Assignment 4: Bézier Curve

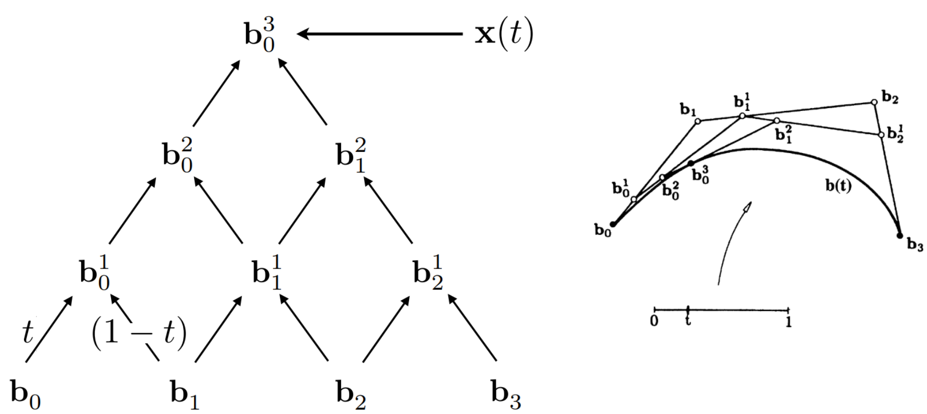
Mengzhu Wang

January 27, 2024

1 de Casteljau Algorithm

The de Casteljau algorithm can be regarded as repeated linear interpolation. From the control points $p_0, ..., p_n$, we define points on the line segments $p_i p_{i+1}$ (the control polyline) by choosing a value t and defining $p_i^1 = (1-t)p_i + tp_{i+1}$. From the n points p_i^1, we can repeat the process and define the $n-1$ points $p_i^2 = (1-t)p_i^1 + tp_{i+1}^1$ on the line segments $p_i^1 p_{i+1}^1$. This process can be repeated defining $p_i^{j+1} = (1-t)p_i^j + tp_{i+1}^j$ for $i = 0, 1, 2, ..., n-j-1$ for $j = 0, 1, 2, ..., n-1$ where $p_i^0 = p_i$. The end of this process produces a point p_0^n on a polynomial curve of degree n.

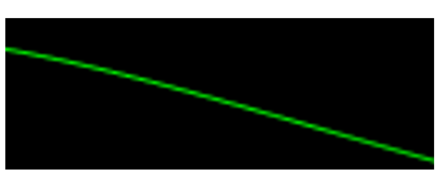
Bernstein form of a Bézier curve of order n is $\mathbf{b}^b(t) = \mathbf{b}_0^n(t) = \sum_{j=0}^n \mathbf{b}_j B_j^n(t)$, where Bernstein polynomials is $B_i^n(t) = \binom{n}{i} t^i (1-t)^{n-i}$. Assume n=3, we would have control points in 3D, and these points define a Bézier curve in 3D that is a cubic polynomial in t: $\mathbf{b}^n(t) = \mathbf{b}_0(1-t)^3 + \mathbf{b}_1 3t(1-t)^2 + \mathbf{b}_2 3t^2(1-t) + \mathbf{b}_3 t^3$.



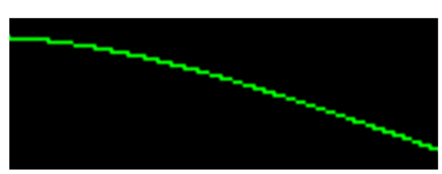
Every rightward arrow is multiplication by t,
Every leftward arrow by (1–t)

2 Anti-Aliasing

Drawing ideas from bilinear interpolatoion, the nearest four pixels are considered. The color of each pixel is calculated by $255 * (1 - distance(pixel, currentpoint))$. If the same pixel has been visited, we chose the maximum value for it.



a) anti-aliasing



b) w/o anti-aliasing