

%spring acceleration

aspringx = Fspringx./m;

aspringy = Fspringy./m;

aspringz = Fspringz./m;

% speed

vx = vx + aspringx\*dt+agx\*dt;

vy = vy + aspringy\*dt+agy\*dt;

vz = vz + aspringz\*dt+g\*dt+agz\*dt;%%

% record position

Distancex(:,n) = posx;

Distancey(:,n) = posy;

Distancez(:,n) = posz;

T = T+dt;

end

tdistance = sqrt(mean(posx)^2 + mean(posy)^2);

%% Plot part

ccc=0;

E=[ 1 2;1 3;1 4;1 5;

1 6;1 7;1 8;2 3;

2 4;2 5;2 6;2 7;

2 8;3 4;3 5;3 6;

3 7;3 8;4 5;4 6;

4 7;4 8;5 6;5 7;

5 8;6 7;6 8;7 8]; %spring link

for loop=10:10:3000

V=[Distancex(:,loop) Distancey(:,loop) Distancez(:,loop)]; %cube

figure(1) %plot cube

scatter3(V(:,1),V(:,2),V(:,3));

hold on;

for i=1:size(E,1)

V1=V(E(i,1),:);

V2=V(E(i,2),:);

line([V1(1) V2(1)],[V1(2) V2(2)],[V1(3) V2(3)]);

end

axis equal

axis([-5 5 -5 5 -1 5])%set frame size ,can change 5 to 1 to get more detail video

hold off

ccc=ccc+1;

M(ccc)=getframe;

end

ylabel('Meter','FontSize',14)

xlabel('Meter','FontSize',14)

zlabel('Meter','FontSize',14)

%title('Wire Frame Cube','FontSize',14)

figure(2)

axis off

movie(M,1,100)

Create video for two cubes

clc

clear

% video part is just directly changed from motion

load( 'data.mat')

bckl=bck;

bck=bckl(1,:);

%% cube1 motion part

global g

g = -9.81; % in m/s^2 %%%%%%

global dt

dt = .001;%%%%%%%%%%

global T;

T = 0;

kg = 1000 ;% ground k

u = 0.5;% ground friction

%load bck

b = bck(1:28);

c = bck(29:56);

k = bck(57:84);

m = .8;% mass

%origin length

l = .1;

L0 = .1;

%cube

Vo = [0 0 0; 0 L0 0; L0 L0 0; L0 0 0; 0 0 L0; 0 L0 L0; L0 L0 L0; L0 0 L0;];

V = [Vo(:,1)+.1,Vo(:,2)+.1,Vo(:,3)+.1];

%initial condition

posx = V(:,1);

posy = V(:,2);

posz = V(:,3);

vx = zeros(length(posx),1);%

vy = zeros(length(posx),1);

vz = zeros(length(posx),1);

agx = zeros(length(posx),1);%

agy = zeros(length(posx),1);

agz = zeros(length(posx),1);

mcposx = NaN(1,3/dt);

mcposy = NaN(1,3/dt);

mcposz = NaN(1,3/dt);

%

n = 0;

while n<=3000

n = n+1;

l0 = -b.\*sin(w\*T+c) + b.\*sin(w\*(T+dt)+c);%%calculate l0

V = [posx,posy,posz];%cube

[Fspring,mo] = spring(l,V,k,l0);%get spring force and l0in xyz

mox = mo(:,1);

moy = mo(:,2);

moz = mo(:,3);

%calculate mmotion

mtx=vx\*dt + mox;

mty=vy\*dt + moy;

mtz=vz\*dt + moz;

% get position

posx = posx + mtx; %%

posy = posy + mty;

posz = posz + mtz;

for i=1: length(posx) %% when touch ground

if posz(i) <= 0

agz(i)=-kg\*posz(i)/m;

if mtx(i)^2+mty(i)^2 ==0

agx(i)=0;

agy(i)=0;

else

agx(i)=kg\*posz(i)/m\*u\*(mtx(i)/sqrt(mtx(i)^2+mty(i)^2));

agy(i)=kg\*posz(i)/m\*u\*(mty(i)/sqrt(mtx(i)^2+mty(i)^2));

end

else

agz(i)=0;

agx(i)=0;

agy(i)=0;

end

end

%spring force

Fspringx = Fspring(:,1); %%

Fspringy = Fspring(:,2);

Fspringz = Fspring(:,3);

%spring acceleration

aspringx = Fspringx./m;

aspringy = Fspringy./m;

aspringz = Fspringz./m;

% speed

vx = vx + aspringx\*dt+agx\*dt;

vy = vy + aspringy\*dt+agy\*dt;

vz = vz + aspringz\*dt+g\*dt+agz\*dt;%%

% record position

Distancex(:,n) = posx;

Distancey(:,n) = posy;

Distancez(:,n) = posz;

T = T+dt;

end

tdistance = sqrt(mean(posx)^2 + mean(posy)^2);

Dx1=Distancex;

Dy1=Distancey;

Dz1=Distancez;

%% cube2 motion part

bck=bckl(20,:);

global g

g = -9.81; % in m/s^2 %%%%%%

global dt

dt = .001;%%%%%%%%%%

global T;

T = 0;

kg = 1000 ;% ground k

u = 0.5;% ground friction

%load bck

b = bck(1:28);

c = bck(29:56);

k = bck(57:84);

m = .8;% mass

%origin length

l = .1;

L0 = .1;

%cube

Vo = [0 0 0; 0 L0 0; L0 L0 0; L0 0 0; 0 0 L0; 0 L0 L0; L0 L0 L0; L0 0 L0;];

V = [Vo(:,1)+.1,Vo(:,2)+.1,Vo(:,3)+.1];

%initial condition

posx = V(:,1);

posy = V(:,2);

posz = V(:,3);

vx = zeros(length(posx),1);%

vy = zeros(length(posx),1);

vz = zeros(length(posx),1);

agx = zeros(length(posx),1);%

agy = zeros(length(posx),1);

agz = zeros(length(posx),1);

mcposx = NaN(1,3/dt);

mcposy = NaN(1,3/dt);

mcposz = NaN(1,3/dt);

%

n = 0;

while n<=3000

n = n+1;

l0 = -b.\*sin(w\*T+c) + b.\*sin(w\*(T+dt)+c);%%calculate l0

V = [posx,posy,posz];%cube

[Fspring,mo] = spring(l,V,k,l0);%get spring force and l0in xyz

mox = mo(:,1);

moy = mo(:,2);

moz = mo(:,3);

%calculate mmotion

mtx=vx\*dt + mox;

mty=vy\*dt + moy;

mtz=vz\*dt + moz;

% get position

posx = posx + mtx; %%

posy = posy + mty;

posz = posz + mtz;

for i=1: length(posx) %% when touch ground

if posz(i) <= 0

agz(i)=-kg\*posz(i)/m;

if mtx(i)^2+mty(i)^2 ==0

agx(i)=0;

agy(i)=0;

else

agx(i)=kg\*posz(i)/m\*u\*(mtx(i)/sqrt(mtx(i)^2+mty(i)^2));

agy(i)=kg\*posz(i)/m\*u\*(mty(i)/sqrt(mtx(i)^2+mty(i)^2));

end

else

agz(i)=0;

agx(i)=0;

agy(i)=0;

end

end

%spring force

Fspringx = Fspring(:,1); %%

Fspringy = Fspring(:,2);

Fspringz = Fspring(:,3);

%spring acceleration

aspringx = Fspringx./m;

aspringy = Fspringy./m;

aspringz = Fspringz./m;

% speed

vx = vx + aspringx\*dt+agx\*dt;

vy = vy + aspringy\*dt+agy\*dt;

vz = vz + aspringz\*dt+g\*dt+agz\*dt;%%

% record position

Distancex(:,n) = posx;

Distancey(:,n) = posy;

Distancez(:,n) = posz;

T = T+dt;

end

tdistance = sqrt(mean(posx)^2 + mean(posy)^2);

Dx2=Distancex;

Dy2=Distancey;

Dz2=Distancez;

%% Plot part

ccc=0;

E=[ 1 2;1 3;1 4;1 5;

1 6;1 7;1 8;2 3;

2 4;2 5;2 6;2 7;

2 8;3 4;3 5;3 6;

3 7;3 8;4 5;4 6;

4 7;4 8;5 6;5 7;

5 8;6 7;6 8;7 8]; %%spring linkk

for loop=10:10:3000

V11=[Dx1(:,loop) Dy1(:,loop) Dz1(:,loop)];

V22=[Dx2(:,loop) Dy2(:,loop) Dz2(:,loop)];

figure(1)

scatter3(V11(:,1),V11(:,2),V11(:,3)); %%plot two cube

hold on;

for i=1:size(E,1)

V1=V11(E(i,1),:);

V2=V11(E(i,2),:);

line([V1(1) V2(1)],[V1(2) V2(2)],[V1(3) V2(3)]);

end

hold on

scatter3(V22(:,1),V22(:,2),V22(:,3));

hold on;

for i=1:size(E,1)

V1s=V22(E(i,1),:);

V2s=V22(E(i,2),:);

line([V1s(1) V2s(1)],[V1s(2) V2s(2)],[V1s(3) V2s(3)]);

end

axis equal

axis([-5 5 -5 5 0 5])%set frame size ,can set to 1 to get more detail video

hold off

ccc=ccc+1;

M(ccc)=getframe;

end

ylabel('Meter','FontSize',14)

xlabel('Meter','FontSize',14)

zlabel('Meter','FontSize',14)

title('Wire Frame Cube','FontSize',14)

figure(2)

axis off

movie(M,1,100)