% Lab 1: Rollercoaster

%

% Objectives:

%

% -Design a rollercoaster with a loop, 0G parabola,

% banked turn, braking section, and smooth transitions between all.

% -Determine G-Forces acting throughout rollercoaster.

% -Plot velocity along length of coaster.

%

% Assumptions:

%

% -Treat rollercoaster cart as a point mass.

% -Neglect drag (always) and friction (except in braking section)

% -Initial velocity is zero

%

% Constraints:

%

% -Max track length: 1250m

% -Upwards G's < 6

% -Downwards G's < 1

% -Forwards G's < 5

% -Backwards G's < 4

% -Lateral G's < 3

%------------------------------------------------------------------------------

% House Cleaning

clc

close all

% Initial Conditions

h0 = 125; % [m]

x0 = 0; % [m]

y0 = 100; % [m]

g = 9.81; % [m/s^2]

%% Roller Coaster

% Ramp

[h1,x1,y1,vel1,G1] = ramp(h0,x0,y0,40,50,g);

% Transition Out of Ramp, Into Loop (Curve)

[h2,x2,y2,vel2,G2,arc\_length\_trans1] = transition\_down\_curve(h0,h1(end),x1(end),y1(end),40,40,g);

% Loop

[h3,x3,y3,vel3,G3,arc\_length\_loop] = loop(h0,h2(end),x2(end),y2(end),17,g);

% Transition Out Of Loop, Into Banked Turn (Straight Line)

[h4,x4,y4,vel4,G4] = transition\_line(h0,h3(end),x3(end),y3(end),50,g);

% Banked Turn

[h5,x5,y5,vel5,lat\_G5,vert\_G5,arc\_length\_banked] = banked\_turn(h0,h4(end),x4(end),y4(end),45,25,g);

% Loop

[h6,x6,y6,vel6,G6,arc\_length\_loop2] = loop(h0,h5(end),x5(end),y5(end),17,g);

% Transition Out of Loop, Into Curve Transition (Straight Line)

[h7,x7,y7,vel7,G7] = transition\_line\_reverse(h0,h6(end),x6(end),y6(end),20,g);

% Transition Out of Straight Line Into Parabolic Hill (Curve)

[h8,x8,y8,vel8,G8,arc\_length\_trans2] = transition\_up\_curve\_reverse(h0,h7(end),x7(end),y7(end),20,40,g);

% Parabolic Hill

[h9,x9,y9,velz9,velx9,G9,arc\_length\_parabola] = parabola(h8(end),x8(end),y8(end),vel8(end),0,20,g);

vel9 = sqrt(velx9^2 + velz9.^2);

% Ramp Out of Parabolic Hill

[h10,x10,y10,vel10,G10] = ramp\_reverse(h0,h9(end),x9(end),y9(end),20,150,g);

% Transition to Ground

[h11,x11,y11,vel11,G11,arc\_length\_trans3] = transition\_down\_curve\_reverse(h0,h10(end),x10(end),y10(end),30,40,g);

% Braking

[h12,x12,y12,vel12,vert\_G12,back\_G12,brake\_length] = braking(h0,h11(end),x11(end),y11(end),g);

% Verify Total Track Length is Within Range

Track\_Length = 50 + arc\_length\_trans1 + arc\_length\_loop + 50 + arc\_length\_banked...

+ arc\_length\_loop2 + 30 + arc\_length\_trans2 + arc\_length\_parabola + 50 + arc\_length\_trans3 + brake\_length(1)

%% Put Together Plotting Vectors

% Positions

x\_pos = [x1,x2,x3,x4,x5,x6,x7,x8,x9,x10,x11,x12];

y\_pos = [y1,y2,y3,y4,y5,y6,y7,y8,y9,y10,y11,y12];

h\_pos = [h1,h2,h3,h4,h5,h6,h7,h8,h9,h10,h11,h12];

% Velocities

vel\_whole\_track = [vel1,vel2,vel3,vel4,vel5,vel6,vel7,vel8,vel9,vel10,vel11,vel12];

% G's Experienced

G\_whole\_vert = [G1,G2,G3,G4,vert\_G5,G6,G7,G8,G9,G10,G11,vert\_G12];

G\_whole\_lat = [zeros(1,length(G1)),zeros(1,length(G2)),zeros(1,length(G3)),zeros(1,length(G4))...

,lat\_G5,zeros(1,length(G6)),zeros(1,length(G7)),zeros(1,length(G8)),zeros(1,length(G9)),zeros(1,length(G10))...

,zeros(1,length(G11)),zeros(1,length(back\_G12))];

G\_whole\_back = [zeros(1,length(G1)),zeros(1,length(G2)),zeros(1,length(G3)),zeros(1,length(G4))...

,zeros(1,length(lat\_G5)),zeros(1,length(G6)),zeros(1,length(G7)),zeros(1,length(G8)),zeros(1,length(G9))...

,zeros(1,length(G10)),zeros(1,length(G11)),back\_G12];

%% Plot Rollercoaster

% Velocity

figure(1);

scatter3(x\_pos,y\_pos,h\_pos,10,vel\_whole\_track,'filled')

hold on

grid on; grid minor

title('Roller Coaster')

c = colorbar;

c.Label.String = 'Speed [m/s]';

xlabel('x Postion [m]')

ylabel('y Position [m]')

zlabel('Height [m]')

hold off

% Vertical G's

figure(2);

scatter3(x\_pos,y\_pos,h\_pos,10,G\_whole\_vert,'filled')

hold on

grid on; grid minor

title('Roller Coaster')

c = colorbar;

c.Label.String = "Vertical G's";

xlabel('x Postion [m]')

ylabel('y Position [m]')

zlabel('Height [m]')

hold off

% Lateral G's

figure(3);

scatter3(x\_pos,y\_pos,h\_pos,10,G\_whole\_lat,'filled')

hold on

grid on; grid minor

title('Roller Coaster')

c = colorbar;

c.Label.String = "Lateral G's";

xlabel('x Postion [m]')

ylabel('y Position [m]')

zlabel('Height [m]')

hold off

% Backward G's

figure(4);

scatter3(x\_pos,y\_pos,h\_pos,10,G\_whole\_back,'filled')

hold on

grid on; grid minor

title('Roller Coaster')

c = colorbar;

c.Label.String = "Backwards G's";

xlabel('x Postion [m]')

ylabel('y Position [m]')

zlabel('Height [m]')

hold off

function[h\_current,x\_current,y\_current,vel,G\_ramp] = ramp(h0,x,y,ramp\_theta,ramp\_length,g)

% Parameters

ramp\_pos = (0:0.1:ramp\_length);

% Current Positions

h\_current = h0 - ramp\_pos \* sind(ramp\_theta);

x\_current = x + ramp\_pos \* cosd(ramp\_theta);

y\_current = y + zeros(1,length(x\_current));

% Velocity

vel = sqrt(2\*g\*(h0-h\_current));

% Normal Force

N = g \* cosd(ramp\_theta);

% G's Experienced

G\_ramp = zeros(1,length(x\_current));

G\_ramp = G\_ramp + N ./ g;

end

function[h\_current,x\_current,y\_current,vel,G\_ramp2] = ramp\_reverse(h0,h,x,y,ramp\_theta2,ramp\_length2,g)

% Parameters

ramp\_pos = (0:0.1:ramp\_length2);

% Current Positions

h\_current = h - ramp\_pos \* sind(ramp\_theta2);

x\_current = x - ramp\_pos \* cosd(ramp\_theta2);

y\_current = y + zeros(1,length(x\_current));

% Velocity

vel = sqrt(2\*g\*(h0-h\_current));

% Normal Force

N = g \* cosd(ramp\_theta2);

% G's Experienced

G\_ramp2 = zeros(1,length(x\_current));

G\_ramp2 = G\_ramp2 + N ./ g;

end

function[h\_current,x\_current,y\_current,vel,G\_trans1,arc\_length] = transition\_down\_curve(h0,h,x,y,theta\_1,R,g)

% Parameters

theta\_range = (360 - theta\_1:360);

arc\_length = (theta\_1 / 360) \* 2 \* pi \* R;

% Current Positions

h\_current = h + R \* cosd(360 - theta\_1) - R \* cosd(theta\_range);

x\_current = x - R \* sind(360 - theta\_1) + R \* sind(theta\_range);

y\_current = y + zeros(1,length(x\_current));

% Velocity

vel = sqrt(2\*g\*(h0-h\_current));

% Normal Force

N = g \* cosd(theta\_range) + (vel.^2 / R);

% G's Experienced

G\_trans1 = N ./ g;

end

function[h\_current,x\_current,y\_current,vel,G\_trans1,arc\_length] = transition\_down\_curve\_reverse(h0,h,x,y,theta\_5,R,g)

% Parameters

theta\_range = (theta\_5:-1:0);

arc\_length = (theta\_5 / 360) \* 2 \* pi \* R;

% Current Positions

h\_current = h + (R\* cosd(theta\_5) - R \* cosd(theta\_range));

x\_current = x - R \* sind(theta\_5) + R \* sind(theta\_range);

y\_current = y + zeros(1,length(x\_current));

% Velocity

vel = sqrt(2\*g\*(h0-h\_current));

% Normal Force

N = g \* cosd(theta\_range) + (vel.^2 / R);

% G's Experienced

G\_trans1 = N ./ g;

end

function[h\_current,x\_current,y\_current,vel,G\_loop,circumference] = loop(h0,h,x,y,R,g)

% Parameters

circumference = 2 \* pi \* R; % [m]

theta\_range = 0:360; % [^o]

% Current Positions

h\_current = h + (R - R\*cosd(theta\_range));

x\_current = x + R\*sind(theta\_range);

y\_current = y + zeros(1,length(x\_current));

% Velocity

vel = sqrt(2\*g\*(h0-h\_current));

% Acceleration for Normal Force

a\_n = vel.^2/R;

% Normal Force

N = g \* cosd(theta\_range) + a\_n;

% G's Experienced

G\_loop = N ./ g;

end

function[h\_current,x\_current,y\_current,vel,G\_trans2] = transition\_line(h0,h,x,y,track\_length,g)

% Parameters

x\_range = (0:track\_length);

% Current Positions

x\_current = x + x\_range;

h\_current = h + zeros(1,length(x\_current));

y\_current = y + zeros(1,length(x\_current));

% Velocity

vel = sqrt(2\*g\*(h0-h\_current));

% Normal Force

N = g;

% G's Experienced

G\_trans2 = zeros(1,length(x\_current));

G\_trans2 = G\_trans2 + N / g;

end

function[h\_current,x\_current,y\_current,vel,G\_trans3] = transition\_line\_reverse(h0,h,x,y,track\_length,g)

% Parameters

x\_range = (0:track\_length);

% Current Positions

x\_current = x - x\_range;

h\_current = h + zeros(1,length(x\_current));

y\_current = y + zeros(1,length(x\_current));

% Velocity

vel = sqrt(2\*g\*(h0-h\_current));

% Normal Force

N = g;

% G's Experienced

G\_trans3 = zeros(1,length(x\_current));

G\_trans3 = G\_trans3 + N / g;

end

function[h\_current,x\_current,y\_current,vel,G\_banked\_lat,G\_banked\_vert,arc\_length] = banked\_turn(h0,h,x,y,bank\_theta,R,g)

% Parameters

theta\_range = (0:180);

arc\_length = (180 / 360) \* 2 \* pi \* R;

% Current Positions

x\_current = x + R \* sind(theta\_range);

y\_current = y - R + R \* cosd(theta\_range);

h\_current = h + zeros(1,length(x\_current));

% Velocity

vel = sqrt(2\*g\*(h0-h\_current));

% Acceleration for Normal Force

a\_n = vel.^2/R;

% Normal Force

N\_vertical = g \* cosd(bank\_theta) + a\_n \* sind(bank\_theta);

N\_lateral = -sind(bank\_theta) \* (g - (a\_n \* cosd(bank\_theta)));

% G's Experienced

G\_banked\_lat = zeros(1,length(x\_current));

G\_banked\_vert = zeros(1,length(x\_current));

G\_banked\_lat = G\_banked\_lat + N\_lateral ./ g;

G\_banked\_vert = G\_banked\_vert + N\_vertical ./ g;

end

function[h\_current,x\_current,y\_current,vel,G\_trans4,arc\_length] = transition\_up\_curve\_reverse(h0,h,x,y,theta\_4,R,g)

% Parameters

theta\_range = (0:theta\_4);

arc\_length = (theta\_4 / 360) \* 2 \* pi \* R;

% Current Positions

h\_current = h + (R - R \* cosd(theta\_range));

x\_current = x - R \* sind(theta\_range);

y\_current = y + zeros(1,length(x\_current));

% Velocity

vel = sqrt(2\*g\*(h0-h\_current));

% Normal Force

N = g \* cosd(theta\_range) + (vel.^2 / R);

% G's Experienced

G\_trans4 = N ./ g;

end

function[h\_current,x\_current,y\_current,vz,vx0,G\_parabola,arc\_length\_parabola] = parabola(h,x,y,vel,~,theta,g)

% Parameters

vx0 = vel \* cosd(theta);

vz0 = vel \* sind(theta);

% Current Positions and Z Velocity Equations

vz1 = @(t) vel \* sind(theta) - (g \* t);

x1 = @(t) x + -vx0 \* t;

y1 = @(t) y + (0 \* t);

h1 = @(t) -0.5 \* g \* t.^2 + (vz0) \* t + h;

% Define Time Vector

t = (0:0.01:10);

% Generate Vectors WRT Time

x\_current = x1(t);

y\_current = y1(t);

h\_current = h1(t);

vz = vz1(t);

% Find Where Track Returns to Initial Height

stop = find(h\_current < h);

stop = stop(1);

t\_stop = t(stop);

% Cut Off Vectors At That Point

x\_current = x\_current(1:stop);

y\_current = y\_current(1:stop);

h\_current = h\_current(1:stop);

vz = vz(1:stop);

% Arc Length

f = @(t) sqrt((vel \* sind(theta) - (g \* t)).^2 + vx0^2 + 1);

arc\_length\_parabola = integral(f,0,t\_stop);

% G's Experienced ... NOTE: Using Kinematic Equations to Generate Parabola

% Makes Our Coaster in "Free Fall"

G\_parabola = zeros(1,stop);

end

function[h\_end,x\_end,y\_end,vel\_end,G\_brake\_vert,G\_brake\_back,track\_length] = braking(h0,h,x,y,g)

% Velocity

vel\_end = (sqrt(2\*g\*(h0-h)):-0.1:0);

% Fix Backwards G's

G\_brake\_back = 2 + zeros(1,length(vel\_end));

G\_brake\_vert = 1 + zeros(1,length(vel\_end));

% Track Length

track\_length = vel\_end.^2./(2\*g\*G\_brake\_back);

% Ending Positions

x\_end = flip(-track\_length + x);

h\_end = h + zeros(1,length(x\_end));

y\_end = y + zeros(1,length(x\_end));

vel\_end = vel\_end + zeros(1,length(x\_end));

end