Mathematics Behind Bézier Curves

1. Lagrange Interpolation

Lagrange Interpolation is a polynomial interpolation method, providing an explicit formula for calculating a smooth curve that passes through a set of control points. This method can be used to approximate Bézier curves by interpolating through multiple control points. However, it is less commonly used for Bézier curves due to stability issues with high-degree polynomials but is effective with lower degrees.

Lagrange Polynomial Formula

For n control points $P_0, P_1, \ldots, P_n$, the Lagrange interpolation polynomial $P(t)$ is defined as:

$$P(t) = \sum_{i=0}^{n} L_i(t) \cdot P_i$$

where t is a parameter in the interval $[0, 1]$ and $L_i(t)$ are the Lagrange basis polynomials.

Lagrange Basis Polynomial

The basis polynomial $L_i(t)$ for each control point P_i is given by:

$$L_i(t) = \prod_{\substack{0 \leq j \leq n \\ j \neq i}} \frac{t - t_j}{t_i - t_j}$$

where t_i and t_j represent the parameter values corresponding to each control point P_i and P_j. Each $L_i(t)$ is a polynomial of degree n, ensuring that $L_i(t_j) = 1$ if $i = j$ and $L_i(t_j) = 0$ otherwise. This property guarantees that the curve passes exactly through each control point.

Example Calculation

To calculate the curve point $P(t)$ at a given parameter t, evaluate each $L_i(t)$ and combine them with the corresponding control points P_i. As t varies from 0 to 1, $P(t)$ traces out the interpolated curve through the control points.

2. De Casteljau's Algorithm

De Casteljau's Algorithm is a recursive, geometric approach for constructing Bézier curves. Unlike Lagrange interpolation, this method is numerically stable and is the preferred method for generating Bézier curves in computer graphics.

Algorithm Description

Given a set of $n+1$ control points $P_0, P_1, \ldots, P_n$, De Casteljau's algorithm computes a point $P(t)$ on the curve for a given parameter t (where $t \in [0,1]$) by recursively interpolating between consecutive control points.

Recursive Formula

For each k-th level of recursion, we compute intermediate points:

$$P_i^{(k)}(t) = (1-t) \cdot P_i^{(k-1)} + t \cdot P_{i+1}^{(k-1)}$$

where:

- $P_i^{(0)}$ are the original control points $P_0, P_1, \ldots, P_n$.
- $P_i^{(k)}$ are the intermediate points generated at each recursive level k.
- The recursion continues until only one point remains, which is $P(t)$, the point on the Bézier curve at the given parameter t.

Example Calculation

1. **Base Case**: Start with control points $P_0, P_1, \ldots, P_n$.
2. **First Recursion**: Interpolate between adjacent points to get $P_0^{(1)}, P_1^{(1)}, \ldots, P_{n-1}^{(1)}$.
3. **Continue Recursion**: Repeat until a single point $P_0^{(n)}$ remains.

The final point $P_0^{(n)}$ is the point on the curve at parameter t.

Advantages of De Casteljau's Algorithm

- **Numerical Stability**: Each interpolation step averages points, reducing numerical instability.
- **Flexibility**: Can be applied to curves of any degree and easily generalized to create higher-dimensional curves (e.g., Bézier surfaces).

Summary

Both methods provide ways to generate smooth curves based on control points:

- **Lagrange Interpolation** directly fits a polynomial that passes through all control points but can suffer from numerical instability with a high number of points.
- **De Casteljau's Algorithm** is a recursive method that is stable and commonly used for generating Bézier curves, especially in computer graphics.

Choose the method that best fits your needs based on the specific application and performance requirements.