B-Spline Trajectory Generation

MPAV Final Presentation

Ahmed M Ali

May 2023

Outline

1. Curve Fitting
2. Bezier Curves
3. B-Spline
   3.1 Proprieties
   3.2 Basis function
   3.3 Clamping
4. Conclusion

Curve Fitting

Curve Fitting

- To fit a nth order polynomial to a set of $m+1$ points $(x_i, y_i), i \in \{0, 1, 2, ..., n\}$, we should minimize this cost function:

$$J(x, y) = \sum_{i=0}^{m} (a_0 + a_1 x + a_2 x^2 + ... + a_n x^n - y_i)^2 \qquad (1)$$

Curve Fitting

- To fit a nth order polynomial to a set of $m+1$ points $(x_i, y_i), i \in \{0, 1, 2, ..., n\}$, we should minimize this cost function:

$$J(x, y) = \sum_{i=0}^{m} (a_0 + a_1 x + a_2 x^2 + ... + a_n x^n - y_i)^2 \tag{1}$$

- This could be written in matrix form as $Ax = b$ where:

$$A = \begin{bmatrix} 1 & x_1 & ... & x_1^n \\ 1 & x_2 & ... & x_2^n \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_m & ... & x_m^n \end{bmatrix}, x = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix}, b = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}$$

- Another method for curve fitting is **Lagrange polynomial**.
- It utilized cardinal functions as follows:

$$P_n(x) = \sum_{i=0}^{n} l_i(x).y_i$$

where $l_i(x) = \prod_{j=0, j\neq i}^{n} \left(\frac{x - x_j}{x_i - x_j} \right)$

- It is not highly sensitive to the data as higher order polynomials. However, the whole Lagrangian polynomial needs to be calculated if a new point to be added.

Bezier Curve

Bezier Curve

- Bezier Curve is another way to draw trajectory curves.
- It can be defined using **Bernstein polynomial**, where any point on the curve can be obtained through the following equation:

$$
\begin{aligned}
C(t) &= \sum_{i=0}^{n} b_{i,n}(t) P_i \\
b_{i,n}(t) &= \binom{n}{i} t^i (1-t)^{n-i}
\end{aligned}
\tag{2}
$$

  where the bezier curve is n degree with $n+1$ control points.
- For example, using Eq. 2, we can obtain the equations for a cubic curve as $(n = 3)$

$$C(t) = (1-t)^3 p_0 + 3t(1-t)^2 P_1 + 3t^2(1-t) P_2 + t^3 P_3$$

- Note that t should always be in $[0, 1]$ and all coefficients must sum to 1.
- it does not support local change. Change in any point affects the whole curve.
- Visually, it can be explained in this link.

B-Spline

B-Spline: Proprieties

- B-spline was introduced to overcome the following problems:
  - Not highly sensitive to new points (High order Polynomials)
  - No need to calculate the whole coefficients if a new point is added (Lagrangian polynomials)
  - Can have many control points but less degree curves (Bezier curve)
  - support local changes (Bezier curve)
- B-spline can be thought as aggregating multiple Bezier curves together.
- It achieves C^0, C^1 and C^2 continuity.

B-Spline: Basis

- It uses **Basis function** in its curve equation:

$$S(t) = \sum_{i=0}^{n} N_{i,k}(t) P_i \tag{3}$$

where $N_{i,k}$ is the basis function, P_i is control point, k is polynomial degree, and $n + 1$ is the number of control points. The basis function is calculated using **Cox-de Boor** recursive formula:

$$N_{i,0}(t) = \begin{cases} 1 & \textit{if } t_i \leq t \leq t_{i+1} \\ 0 & \textit{otherwise} \end{cases}$$
$$N_{i,j}(t) = \frac{t - t_i}{t_i + j - t_i} N_{i,j-1}(t) + \frac{t_{i+j+1} - t}{t_{i+j+1} - t_{i+1}} N_{i+1,j-1}(t) \tag{4}$$

B-Spline: Basis

- Given the B-spline degree k and number of control points $n+1$, we can calculate:
  - Number of generated Bezier curves $= n - k + 1$
  - Number of Knots is $m+1$ where $m = k + n + 1$
- For example, if we have a cubic b-spline that has five control points, the knot vector will contain $m = 3 + 4 + 1 = 8$ so we will have 9 knots.
- In contrast to Bezier curve, the value of t can take any number between $[t_0, t_m]$.
- Uniform B-spline is generated when the knot values are equidistant.

B-Spline: Basis

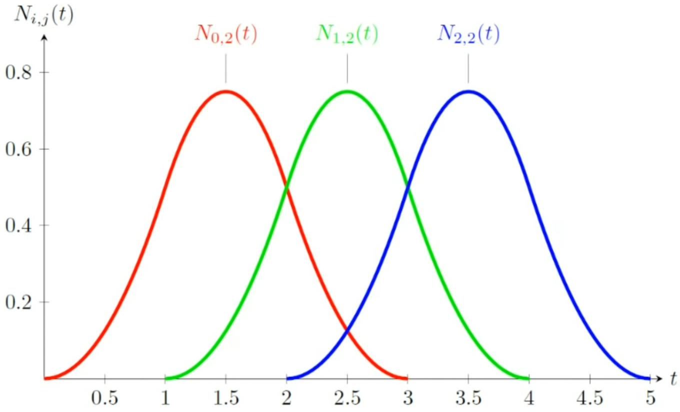
- For a uniform quadratic B-spline with three control points, we got the following basis functions:

$$N_{0,2}(t) = \begin{cases} \frac{1}{2}t^2 & \textit{if } 0 \leq t \leq 1 \\ \frac{1}{2}[-2(t-1)^2 + 2(t-1) + 2] & \textit{if } 1 \leq t \leq 2 \\ \frac{1}{2}[(t-2)^2 - 2(t-2) + 1] & \textit{if } 2 \leq t \leq 3 \\ 0 & \textit{otherwise} \end{cases}$$

$$N_{1,2}(t) = \begin{cases} \frac{1}{2}(t-1)^2 & \textit{if } 1 \leq t \leq 2 \\ \frac{1}{2}[-2(t-2)^2 + 2(t-2) + 2] & \textit{if } 2 \leq t \leq 3 \\ \frac{1}{2}[(t-3)^2 - 2(t-3) + 1] & \textit{if } 3 \leq t \leq 4 \\ 0 & \textit{otherwise} \end{cases}$$

$$N_{2,2}(t) = \begin{cases} \frac{1}{2}(t-2)^2 & \textit{if } 2 \leq t \leq 3 \\ \frac{1}{2}[-2(t-3)^2 + 2(t-3) + 2] & \textit{if } 3 \leq t \leq 4 \\ \frac{1}{2}[(t-4)^2 - 2(t-4) + 1] & \textit{if } 4 \leq t \leq 5 \\ 0 & \textit{otherwise} \end{cases}$$

B-Spline: Basis

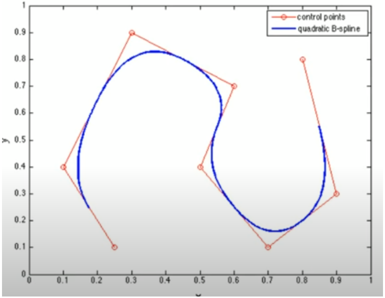


B-Spline: Clamping

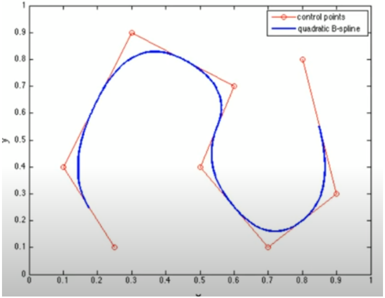
- Generating a B-spline, it does not begin and end at the desired start and end points respectively.

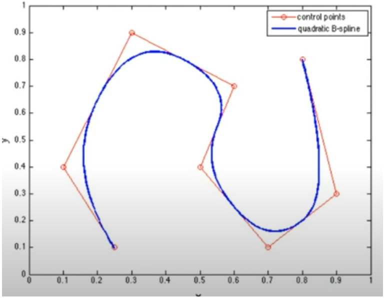
- Generating a B-spline, it does not begin and end at the desired start and end points respectively.
  Solution: Apply clamping method to the first and last knots.
  For example knot vector of $T = (0, 1, 2, 3, 4, 5, 6)$ will be $T = (0, 0, 1, 2, 3, 3, 3)$

- Generating a B-spline, it does not begin and end at the desired start and end points respectively.
  Solution: Apply clamping method to the first and last knots. For example knot vector of $T = (0, 1, 2, 3, 4, 5, 6)$ will be $T = (0, 0, 1, 2, 3, 3, 3)$



- Generating a B-spline, it does not begin and end at the desired start and end points respectively.
  Solution: Apply clamping method to the first and last knots.
  For example knot vector of $T = (0, 1, 2, 3, 4, 5, 6)$ will be
  $T = (0, 0, 1, 2, 3, 3, 3)$





Conclusion

	Polynomial	Lagrangian Polynomials	Bezier Curve	B-spline
Overfit the data	X	X	✓	✓
Support Local change	X	X	X	✓
Degree = n-1	X	X	X	✓

B-spline overcomes many problems that arise with other types of curve fitting.

References

1. https://github.com/GPrathap/motion_planning/blob/main/lectures/mpav_spline.pdf
2. https://www.youtube.com/watch?v=qhQrRCJ-mVg
3. https://www.youtube.com/watch?v=JwN43QAIF50

Thank you