ASEN 2003-019 Lab 1
Roller Coaster Design

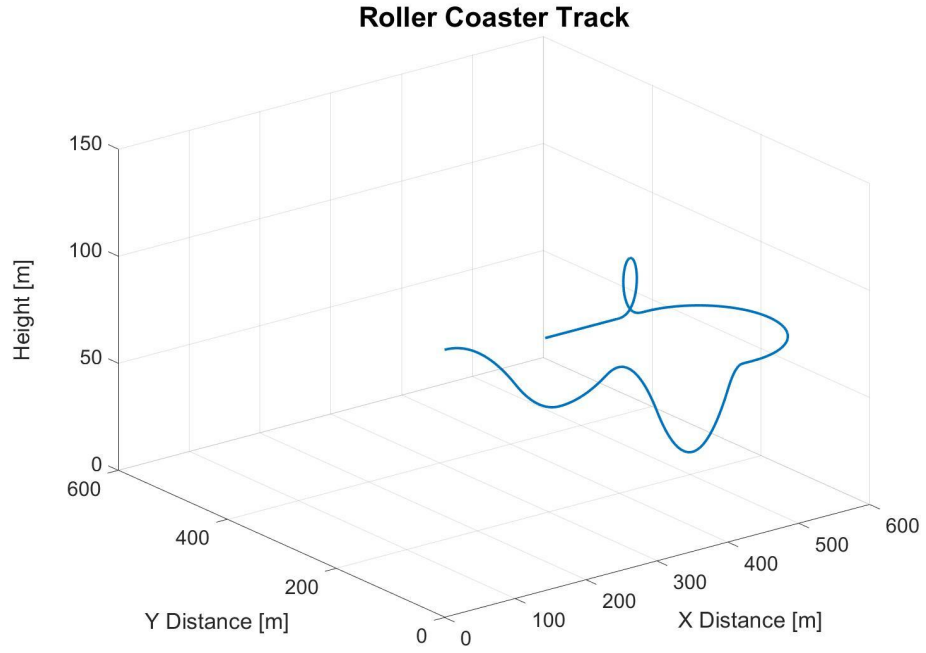
Jack Foster
Andrew Logue
Devon Paris
Siyang Liu

January 29, 2021

[Abstract]The origin of roller coasters can be traced back to ice slides that are constructed in 18th century Russia. The features that make a roller coaster a main attraction of amusement and theme parks are novelty, speed, and G's experienced. In this project, our team tries to analyze the dynamics of typical coaster track elements, design and track assembly, analyze overall track performance and document all the results in a group report.

I. Introduction

The roller coaster design project provided an important introduction to the design process of dynamic systems. The aim for the project was to create a roller coaster track and apply equations of motion in order to inspect key elements of the track's performance. During our investigation, we looked at varying types of track sections and how their profiles relate to designing the ride within allowable g force requirements. In order for the roller coaster to operate safely, it must not exceed the given limit of g force in any direction. For the design, we worked with a few key assumptions in order to make the calculations more practical. The roller coaster car was treated as a point moving through space, friction was considered negligible until the braking section, and the roller coaster started at the top of a 125 meter hill. In addition to the g force requirements, the roller coaster must also have continuous transitions, consist of at least 3 track elements, have a zero g section and banked curve, and will not exceed a track length of 1250 meters.



II. Design

Hill section

During this portion of the track the roller coaster gains the necessary velocity required to experience 0 Gs in the next section. This section consists of three parabolas:

$$z(x) = 125 - 0.0025 \times x^{2} \quad \text{when } 0 \leq x < 100$$

$$z(x) = 0.004 \times (x - 162.5)^{2} + 84.375 \quad \text{when } 100 \leq x < 162.5$$

$$z(x) = 0.002 \times (x - 162.5)^{2} + 84.375 \quad \text{when } 162.5 \leq x < 225$$

These paths of motion give linear functions of velocity. The functions do not have a constant radius so to calculate the Gs experienced during the individual parts of the hill section we must calculate the instantaneous radius for each distance equation to do so we use

$$\rho = [(1 + (dy/dx)^{2})^{3/2}] / (d^{2}y/dy^{2})$$

Once we calculate the instantaneous radius we are able to calculate the Gs from the normal acceleration at a given point.

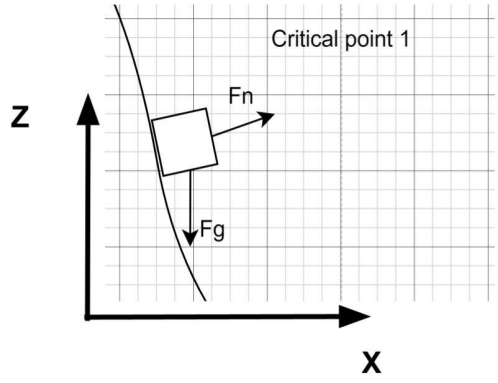


Figure 1.

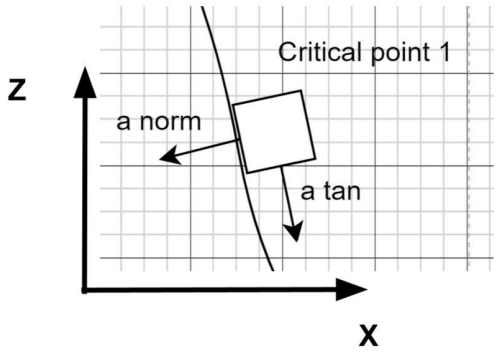


Figure 2.

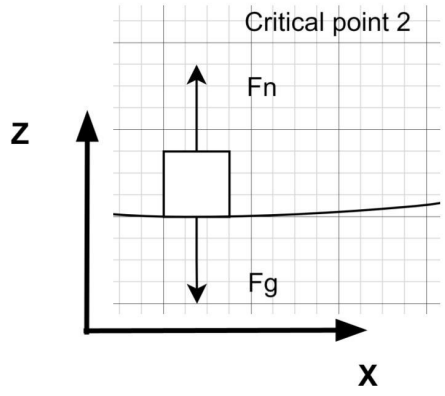


Figure 3.

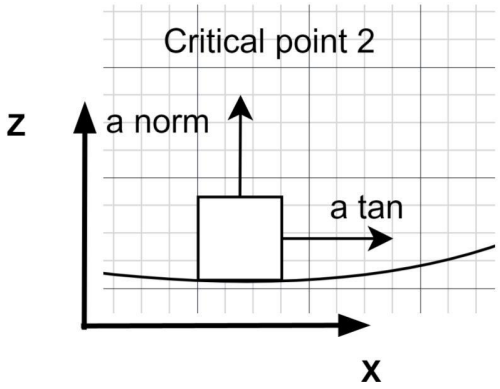


Figure 4.

Zero G section

During this element of the track the roller coaster follows its projectile trajectory at the end of a linear portion of track giving it a constant launch angle of the arctan(.25) . The trajectory was calculated by the parametric equations for projectile motion $< V_x t,\ V_y t + \frac{1}{2} g t^2 >$, using the entry velocity and launch angle. Since the roller coaster would follow the same path even without having a track underneath it, there is no normal force on the coaster by the track, thus the coaster is in free fall resulting in our zero G section.

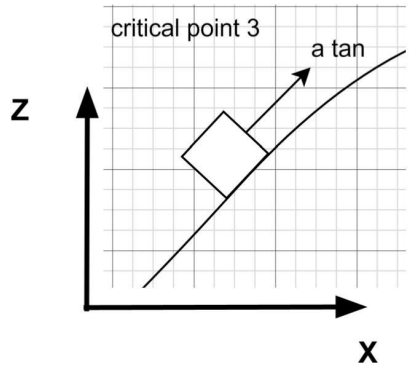


Figure 5.

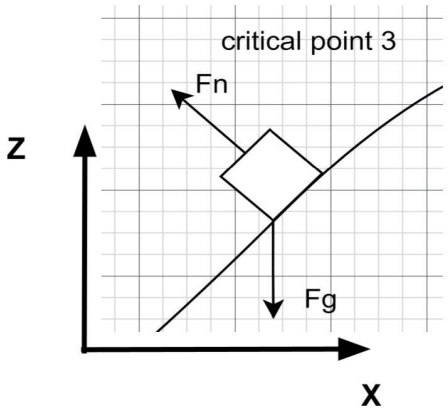


Figure 6.

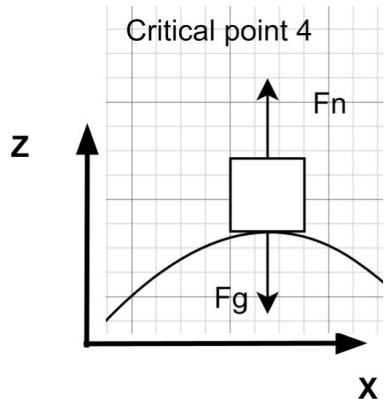


Figure 7.

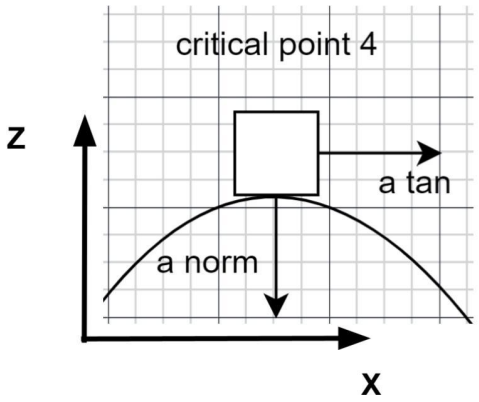


Figure 8.

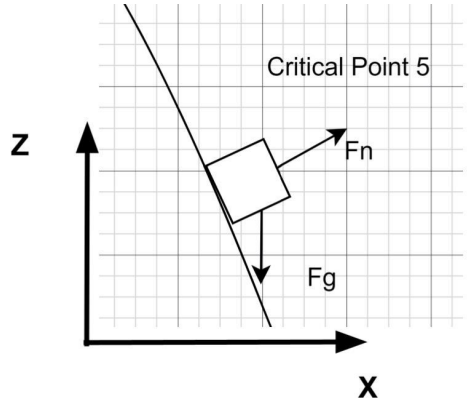


Figure 9.

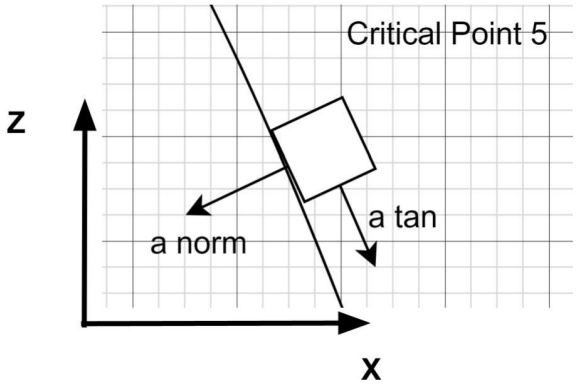


Figure 10.

Banked curve

In the banked curve section, the track was generated using a half circle equation in terms of x and y while holding z to be constant. The bank angle through the curve starts at zero degrees and gradually increases to 25 halfway through the curve, it then resets to zero in the same manner. By calculating the acceleration normal using

kinematic equations of motion, it can then be broken into its left and downward components with geometry and the given angle at any location. From the components of normal acceleration, the g forces in each direction could be found. Since the curve does not change height through its entire section, the tangential acceleration and therefore the tangential g force will be zero throughout.

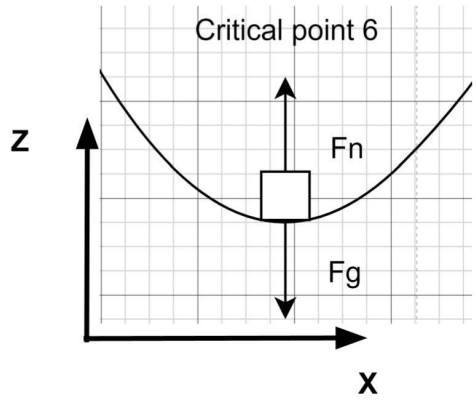


Figure 11.

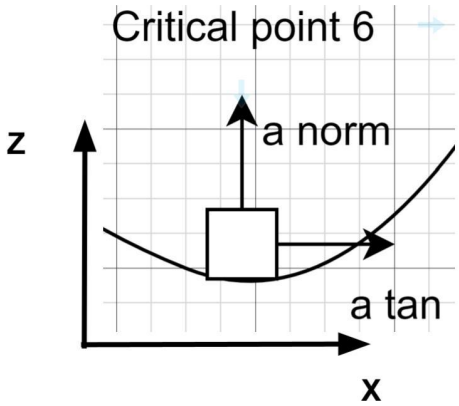


Figure 12.

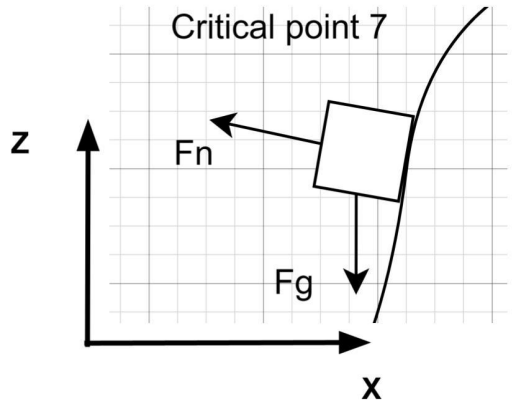


Figure 13.

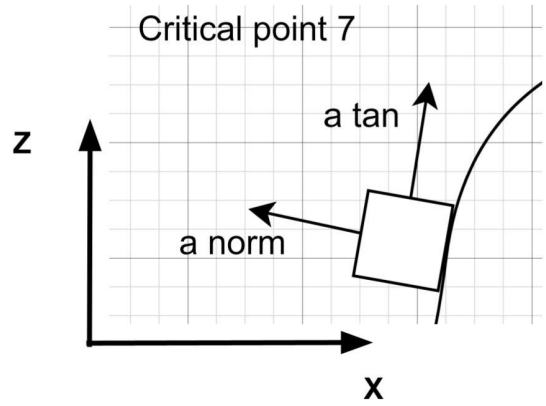


Figure 14.

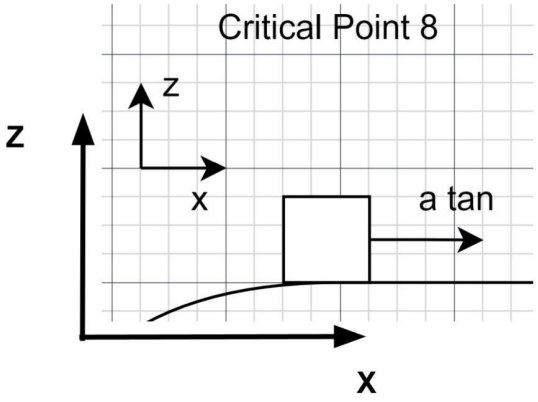


Figure 15.

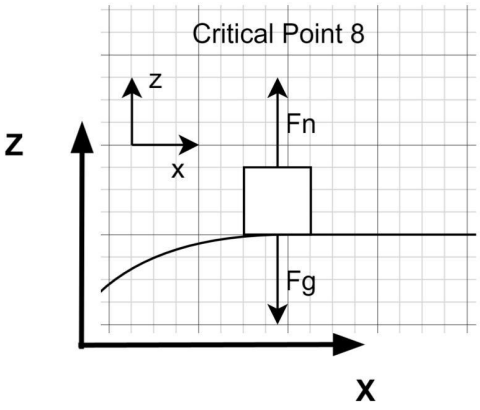


Figure 16.

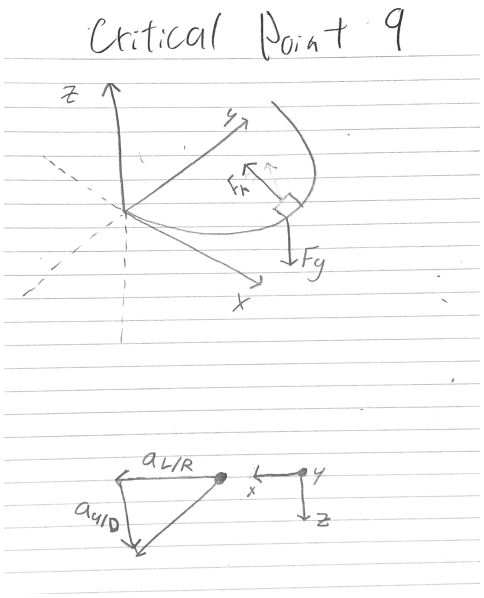


Figure 17.

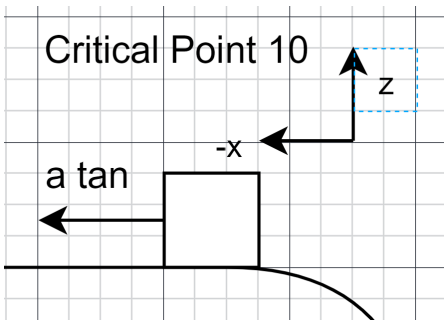


Figure 18.

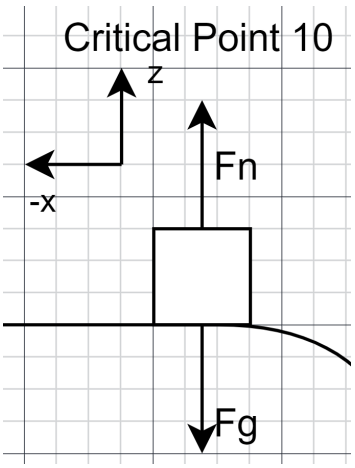


Figure 19.

Loop

A Clothoid loop was used as the loop element of the roller coaster, as it allows for constant centripetal acceleration to be achieved. This is proven by equations 1 and 2 below, and the final loop was determined to have a constant centripetal acceleration of 28.4 meters per second squared based on the 29.2 meters per second entry speed, which resulted in our loop having approximately 3.89 Gs in the upwards direction throughout the loop. Eq. 3 is the full equation using Fresnel integrals in order to graph the Clothoid loop in the x-z plane. The transition from the curve to the loop, and the loop to the braking section is a simple horizontal track, as the beginning and end of the loop has a slope of 0.

$$a_c = \frac{v^2}{r} \qquad \textbf{Eq. 1}$$

$$G = \frac{N}{mg} = \frac{a_c}{g} + 1 \qquad \textbf{Eq. 2}$$

$$X(s) = r * \sqrt{\frac{\pi}{2}} * C(\sqrt{\frac{2}{\pi}} * s); Z(s) = r * \sqrt{\frac{\pi}{2}} * S(\sqrt{\frac{2}{\pi}} * s) \qquad \textbf{Eq. 3}$$

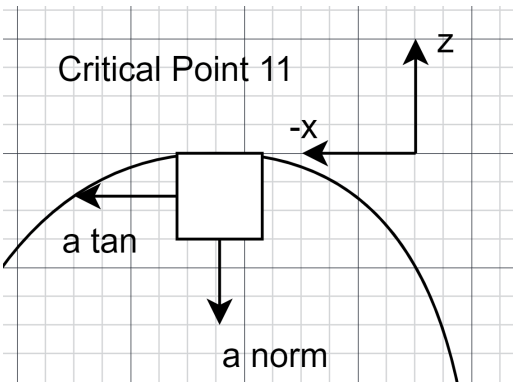


Figure 20.

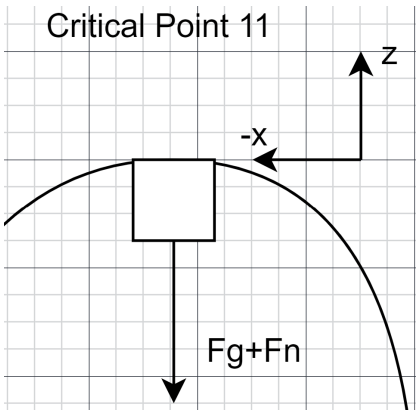


Figure 21.

Breaking section

The braking section consists of a 100 meter long section of straight track where friction is applied in order for the roller coaster to come to a complete stop. By applying the kinematic equations of motion for straight line acceleration, the necessary value of acceleration in order to come to a complete stop by the end of the 100 meters was found. Since the track is simple and straight during this portion, there is no normal acceleration and hence the only g force other than in the direction of motion is equivalent to one g downward. The tangential g force is equal to the braking deceleration divided by g and falls well under the tolerance limit for g forces in the backwards direction.

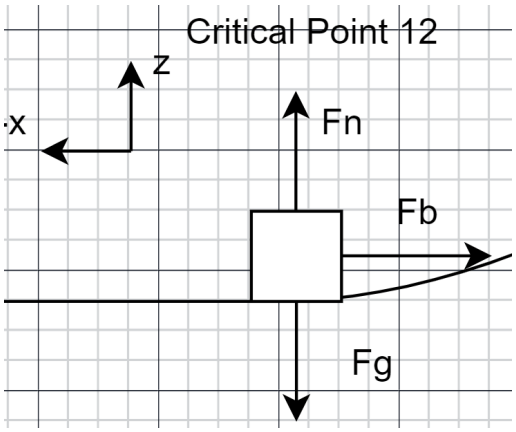


Figure 22.

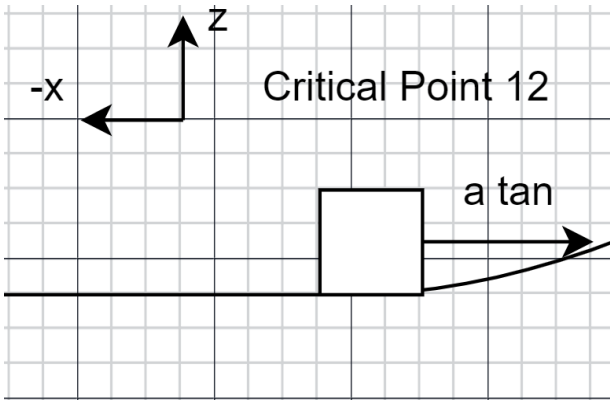


Figure 23.

III. Performance Analysis

The roller coaster created in this lab has a variety of elements and transitions, and reaches a maximum speed of 39.2 meters per second. The maximum Gs experienced on this roller coaster are 3.8922 Gs in the up direction, thus making it a safe yet exhilarating ride. This roller coaster starts at an elevation of 125 meters and finishes at an elevation of approximately 81 meters, so it would be more efficient to lower the entire coaster by 80 meters so that it finishes near ground level, and the single particle rider only needs to climb 45 meters. This coaster would be

practical to build and run assuming this change was made, the track remained frictionless, and that the only riders were individual particles. The only remaining problem with our design is that the 0G section is actually slightly larger than 0G, however it is to a degree that would not be noticeable to the rider. There are no other remaining issues in our design, as the coaster maintains a positive velocity until it comes to a complete stop in the braking section, and all of the transitions are seamless. Therefore, as our designed roller coaster has no dangerous issues, it would be perfectly safe for its intended rider, a particle.

IV. Conclusion & Recommendations

In this project, our team successfully designed a track and analyzed overall track performance. We have gained design experience in particle dynamics, learned to solve problems in dynamics. Additionally, we learned how complex modern roller coasters have become, specifically regarding how the coaster's G forces are minimized and maximized over the entire course in order to increase the rider's thrill. We similarly attempted this by adding elements with both high and low G forces as seen in our 0G section and our ~4G loop. The track meets all the requirements listed in the lab document[1]: circular or parabolic hill, circular or parabolic valley, loop, horizontal banked turn, and breaking section. The track itself measured 964.75 meters, this was calculated by hand and is well below the specified 1250 meter length limit.

In the future, this assignment could be improved by lengthening the timeline over which it is to be completed, and further reviewing expectations, and fundamentals for the lab during lecture.

V. References

[1] Hodgkinson, R, "*ASEN_2003_LAB_1_Document_2021"*. Jan, 2021.

[2] Hodgkinson, R, "*Slide Deck"*. Jan, 2021.

[3] Hodgkinson, R, "*Technical Writing"*. Jan, 2021.

[4] Zyba Ltd, "*latex.codecogs"*. 2021.

[5] Desmos, "*desmos/calculator"*. 2021.

[6] NASA, "*The Clothoid"*. 2004.

[7] Physics Classroom, "*Roller Coasters and Amusement Park Physics"*. 2021.

[8] MathWorks, "*Help Center"*. 2021.

VI. Appendix A

[RollerCoaster.m]----------

% ASEN 2003 Roller Coaster Project Code

% Lab Section 019

```
% Andrew Logue, Devon Paris, Jack Foster, Siyang Liu
% This code will plot the designed path of our group's roller coaster and analyze its
%   dynamic motion as it travels along the track and separate elements. From this analysis, we
will
%   investigate the coaster's velocity and G forces through specifically
%   designed track elements

%% Clearing Commands

clc
clear all
close all

%% Track Section Generation

% Straight section generation
x1 = 0:.5:100;
x2 = 100:.5:162.5;
x3 = 162.5:.5:225;
x4 = 225:.5:230;
x5 = 230:.5:300;
x6 = 300:.5:350;
x7 = 350:.5:400;
x8 = 400:.5:420;
x9 = 420:.5:425;
   % Straight sections prior to banked curve inputs (m)
Sec1 = 125-(0.0025*(x1.^2));
Sec2 = (.004*(x2-162.5).^2)+84.375;
Sec3 = (.002*(x3-162.5).^2)+84.375;
Sec4 = .25*(x4-225)+92.1875;
Sec5 = (6.035*((x5-230)/24.142))-(4.905*((x5-230)/24.142).^2)+93.4375;
```

```
Sec6 = (.009233*(x6-350).^2)+46.6169;
Sec7 = (.01*(x7-350).^2)+46.6169;
Sec8 = (-.025*(x8-420).^2)+81.617;
Sec9 = 81.617*ones(1,length(x9));
   % Straight sections prior to banked curve height outputs (m)
ds1 = .005.*x1;
ds2 = .008.*(x2-162.5);
ds3 = .004.*(x3-162.5);
ds4 = .25*ones(1,length(x4));
ds5 = (6.035/24.142) - ((2*4.905)/24.142).*((x5-230)/24.243);
ds6 = 2*.009233*(x6-350);
ds7 = .02*(x7-350);
ds8 = -.05*(x8-420);
ds9 = 0*ones(1,length(x9));
   % Straight sections prior to banked curve first derivatives dz/dx
dd1 = .005*ones(1,length(x1));
dd2 = .008*ones(1,length(x2));
dd3 = .004*ones(1,length(x3));
dd4 = 0*ones(1,length(x4));
dd5 = (((2*4.905)/24.142)*(1/24.142))*ones(1,length(x5));
dd6 = 2*.009233*ones(1,length(x6));
dd7 = .02*ones(1,length(x7));
dd8 = -.05*ones(1,length(x8));
dd9 = ds9;
   % Straight section prior to banked curve second derivatives d2z/dx2

% Banked curve generation
ycurve = 0:.5:200;
   % Y coordinate inputs (m)
th1 = 0:.125:25;
th2 = 24.875:-.125:0;
```

```
BankTheta = [th1,th2];
   % Bank angle through curve (degrees)
x10 = sqrt(10000-(ycurve-100).^2)+435;
   % X coordinate outputs
Sec10 = 81.617*ones(1,length(x10));
   % Hold constant height (m)
ds10 = -(ycurve-100)./sqrt(-ycurve.^2+200.*ycurve);
   % First derivative dx/dy
dd10 = -10000./((-ycurve.^2+200.*ycurve).^(3/2));
   % Second derivative d2x/dy2

% Clothoid loop generation(constant g force throughout)
[X_arr,Y_arr,Z_arr] = LoopGeneration(30);
   % Generate xyz coordinates (m)
x11 = X_arr;
Sec11 = Z_arr;

% Braking section generation
x12 = 395.3807:-.5:295.3807;
   % 100 m total braking distance
Sec12 = 81.617*ones(1,length(x12));
   % Constant height (m)
ypostcurve = 200*ones(1,length(x12));
   % Constant Y position (no turns, m)

%% Generate Vectors for Plot

fpv = [Sec1,Sec2,Sec3,Sec4,Sec5,Sec6,Sec7,Sec8,Sec9,Sec10,Sec11,Sec12];
   % Vector of Z coordinates (m)
fd = [ds1,ds2,ds3,ds4,ds5,ds6,ds7,ds8,ds9,ds10];
   % Vector of derivatives prior to loop
```

```
fdd = [dd1,dd2,dd3,dd4,dd5,dd6,dd7,dd8,dd9,dd10];
   % Vector of second derivative prior to loop
dy = [ds1,ds2,ds3,ds4,ds5,ds6,ds7,ds8,ds9,zeros(1,401)];
   %  Vector of derivatives dz/dx
xt = [x1,x2,x3,x4,x5,x6,x7,x8,x9,x10,x11,x12];
   % Vector of X coordinates (m)
pc = [Sec1,Sec2,Sec3,Sec4,Sec5,Sec6,Sec7,Sec8,Sec9];
   % Vector of Z coordinates prior to curve (m)
yprecurve = zeros(1,length(pc));
y = [yprecurve,ycurve,ypostcurve];
   % Vector of Y coordinates (m)
y = [y Y_arr];

%% Plotting the Full Track

figure
plot3(xt,y,fpv)
title('Roller Coaster Track')
xlabel('X Distance')
ylabel('Y Distance')
zlabel('Height')
hold on
plot3(00,500,00)
   % Point plotted to keep axes properly scaled

%% Calculations

Vel1 = VelFinder(fpv);
   % Velocity prior to braking section (m/s)
Rho = IR(fpv,fd,fdd);
   % Instantaneous radius full track (m)
```

```
[atan, anorm] = aFinder(dy,Rho,Vel1);
   % Tangential and normal acceleration prior to loop (m/s^2)
[gtan, gnorm] = gFinder(atan, anorm);
   % Tangential and normal g force prior to loop


   % For loop splits g force on the banked curve into left/right and up/down
   % components
c=1;
for i=860:1260
   gnormlr(c) = (gnorm(i)+1).*cosd(BankTheta(c));
   gnormud(c) = gnorm(i).*sind(BankTheta(c));
   c=c+1;
end
braked = -4.25587395;
   % Brake deceleration (m/s^2)
brakevel = sqrt(Vel1(1971)^2+2*braked*(395.3807-x12));
   % Velocity through braking section (m/s)
brakevel(201) = 0;
   % Final velocity of 0 (m/s)
Vel1 = [Vel1(1:1971),brakevel];
   % Vector of velocity through full track (m/s)
gbrake = braked/9.81;
   % G force of braking section
gloop = 3.8922*ones(1,711);
   % Normal g force of loop section
for i=1:710
   atanloop(i) = -1*(Vel1(i+1260)*(Sec11(i+1)-Sec11(i)));
      % Approximation of tangential acceleration from loop (atan = V*dy,
      % dy is approximated as the difference between y coordinates
   gtanloop(i) = atanloop(i)/9.81;
      % Approximation of tangential g force of loop section
```

```
end
gtanloop(711) = 0;
   % Final tangential g force of loop (goes to flat, no tangential a)
gnorm = [gnorm, gloop, zeros(1,201)];
gtan = [gtan, gtanloop, gbrake*ones(1,201)];
table = gPrint(gnorm, gtan, gnormud, gnormlr);
   % gPrint outputs a matrix of 3 rows
      % Row 1: Up (+) and Down (-) g force
      % Row 2: Left (+) and Right (-) g force
      % Row 3: Forward (+) and Backward (-) g force
```

[LoopGeneration.m]----------

```
function [X_arr,Y_arr,Z_arr] = LoopGeneration(r)
%Andrew Logue - 1/27/2021
% graph clothoid loop with constant centripital acceleration
% input: r multiplier
% output: coordinate arrays for plotting
% number upward Gs experienced in loop is 3.8922 Gs

% multiplier
% r = 30; % meters
% acceleration constant
a = 9.81;
% x distance
x6 = 300:.5:340;
% section 6
Sec6 = [];
% arc length
syms s;
% X and Z equations with Fresnel integration
```

```
X = r*sqrt(pi/2)*fresnelc(sqrt(2/pi)*s);
Z = r*sqrt(pi/2)*fresnels(sqrt(2/pi)*s);

% get first two values so while loop works
i = 1;
X_arr(1,i) = r*sqrt(pi/2)*fresnelc(sqrt(2/pi)*((i-1)/200));
Z_arr(1,i) = r*sqrt(pi/2)*fresnels(sqrt(2/pi)*((i-1)/200));
i = i+1;
X_arr(1,i) = r*sqrt(pi/2)*fresnelc(sqrt(2/pi)*((i-1)/200));
Z_arr(1,i) = r*sqrt(pi/2)*fresnels(sqrt(2/pi)*((i-1)/200));

% generate the first half of the loop
while Z_arr(1,i-1) <= Z_arr(1,i)
   i = i+1;
   X_arr(1,i) = r*sqrt(pi/2)*fresnelc(sqrt(2/pi)*((i-1)/200));
   Z_arr(1,i) = r*sqrt(pi/2)*fresnels(sqrt(2/pi)*((i-1)/200));
end

% find X coordinate offset
X_offset = X_arr(1,i);
j = i;
% generate the second half of the loop
while Z_arr(1,i) ~= 0
   i = i+1;
   j = j-1;
   X_arr(1,i) = -X_arr(1,j)+(2*X_offset);
   Z_arr(1,i) = Z_arr(1,j);
end

% set Y_arr to tie into final roller coaster plot
Y_arr = zeros(1,i)+200;
```

```
% offset X_arr and Z_arr to line up with roller coaster plot
X_arr = X_arr+435-(2*X_offset);
X_arr = flip(X_arr);
Z_arr = Z_arr+81.617;
% figure
% plot3(X_arr,Y_arr,Z_arr)

% figure
% fplot(X,Z,[0,3])
% figure
% fplot(X,[0,3])
% figure
% fplot(Z,[0,3])
% figure
% fplot(fresnels(sqrt(2/pi)*x),[-3,3])
% ezplot('x.^3 + y.^3 - 3*x.*y'), axis equal
end
```

[VelFinder.m]----------

```
function [Velocity] = VelFinder(Fcn)
    Velocity = sqrt(2*9.81*(125-Fcn));
       % Calculate velocty from height (m/s)
end
```

[IR.m]----------

```
function [rho] = IR(TrackSec, d1, d2)
   rho = ((1+d1.^2).^1.5)./abs(d2);
      % Calculate the instantaneous radius from predefined derivatives (m)
end
```

[gPrint.m]----------

```
function [gTable] = gPrint(gnorm, gtan, gnormud, gnormlr, gbrake)
   updown = [gnorm(1:859),gnormud,gnorm(1261:2172)];
      % Up and Down g forces
```

```
    leftright = [zeros(1,859),gnormlr,zeros(1,912)];
        % Left and Right g forces
    frontback = [gtan];
        % Front and Back g forces
    gTable = [updown;leftright;frontback];
        % Organize table output
end
```

[gFinder.m]----------

```
function [gtan, gnorm] = gFinder(atan, anorm)
    gtan = atan./9.81;
        % Tangential g force
    gnorm = (anorm./9.81)-1;
        % Normal g force (No accel = -1 g, g downward)
end
```

[aFinder.m]----------

```
function [atan, anorm] = aFinder(dy, rho, Velocity)
    atan = Velocity(1:1260).*dy;
        % Calculate tangential acceleration from velocity and derivatives
        % (m/s^2)
    anorm = (Velocity(1:1260).^2)./rho;
        % Calculate normal acceleration from velocity and IR (m/s^2)
end
```