International Baccalaureate

Internal Assesment

Mathematics: Application and Interpretation

Investigating Roller coasters using Bézier Curves

contact: research[at]aryanmukherjee.com

Contents

Figures

1 Introduction

When I was a child, my mother would never let me ride a roller coaster alone, despite my repeated protests for independence, citing how these roller coasters are tested for safety, I was stumped when she said 'Prove it.'

While the days of careless childhood innocence is gone, and some mathematical knowledge acquired, I would like to revisit the problem of modelling roller coasters using Bézier curves, with the aim of using it as a metric to analyse the physiological effects of a rollercoaster on the human body and consequently its safety.

The roller coaster of choice is one from my birthplace in Hyderabad, called Wonderla. However, before modelling the roller coaster with Bézier curves, a literature review was conducted to gain a thorough understanding of the mathematics that underpin the modelling of a rollercoaster.

2 Literature Review

2.1 Parametric Equations

Bézier curves equations are a type of Parametric equation. Parametric equations are an equation that use a parameter, in this case, t, and in which dependent variables are defined as continuous functions of the parameter and are not dependent on another existing variable (Parametric equation).

Using such models is advantageous in this case as it provides more flexibility over traditional rectangular equations without resorting to piecewise functions. Using parametric equations offers a homogenous approach to analysing the functions which make them highly extensible for analysing complex shapes, as found in rollercoasters.

2.2 Bézier Curves

Bézier curves are a subset of parametric equations, which utilize an ordered set of discrete control points, which determine its shape. Although, except the first and last control point, the curve does not necessarily pass through them.
Formal Definition:
Given a set of $n+1$ control points, $P_0, P_1 \cdots P_n \in \mathbb{Z}$, and the normalized parameter, t, for $0 \leq t \leq 1$, we can define a Bézier curve $P(t)$ of degree n as

$$P(t) = \sum_{i=0}^{n} B_{n,i}(t) P_i \tag{1}$$

where $B_{n,0}, B_{n,1} \cdots B_{n,n}$ are the Bernstein polynomials in the form, for a particular t

$$B_{n,i}(t) = \binom{n}{i} t^i (1-t)^{n-i} \tag{2}$$

The following figure represents how the values of the Bernstein polynomial, of 3^{rd} order, changes for $0 \leq t \leq 1$. At any given point, the sum of all the Bernstein Polynomials $\left(\sum_{i=0}^{n} B_{n,i}(t)\right)$ is one. This is also known as the partition of unity (Wolfram Research).

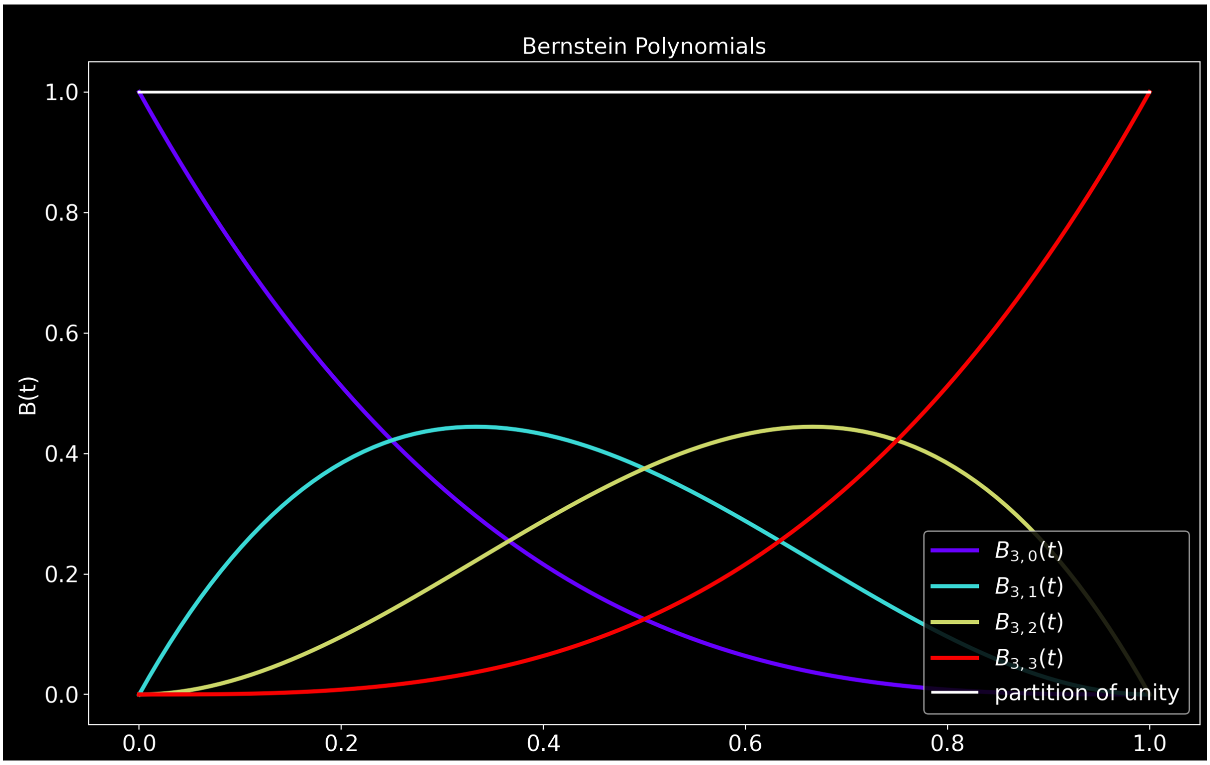

Figure 1: The value of the Bernstein Polynomials for $0 \leq t \leq 1$ and degree 3

Therefore, if P_i is the $(i+1)^{th}$ control point with the coordinates $\begin{bmatrix} x_i \\ y_i \end{bmatrix}$, the generalized form of a Bézier curve will be as follows:

$$\begin{aligned} P(t) = \binom{n}{0}(1-t)^n t^0 \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} + \binom{n}{1}(1-t)^{n-1} t^1 \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} \\ + \binom{n}{2}(1-t)^{n-2} t^2 \begin{bmatrix} x_2 \\ y_2 \end{bmatrix} + \cdots + \binom{n}{n}(1-t)^0 t^n \begin{bmatrix} x_n \\ y_n \end{bmatrix} \end{aligned} \tag{3}$$

where,

$$\binom{n}{r} = \frac{n!}{(n-r)!r!} \tag{4}$$

Applying $n = 3$, we get an arbitrary 3^{rd} order Bézier curve as such:

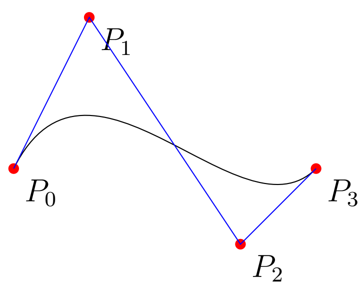

Figure 2: An example of a 3^{rd} order Bézier curve with P_0 and P_3 as start and end points

Matrix Representation of a Bézier Curve:
A Bézier curve can also be represented as a set of matrices. This is useful as this report aims to fit Bézier curves using the method of least squares.

Let,

$$P(t) = \mathbf{TMP} \tag{5}$$

where, T is a power series of t, of order n, given $0 \leq t \leq 1$.

$$\mathbf{T} = \begin{bmatrix} t^n & t^{n-1} & \cdots & t & 1 \end{bmatrix}_{1\times(n+1)} \tag{6}$$

M is the coefficient matrix of the n^{th} order Bézier curve. (Generalized to the n^{th} degree by the IBDP Candidate from 'Theorem 1.' (Kiliçoglu and Şenyurt)).

$$\mathbf{M} = \begin{bmatrix} (-1)^n\binom{n}{0}\binom{n}{n} & (-1)^{n-1}\binom{n}{1}\binom{n-1}{n-1} & \cdots & (-1)^1\binom{n}{n-1}\binom{1}{1} & \binom{n}{n}\binom{0}{0} \\ (-1)^{n-1}\binom{n}{0}\binom{n}{n-1} & (-1)^{n-2}\binom{n}{1}\binom{n-1}{n-2} & \cdots & \binom{n}{n-1}\binom{1}{0} & 0 \\ (-1)^{n-2}\binom{n}{0}\binom{n}{n-2} & (-1)^{n-3}\binom{n}{1}\binom{n-1}{n-3} & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \binom{n}{0}\binom{n}{0} & 0 & \cdots & 0 & 0 \end{bmatrix}_{(n+1)\times(n+1)} \tag{7}$$

and, **P** is the column matrix, consisting of the control points

$$\mathbf{P} = \begin{bmatrix} P_0 \\ P_1 \\ P_2 \\ \vdots \\ P_n \end{bmatrix}_{(n+1)\times 1} \tag{8}$$

Sixth order Bézier Curve:
Sixth order Bézier curves as they are used to model the track or the position. Subsequently, taking its derivatives yields non-degenerate velocity, acceleration, jerk, snap, crackle and jounce, 'each triggering a biomechanical response in the human body' (Eager et al.). While evaluating all of these is beyond the scope of this report, it provides an easily extensible way to do so, further showcasing the appropriateness of using Bézier curves to model roller coasters

Using Equation (5), the 6^{th} order Bézier curve becomes:

$$P(t) = \left(\begin{bmatrix} t^6 \\ t^5 \\ t^4 \\ t^3 \\ t^2 \\ t \\ 1 \end{bmatrix}_{7\times 1}^T \times \begin{bmatrix} 1 & -6 & 15 & -20 & 15 & -6 & 1 \\ -6 & 30 & -60 & 60 & -30 & 6 & 0 \\ 15 & -60 & 90 & -60 & 15 & 0 & 0 \\ -20 & 60 & -60 & 20 & 0 & 0 & 0 \\ 15 & -30 & 15 & 0 & 0 & 0 & 0 \\ -6 & 6 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}_{7\times 7} \right) \times \begin{bmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \\ P_4 \\ P_5 \\ P_6 \end{bmatrix}_{7\times 1} \tag{9}$$

After simplification, Equation (9) becomes,

$$
\begin{aligned}
P(t) = &(1-t)^6 P_0 + 6t(1-t)^5 P_1 \\
&+ 15t^2(1-t)^4 P_2 + 20t^3(1-t)^3 P_3 \\
&+ 15t^4(1-t)^2 P_4 + 6t^5(1-t) P_5 + t^6 P_6
\end{aligned} \tag{10}
$$

With $n = 6$, the Bernstein coefficients take the following form.

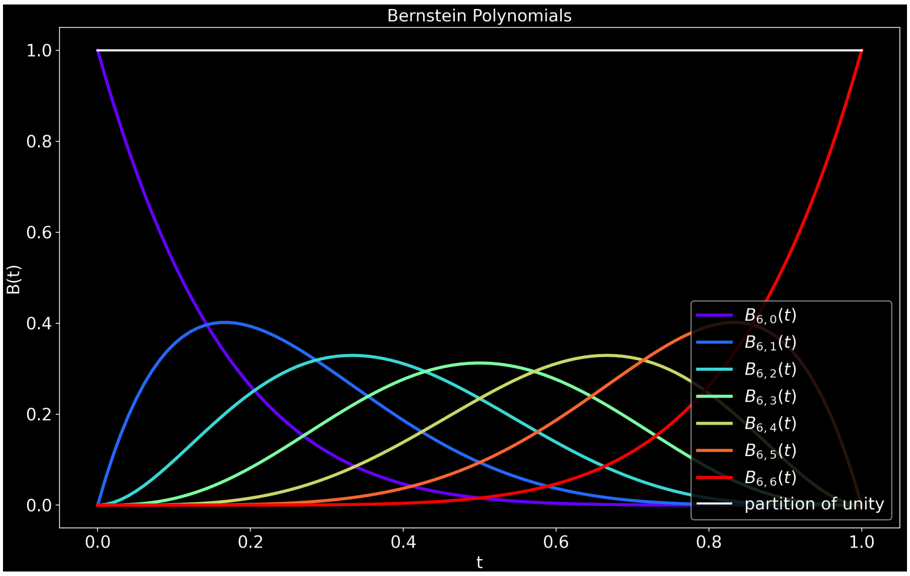

Figure 3: Bernstein Polynomials

Figure (3) helps to visualize the nature of the curves. If the control points are though of as vectors, and the Bernstein biases are applied respectively, we achieve the result depicted by Figure 4.

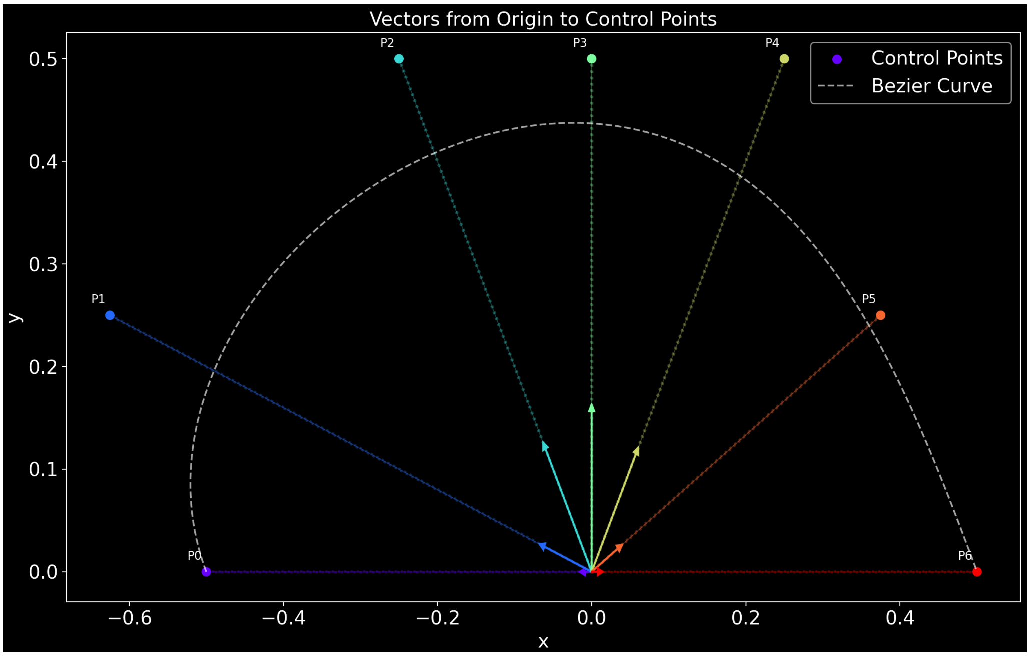

Figure 4: Bézier curve in Bernstein form

Following the law of vector addition, if each of these biased vectors are joined from head to tail, we can see that they trace out the Bézier curve, which is shown in Figure (5) as a dotted curve from P_0 to P_6, as the value of the parameter t ranges from 0 to 1.

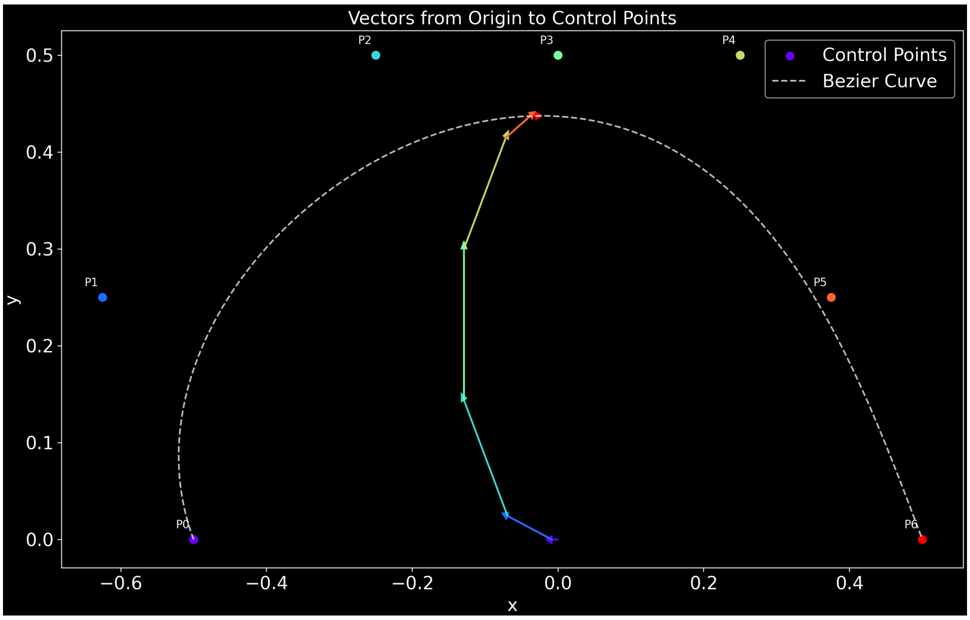

Figure 5: Vector sum of the biased control points

2.3 Properties of Bézier curves

Derivatives of a Bézier curve

It is quite easy to analytically create a general formula for the derivative of a Bézier curve. Let $Q(t)$ represent the first order derivative of a Bézier curve $P(t)$. Therefore

$$P'(t) = Q(t) = \sum_{i=0}^{n-1} B_{n-1,i}(t) \times n\,(P_{i+1} - P_i) \tag{11}$$

Since the first order derivative of an n^{th} order Bézier curve is a $(n-1)^{th}$ order Bézier curve with control points $Q_0 = n(P_1 - P_0), Q_1 = n(P_2 - P_1), \ldots, Q_{n-1} = n(P_n - P_{n-1})$, $P'(t)$, the first order derivative, can be represented as another Bézier curve $Q(t)$ of order $(n-1)$ with control points $Q_0, Q_1 \cdots Q_{n-1}$, where $Q_i = n(P_i - P_{i-1})$ (Kamermans).

First order derivative (Velocity)

Substituting $n = 6$, we get

$$\begin{aligned} P'(t) = Q(t) = (1-t)^5 Q_0 + 5t(1-t)^4 Q_1 + 10t^2(1-t)^3 Q_2 \\ + 10t^3(1-t)^2 Q_3 + 5t^4(1-t) Q_4 + t^5 Q_5 \end{aligned} \tag{12}$$

which is a Bézier curve of order 5.

Rewriting in terms of P,

$$\begin{aligned} P'(t) = Q(t) = 6((1-t)^5(P_1 - P_0) + 5t(1-t)^4(P_2 - P_1) + 10t^2(1-t)^3(P_3 - P_2) \\ + 10t^3(1-t)^2(P_4 - P_1) + 5t^4(1-t)(P_5 - P_4) + t^5(P_6 - P_5)) \end{aligned} \tag{13}$$

Similarly, this can be done for the second, third and any n^{th} order derivative. However, for the scope of this report, only up to the 3^{rd} order is shown.

Second order derivative (Acceleration)

$$\begin{aligned} P''(t) = R(t) = (1-t)^4 R_0 + 4t(1-t)^3 R_1 + 6t^2(1-t)^2 R_2 \\ + 4t^3(1-t)^1 R_3 + t^4 R_4 \end{aligned} \tag{14}$$

where, $R(t)$ is a 4^{th} order Bézier curve with control points $R_0, R_1 \cdots R_4$, where $R_i = (n-1)(Q_i - Q_{i-1})$

Third order derivative (Jerk)

$$P'''(t) = S(t) = (1-t)^3 S_0 + 3t(1-t)^2 S_1 + 3t^2(1-t)^1 S_2 + t^3 S_3 \tag{15}$$

where, $S(t)$ is a 3^{rd} order Bézier curve with control points S_0, S_1, S_2, S_3, where $S_i = (n-2)(R_i - R_{i-1})$

This showcases the recursive way in which higher order derivatives of the original Bézier curve can be calculated, which is very easy to algorithmically compute.

3 Methodology

To gather the data for velocity, acceleration and jerk, a Bézier curve will first need to be fit onto the roller coaster.

Curve Fitting:
Let the data, be an ordered set, $H_0, H_1 \cdots H_m$. Note that the number of input points to fit must always be greater than the order of the Bézier curve.

Therefore, if $\mathbf{H}$ represents the array of input points,

$$\mathbf{H} = \begin{bmatrix} H_0 \\ H_1 \\ H_2 \\ \vdots \\ H_m \end{bmatrix}_{(m+1)\times 1} = \left[\begin{array}{c|c} x_0 & y_0 \\ x_1 & y_1 \\ x_2 & y_2 \\ \vdots & \vdots \\ x_m & y_m \end{array}\right]_{(m+1)\times 1} \tag{16}$$

such that $H_i = \{x_i, y_i\}$, representing a point in the Cartesian space, with coordinates of the point that we are trying to fit and $m > n$, where m is the number of data points to be fit and n is the order of the Bézier curve.

Due to the affine invariance property of the Bézier curves, and the parametrization of Bézier curves, to apply the least squares method, $\mathbf{x}$ values are considered individually from $\mathbf{y}$. This does not effect the solution as the Bézier curve is a parametric equation and the control points are already in $\{x, y\}$ form.

Therefore,

$$\mathbf{x} = \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_m \end{bmatrix}_{(m+1)\times 1} \qquad \mathbf{y} = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_m \end{bmatrix}_{(m+1)\times 1}$$

From Equation 1, $P(t) = B(t)P$, where $B(t) = \mathbf{TM}$ (from Equation 4). Therefore to form the error function that we need to minimize, a Vandermonde matrix ($\mathbb{T} = \mathbb{T}(t_0, t_1, \ldots, t_m)$) was created to match each data point.

$$\mathbb{T} = \begin{bmatrix} t_1^6 & t_1^5 & t_1^4 & t_1^3 & t_1^2 & t_1^1 & 1 \\ t_2^6 & t_2^5 & t_2^4 & t_2^3 & t_2^2 & t_2^1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ t_m^6 & t_m^5 & t_m^4 & t_m^3 & t_m^2 & t_m^1 & 1 \end{bmatrix}_{(m+1)\times 7} \tag{17}$$

Then, to match the stoke segment from the input data to the Bézier curve, we use the path length of the segment S_i at the i_{th} point in the stoke as:

$$S_i = |H_i - H_{i-1}| \tag{18}$$

where, S_0 is 0

Therefore, we can define the vector that maps each point on the stoke to the corresponding segment in the Bézier curve

$$t_i = \frac{|S_i - S_{i-1}|}{\sum_{j=1}^{n+1} |S_j - S_{j-1}|} \tag{19}$$

Then the error function for the $\mathbf{y}$ values becomes, using the residual sum of squares, summing the squared distance from each hand drawn point to its Bézier curve approximation (Herold). To expand further, to optimize the fitted Bézier curve, we are minimizing the error $\|B(t)P - \mathbf{H}\|$, where $\mathbf{H}$ is the matrix containing the data points.

$$E(\mathbf{P_y}) = \sum_{i=0}^{n} (y_i - P(t_i))^2 \tag{20}$$

"Then we can rewrite the error equation and find its maximum:" (Herold)

$$E(\mathbf{P_y}) = (\mathbf{y} - \mathbb{T}\mathbf{MP_y})^T(\mathbf{y} - \mathbb{T}\mathbf{MP_y}) \tag{21}$$

Then we take the derivative and equate it to zero to find the optimal solution for the normal equation. (Herold)

$$\frac{\partial E(\mathbf{P_y})}{\partial \mathbf{P_y}} = -2\mathbb{T}^T(y - \mathbb{T}\mathbf{MP_y}) = 0 \tag{22}$$

Solving for $\mathbf{P_y}$

$$\mathbf{P_y} = \mathbf{M^{-1}}(\mathbb{T}^T\mathbb{T})^{-1}\mathbb{T}^T\mathbf{Y} \tag{23}$$

Similarly $\mathbf{P_x}$ is,

$$\mathbf{P_x} = \mathbf{M^{-1}}(\mathbb{T}^T\mathbb{T})^{-1}\mathbb{T}^T\mathbf{X} \tag{24}$$

To avoid a lot of calculation, a python algorithm was used to manually trace the path of track, giving rise to the series of points $\mathbf{H}$, using the method outlined above, to output the control points $\mathbf{P}$.

Bezier Curves > bezier.py > ...

```python
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
from scipy.special import comb
from sklearn.metrics import r2_score
from scipy.interpolate import interp1d
from sklearn.metrics import mean_squared_error, mean_absolute_error

#-----START-----#
#modified from https://github.com/jegork/dysgraphia/blob/7edffa08a5c08571aeb080a625792797981d0607/bezier.py
def get_bezier_parameters(X, Y, degree=3):
    if degree < 1:
        raise ValueError('degree must be 1 or greater.')
    if len(X) != len(Y):
        raise ValueError('X and Y must be of the same length.')
    if len(X) < degree + 1:
        raise ValueError(f'There must be at least {degree + 1} points to '
                         f'determine the parameters of a degree {degree} curve. '
                         f'Got only {len(X)} points.')

    def bpoly(n, t, k):
        return t ** k * (1 - t) ** (n - k) * comb(n, k)

    def bmatrix(T):
        return np.matrix([[bpoly(degree, t, k) for k in range(degree + 1)] for t in T])

    def least_square_fit(points, M):
        M_ = np.linalg.pinv(M)
        return M_ * points

    T = np.linspace(0, 1, len(X))
    M = bmatrix(T)
    points = np.array(list(zip(X, Y)))

    final = least_square_fit(points, M).tolist()
    final[0] = [X[0], Y[0]]
    final[len(final)-1] = [X[len(X)-1], Y[len(Y)-1]]

    control_points = []
    for point in final:
        control_points.append([point[0], point[1]])

    print(control_points)

    return final

def bernstein_poly(i, n, t):
    return comb(n, i) * ( t**(n-i) ) * (1 - t)**i

def bezier_curve(points, nTimes=100):
    nPoints = len(points)
    xPoints = np.array([p[0] for p in points])
    yPoints = np.array([p[1] for p in points])

    t = np.linspace(0.0, 1.0, nTimes)

    polynomial_array = np.array([ bernstein_poly(i, nPoints-1, t) for i in range(0, nPoints)   ])

    xvals = np.dot(xPoints, polynomial_array)
    yvals = np.dot(yPoints, polynomial_array)

    return xvals, yvals
#-----END-----#
```

Figure 6: Part of the algorithm used to fit the Bézier curve (jegork)

4 Modelling

This is the image of the roller coaster that will be modelled.

Figure 7: The Rollercoaster Track

As some parameters of the roller coaster were unknown, like the vertical and horizontal height, the dimensions of the roller coaster were normalized to lie between 0 and 1 for the scope of this report. This can be done due to the affine invariant nature of Bézier curves.

As stated above, the roller coaster was then traced out by hand and the following picture showcases the input points $\mathbf{H}$ in red and then overlays the approximate 6^{th} order Bézier curve with a blue line.

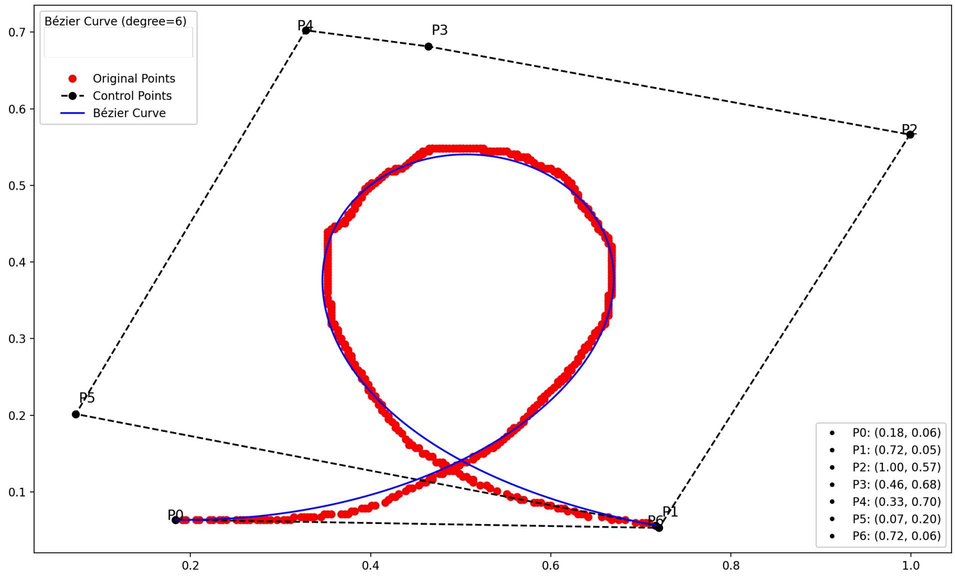

Figure 8: Tracing the curve using the algorithm and the Bézier curve

Here Bézier curve, superimposed on top of the image of the roller coaster. Overall, it was a good fit with a mean square error of only 0.6 and the control points as

$$\mathbf{P} = \begin{bmatrix} 0.18, 0.06 \\ 0.72, 0.05 \\ 1.00, 0.57 \\ 0.46, 0.68 \\ 0.33, 0.70 \\ 0.07, 0.20 \\ 0.72, 0.06 \end{bmatrix}_{7\times 1} \tag{25}$$

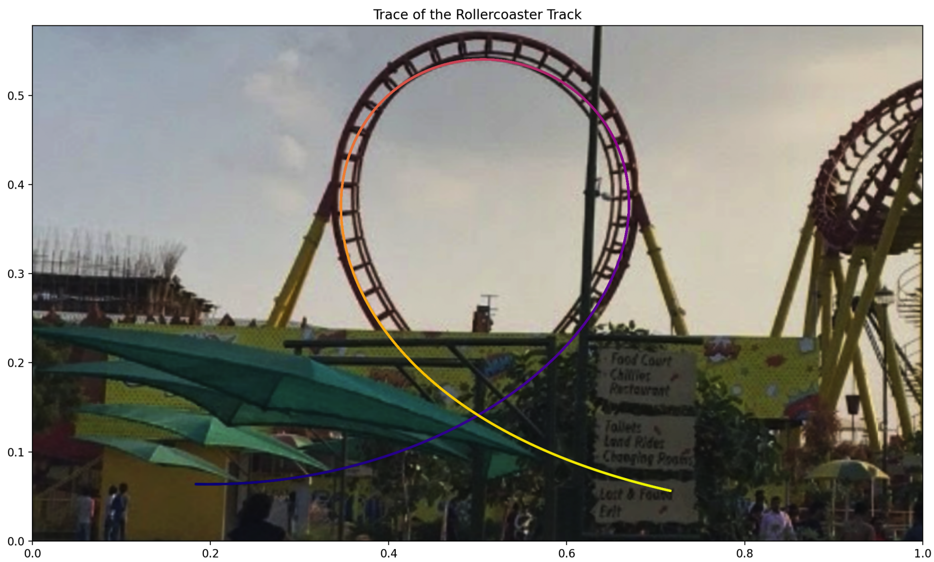

Figure 9: Curve fitting on the rollercoaster track

4.1 Derivatives of the Bézier curve

4.1.1 Position/Displacement

This is the original Bézier curve with the axis of x and y.

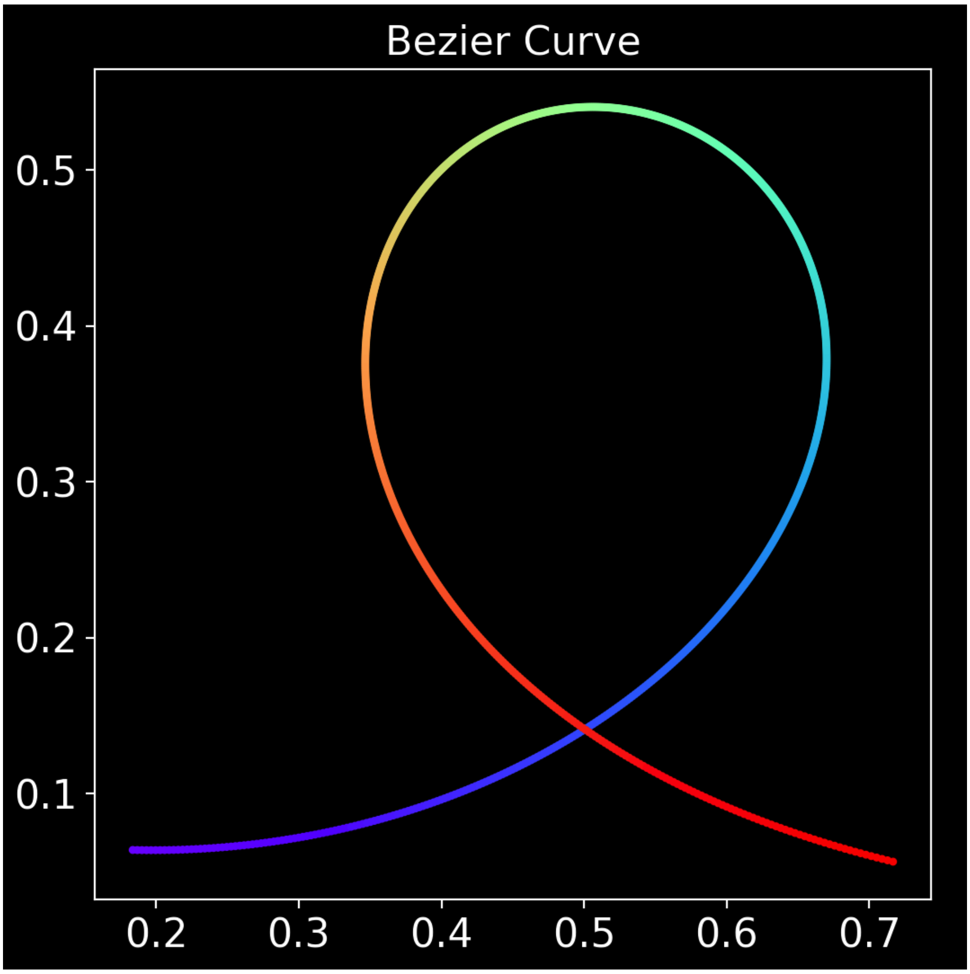

Figure 10: Bézier curve

4.2 Velocity

The first derivative of position with respect to time is velocity. Velocity is proportional to the amount of thrill felt by the human body (Eager et al.).

To compute the velocity, The derivative of the Bézier curve is taken as shown in Equation (12). Then the $\mathbf{x}$ and $\mathbf{y}$ components of the Bézier curve are resolved using $\sqrt{\mathbf{x}^2 + \mathbf{y}^2}$ and the result is plotted against time.

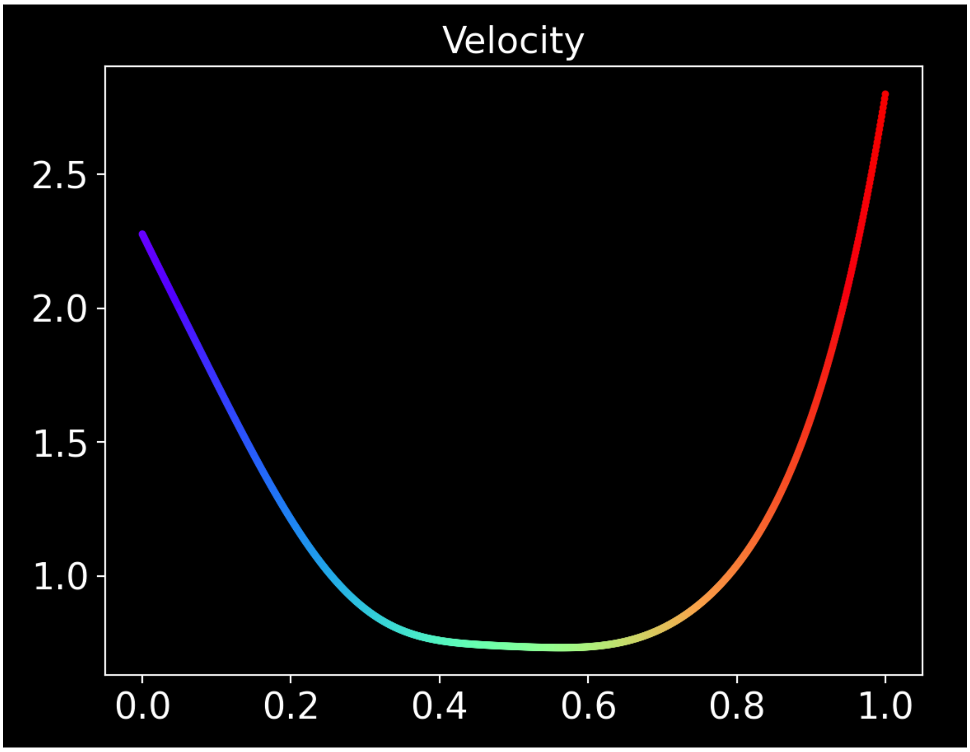

Figure 11: Velocity

4.3 Acceleration

Similarly, the second order derivative of position with respect to time is acceleration. Acceleration is an important metric as it governs the safety of the ride. The levels of accelerations that are safe for humans are governed by the ASTM standards set by the F24 committee, as shown below (Committee F24 on Amusement Rides and Devices).

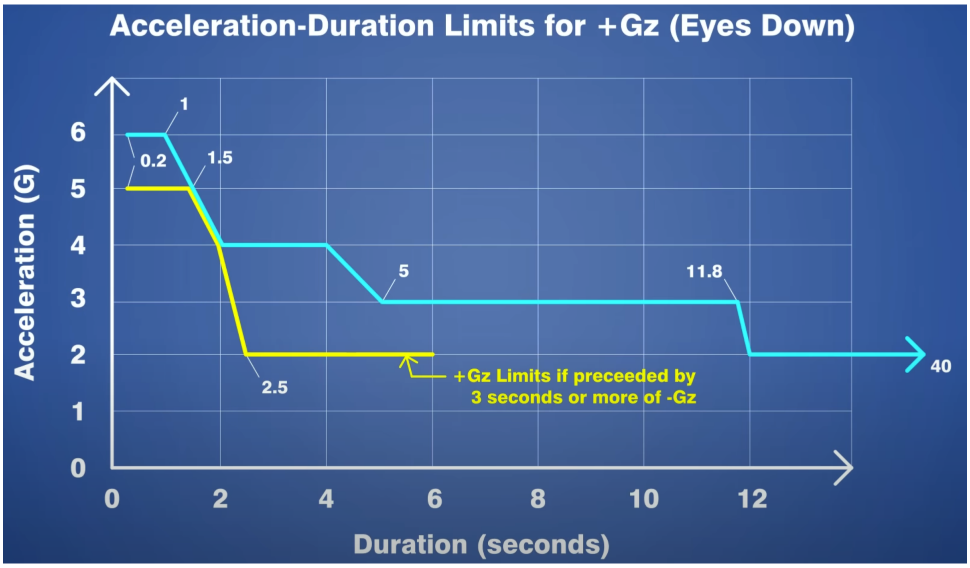

Figure 12: ASTM standards for acceleration in g's per second (ArtOfEngineering)

This is what the acceleration looks like for the modelled Bézier curve. The y axis is acceleration in meters per second square and the x axis is time.

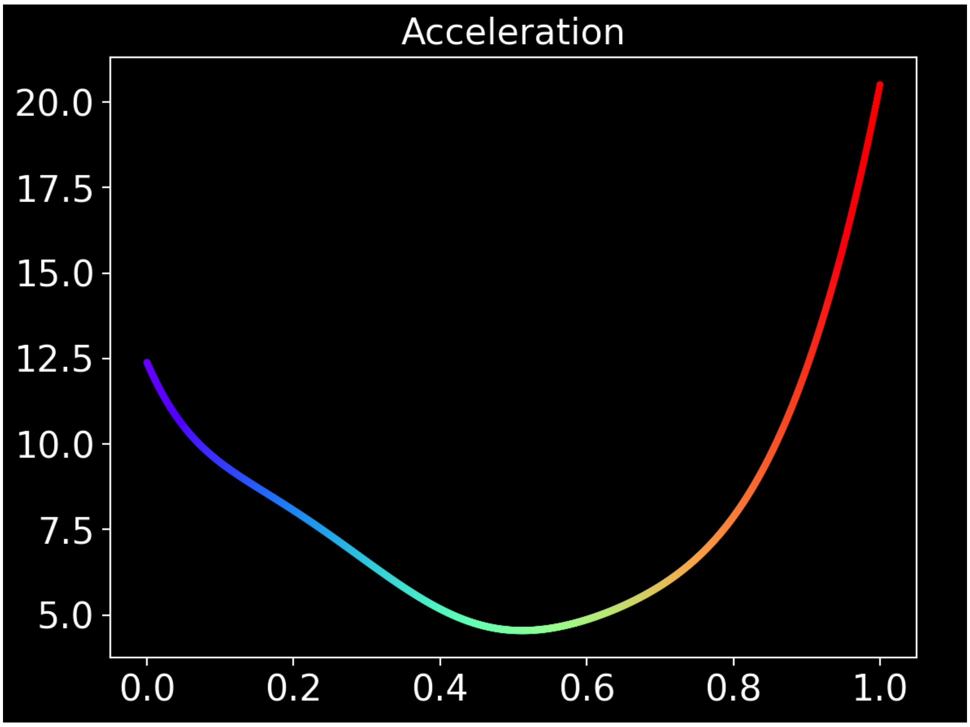

Figure 13: Acceleration

To convert the values from meters per second square to G's per second, to evaluate it against the standards, it simply needs to be divided by 9.81 meters per second square, which is the average gravitational acceleration on the surface of the earth. With the peaks in our Bézier curve only reaching a maximum of 2 G's per second, it is deemed safe as per the international standards provided.

5 Results

As stated earlier the model achieved a mean square error of only 0.6. However, this does not always mean it is a good fit. As the mean square error calculates the error without absolute values, it is possible to have overestimates and underestimates cancel out and result in a lower error.

Regardless, when compared to the ASTM standards, the roller coaster is deemed to be safe as it is well within the safety limitations of the standards

5.1 Constraints

One of the major constraints of this report was that it modelled roller coasters (3D) using 2-dimensional Bézier curves. While it is possible to use 3D Bézier curves, It is significantly more difficult due to problems arising from data collection and requiring sophisticated instruments like LiDAR sensors.

The data was gathered using a custom program, which took the image of the rollercoaster, and normalized the height and width of the rollercoaster to lie within 0 and 1, while also maintaining the aspect ratio. As the actual data for the rollercoasters was unavailable to the public, if those data points were known, the model could simply be scaled to fit the dimensions of the real rollercoaster.

Also, this model does not take into account, external forces like acceleration due to gravity and the engine of the roller coaster.

6 Conclusion

Bézier curves provide an elegant and highly scalable way to model complex paths like that of a roller coaster. They are also affine invariant, and can undergo transformations without changing the shape of the curve, unlike trigonometric functions. They also provide a homogenous approach to analysis unlike piecewise functions which are often used for complex shapes.

With the use of Bézier curve, a model with a mean square error of only 0.6. While this is not a perfect method sampling error, it is representative of the extensive capabilities of Bézier curves.

With this, it was found that the roller coaster that was modelled, was within international standards and is therefore, safe.

Additionally, while there are some constraints and limitations of this report, Bézier curves offer a framework for future research into the field of modelling.

Works Cited

ArtOfEngineering. The Real Physics of Roller Coaster Loops. *YouTube*, May 2019. www.youtube.com/watch?v=4q2W5SJc5j4.

Committee F24 on Amusement Rides and Devices. *Astm.org*, 2024. www.astm.org/committee-f24.

Eager, David, et al. "Beyond velocity and acceleration: jerk, snap and higher derivatives". *European Journal of Physics*, vol. 37, no. 6, Sept. 2016, p. 065008. https://doi.org/10.1088/0143-0807/37/6/065008.

Herold, Jim. Jim Herold. *Jim Herold*, Jan. 2021. www.jimherold.com/computer-science/best-fit-bezier-curve.

jegork. *GitHub*, 2024. github.com/jegork/dysgraphia/.

Kamermans, Mike 'Pomax'. A Primer on Bézier Curves. July 2023. pomax.github.io/bezierinfo/#derivatives.

Kiliçoglu, Şeyda, and Süleyman Şenyurt. "On the matrix representation of 5th order Bézier Curve and derivatives in E^3". *Communications Faculty Of Science University of Ankara Series A1Mathematics and Statistics*, vol. 71, no. 1, Mar. 2021, pp. 133–52. https://doi.org/0.31801/cfsuasmas.895598.

Parametric equation. *Encyclopædia Britannica*, 2024. www.britannica.com/science/parametric-equation.

Wolfram Research. *Wolfram.com*, 2024. mathworld.wolfram.com/PartitionofUnity.html.

A Appendix

Link to all source code used: https://github.com/Wayrion/Bezier-Fitter