ASEN 2003 Lab 1: Roller Coaster Design

Section 301 group 17: Ian Faber, Justin Travis, and Niko Von Unger

```
clc; clear; close all;

% Constants
h0 = 125;
g = 9.81;
posStart = [0, 0, h0];

% Start cone
nCone = 100;
s = linspace(0*pi, 4.5*pi, 100)' - pi/2;
rCone = 3*s + 10;
cone_bank = 0*pi/180;

pathCone = posStart + [rCone.*cos(s), rCone.*sin(s), (-3*(s
+pi/2))]; %x, y, z
pathConeVel = [-3*s.*sin(s) + 3*cos(s) - 10*sin(s), 3*s.*cos(s) +
 3*sin(s) + 10*cos(s), -3*ones(length(s),1)];
pathConeNorm = sqrt(pathConeVel(:,1).^2 + pathConeVel(:,2).^2 +
 pathConeVel(:,3).^2);

distanceCone = cumtrapz(s, pathConeNorm);

mag = sqrt(2*g*(h0-pathCone(:,3)));

velCone = mag.*(pathConeVel./pathConeNorm); % vx, vy, vz

phi = asin(abs(velCone(:,3)./mag));

gsCartesian = [((mag.^2).*cos(s))./
(rCone.*g.*cos(phi)*cos(cone_bank)), ((mag.^2).*sin(s))./
(rCone.*g.*cos(phi)), 1./(cos(phi)*cos(cone_bank))];

for k = 1:length(s)
    gsCone(k,:) = (rotate(gsCartesian(k,:)', s(k)*180/pi, phi(k),
 cone_bank))'; % Front/back, Left/right, Up/down
end


% % Transition cone to level
%
% s = ((2*pi) - atan(1/3):pi/200:2*pi)';
% %s = (2*pi:-pi/200:(2*pi) - atan(1/3))';
% rTrans1 = 40;
%
% pathTrans1 = [0*s, (rTrans1).*sin(s + pi), (rTrans1).*cos(s + pi) +
 rTrans1] + pathCone(end,:);
%
% pathTrans1Vel = [0*s, rTrans1.*cos(s + pi), -rTrans1.*sin(s + pi)];
% pathTrans1Norm = sqrt(0 + (rTrans1.*cos(s + pi)).^2 + (-
rTrans1.*sin(s + pi)));
%
% distanceTrans1 = cumtrapz(-s, pathTrans1Norm);
%
% pathCoaster = [pathCone; pathTrans1];
% distanceCoaster = [distanceCone; distanceCone(end) +
 distanceTrans1];

nTrans1 = 100;
rTrans1 = 5;

[~, ~, ~, pathTrans1, distanceTrans1, velOutTrans1, gsTrans1] =
 transition(velCone(end,:), pathCone(end,:), [0, sqrt(2*g*(h0 -
 (pathCone(end, 3) + rTrans1))), 0], nTrans1, -1, h0, nTrans1, NaN,
 NaN, NaN);

% Straight line after transition
nStraight1 = 100;
dStraight1 = 25;
s = (linspace(0, dStraight1, nTrans1))';

pathStraight1 = pathTrans1(end,:) + [zeros(length(s),1), s,
 zeros(length(s),1)];
distanceStraight1 = s;
velOutStraight1 = velOutTrans1;
gsStraight1 = [zeros(nStraight1, 1), zeros(nStraight1, 1),
 ones(nStraight1, 1)];

% Loop
nLoop = 100;
rLoop = 20;

[~, pathLoop, distanceLoop, ~, velOutLoop, gsLoop] =
 loop(velOutStraight1, pathStraight1(end,:), nLoop, 1, rLoop, h0, NaN,
 NaN, NaN);

% Banked turn
nTurn1 = 100;
rTurn1 = 25;
turn1BankAngle = 20*pi/180;

[pathTurn1, distanceTurn1, velOutTurn1, gsTurn1] =
 banked_turn(velOutLoop, pathLoop(end,:), nTurn1, rTurn1, -1,
 turn1BankAngle, h0, NaN);

% Drop
hDrop = 40;
dDrop = 150;
nDrop = 100;

a = 16;

s = (linspace(0, dDrop, nDrop))';
```

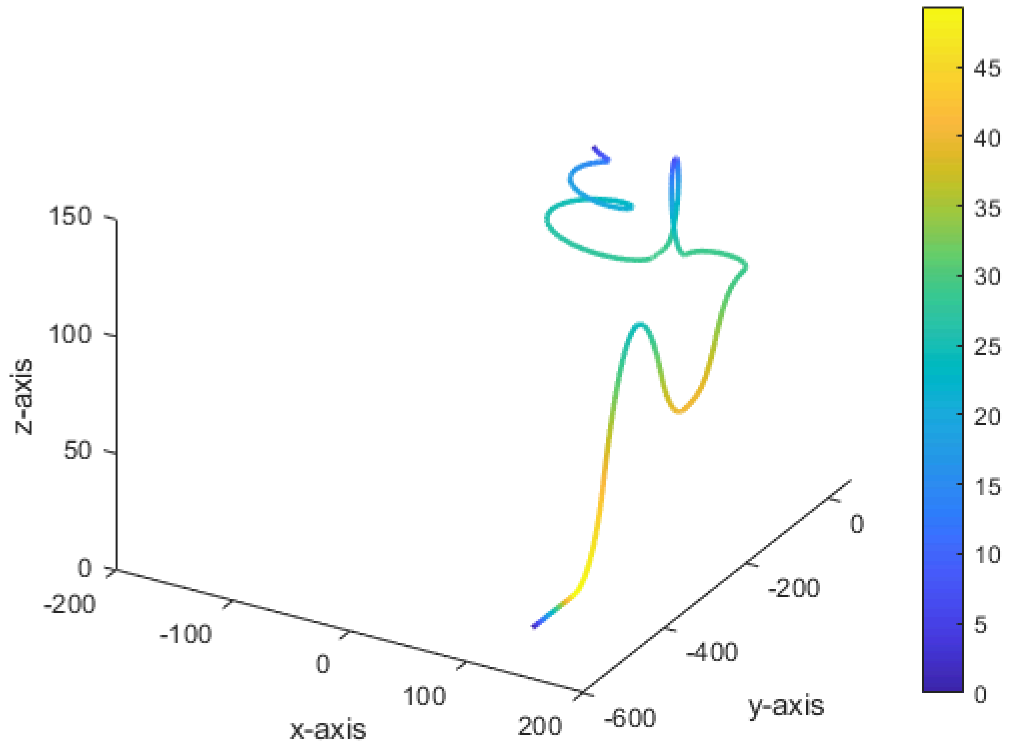
```
pathDrop = pathTurn1(end,:) + [zeros(length(s),1), -s, 0.5 + hDrop *
 (1./(1+exp((s-70)/a))) - hDrop];

pathDropVel = [zeros(length(s),1), -1*ones(length(s),1),
 hDrop*exp((s-70)./a) ./ (a*(1+exp((s-70)./a)).^2)];
thetaDrop = atan2(abs(pathDropVel(:,2)), abs(pathDropVel(:,3)));

distanceDrop = cumtrapz(s, sqrt(0 + 1 + ( (hDrop*exp((s-70)./a)) ./
 (a*(1+exp((s-70)./a)).^2) ).^2));

rDrop = ((1+((-hDrop*exp((s-70)./a))./(a*(1+exp((s-70)./
a)).^2)).^2).^(3/2))./((-hDrop*(exp((s-70)./a) - exp(2*(s-70)./a)))./
(a^2*(1+exp((s-70)./a)).^3));

mag = sqrt(2*g*(h0-pathDrop(:,3)));

gsDrop = [sin(thetaDrop), zeros(length(s),1), cos(thetaDrop) +
 (mag.^2)./(rDrop*g)];

% Transition into parabola
nTrans2 = 100;
rTrans2 = 75;
vParabolaStart = [0, -10, 10];

[~, ~, ~, pathTrans2, distanceTrans2, velOutTrans2, gsTrans2] =
 transition(pathDropVel(end,:), pathDrop(end,:), vParabolaStart,
 rTrans2, 1, h0, nTrans2, NaN, NaN, NaN);

% Zero-G parabola
nParabola = 100;
dParabola = 143;
aParabola = -9.81;

[pathParabola, distanceParabola, velOutParabola, gsParabola] =
 parabola(velOutTrans2, pathTrans2(end,:), nParabola, aParabola,
 dParabola, h0, NaN, NaN);

% Transition out of parabola
nTrans3 = 100;
rTrans3 = 87.6;
vParabolaEnd = [0, -sqrt(2*g*(h0-(pathParabola(end,3)+rTrans3))), 0];

[~, ~, ~, pathTrans3, distanceTrans3, velOutTrans3, gsTrans3] =
 transition(velOutParabola, pathParabola(end,:), vParabolaEnd,
 rTrans3, 1, h0, nTrans3, NaN, NaN, NaN);

pathTrans3(end,:);

% % Turn after parabola
% nTurn2 = 100;
% rTurn2 = 50;
% turn2BankAngle = 50*pi/180;
%
```

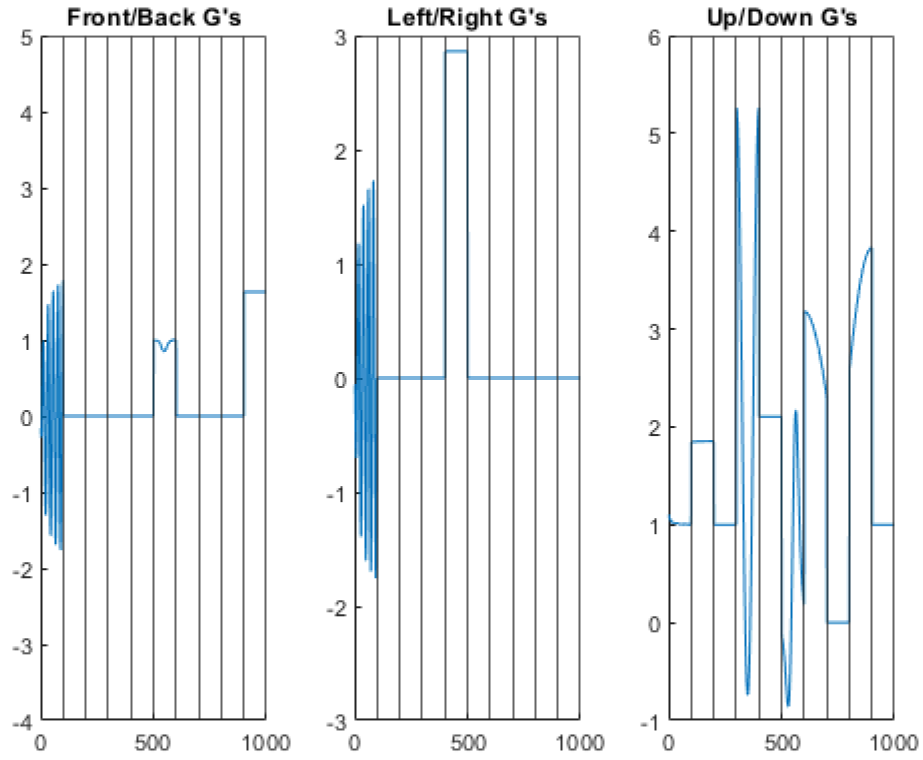
```
% [pathTurn2, distanceTurn2, velOutTurn2, gsTurn2] =
 banked_turn(velOutTrans3, pathTrans3(end,:), nTurn2, rTurn2, -1,
 turn2BankAngle, h0, NaN);
%
% pathTurn2 = pathTurn2 - [2*rTurn2, 0, 0];

% Braking section

nBraking = 100;
dBraking = 100;
s = (linspace(0, dBraking, nBraking))';

aBrake = (velOutTrans3(2).^2)/dBraking;

pathBraking = pathTrans3(end,:) + [zeros(length(s),1), -s,
 zeros(length(s),1)];
distanceBraking = s;
gsBraking = [ones(nBraking, 1)*a/g, zeros(nStraight1, 1),
 ones(nStraight1, 1)];


% Final visualization
pathCoaster = [pathCone; pathTrans1; pathStraight1; pathLoop;
 pathTurn1; pathDrop; pathTrans2; pathParabola; pathTrans3;
 pathBraking];
distanceCoaster = [distanceCone;
                   distanceCone(end) + distanceTrans1;
                   distanceCone(end) + distanceTrans1 +
 distanceStraight1;
                   distanceCone(end) + distanceTrans1(end) +
 distanceStraight1(end) + distanceLoop;
                   distanceCone(end) + distanceTrans1(end) +
 distanceStraight1(end) + distanceLoop(end) + distanceTurn1;
                   distanceCone(end) + distanceTrans1(end) +
 distanceStraight1(end) + distanceLoop(end) + distanceTurn1(end) +
 distanceDrop;
                   distanceCone(end) + distanceTrans1(end) +
 distanceStraight1(end) + distanceLoop(end) + distanceTurn1(end) +
 distanceDrop(end) + distanceTrans2;
                   distanceCone(end) + distanceTrans1(end) +
 distanceStraight1(end) + distanceLoop(end) + distanceTurn1(end) +
 distanceDrop(end) + distanceTrans2(end) + distanceParabola;
                   distanceCone(end) + distanceTrans1(end) +
 distanceStraight1(end) + distanceLoop(end) + distanceTurn1(end) +
 distanceDrop(end) + distanceTrans2(end) + distanceParabola(end) +
 distanceTrans3;
                   distanceCone(end) + distanceTrans1(end) +
 distanceStraight1(end) + distanceLoop(end) + distanceTurn1(end) +
 distanceDrop(end) + distanceTrans2(end) + distanceParabola(end) +
 distanceTrans3(end);
                   distanceCone(end) + distanceTrans1(end) +
 distanceStraight1(end) + distanceLoop(end) + distanceTurn1(end) +
 distanceDrop(end) + distanceTrans2(end) + distanceParabola(end) +
 distanceTrans3(end) + distanceBraking;
```

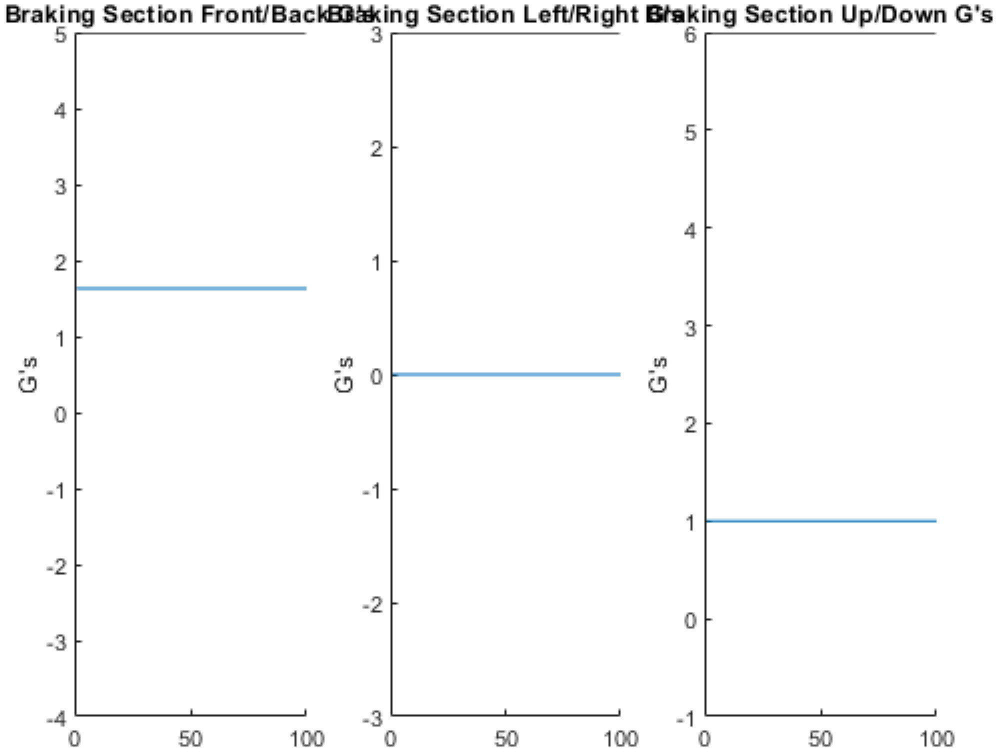
```
                  ];
gsCoaster = [gsCone; gsTrans1; gsStraight1; gsLoop; gsTurn1; gsDrop;
 gsTrans2; gsParabola; gsTrans3; gsBraking];

numSections = length(pathCoaster)/100;
pathLength = distanceCoaster(end)
pathVelocity = sqrt(2*g*(h0 - pathCoaster(:,3)));

pathVelocity((end-99):end) = pathVelocity((end-99):end).*((nBraking-
s)/nBraking);

figure
color_line3d(pathVelocity, pathCoaster(:,1), pathCoaster(:,2),
 pathCoaster(:,3));
xlim([-200 200])
xlabel("x-axis")
ylabel("y-axis")
zlabel("z-axis")
view([30, 35]);

figure

subplot(1,3,1)
hold on
title("Front/Back G's")
plot(gsCoaster(:,1));

for k = 1:numSections
   xline(k*100);
end

yline(5)
yline(-4)
hold off

subplot(1,3,2)
hold on
title("Left/Right G's")

plot(gsCoaster(:,2));
for k = 1:numSections
   xline(k*100);
end

yline(3)
yline(-3)
hold off

subplot(1,3,3)
hold on
title("Up/Down G's")
plot(gsCoaster(:,3));

for k = 1:numSections
```

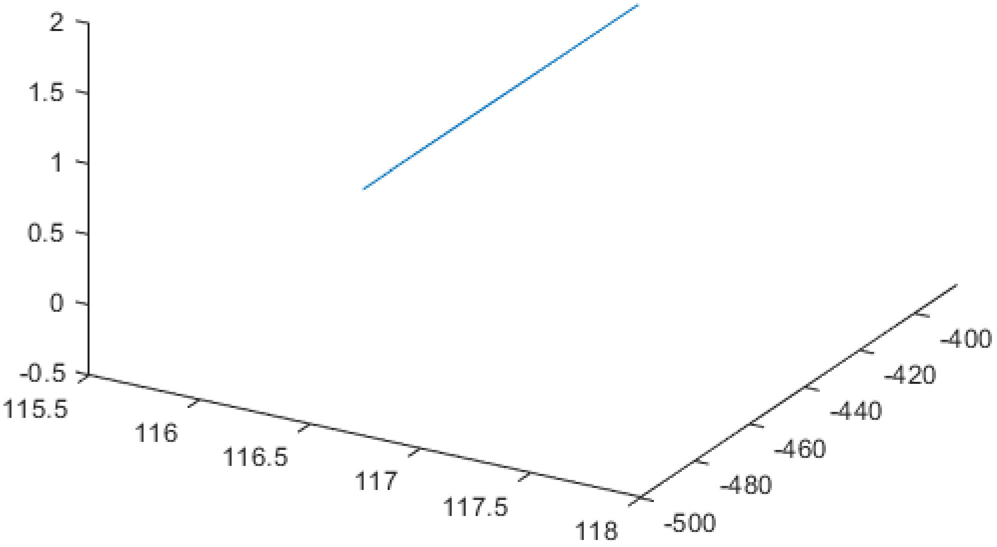
```
    xline(k*100);
end

yline(6)
yline(-1)

figure

subplot(1,3,1)
hold on;
title("Braking Section Front/Back G's")
plot(gsBraking(:,1))
yline(5)
yline(-4)
ylabel("G's")
hold off;

subplot(1,3,2)
hold on;
title("Braking Section Left/Right G's")
plot(gsBraking(:,2))
yline(3)
yline(-3)
ylabel("G's")
hold off;

subplot(1,3,3)
hold on;
title("Braking Section Up/Down G's");
plot(gsBraking(:,3))
yline(6)
yline(-1)
ylabel("G's")
hold off;

figure
plot3(pathBraking(:,1), pathBraking(:,2), pathBraking(:,3))
view([30 35])
```

```
pathLength =

    1.1899e+03
```


150
100
50
0
z-axis
-200
-100
0
100
200
x-axis
-600
-400
-200
0
y-axis
45
40
35
30
25
20
15
10
5
0


Front/Back G's
5
4
3
2
1
0
-1
-2
-3
-4
0
500
1000
Left/Right G's
3
2
1
0
-1
-2
-3
0
500
1000
Up/Down G's
6
5
4
3
2
1
0
-1
0
500
1000


Braking Section Front/Back G's
Braking Section Left/Right G's
Braking Section Up/Down G's
G's
G's
G's
5
4
3
2
1
0
-1
-2
-3
-4
3
2
1
0
-1
-2
-3
6
5
4
3
2
1
0
-1
0
50
100
0
50
100
0
50
100


2
1.5
1
0.5
0
-0.5
115.5
116
116.5
117
117.5
118
-500
-480
-460
-440
-420
-400

Published with MATLAB® R2021a