Bézier Curves

- An alternative to cubic splines for smooth curve generation are Bézier curves.
- They are parametric curves commonly used in computer graphics for generate smooth curves that can be scaled indefinitely (to higher dimensions).
- Combinations of linked Bézier curves are called paths that are easy to manipulate and can even be used in animation as a tool to control motion.

Bézier Curves

- Used in the time domain, they can be used to specify the velocity over time of an object such as an icon moving from point A to point B, rather than simply moving a fixed number of pixels per step.
- The mathematical basis for Bézier curves is the Bernstein polynomial (circa 1912), and the French engineer Pierre Bézier demonstrated their use in the design of automobile bodies (for Renault) in 1962.

Bézier Curves

- Applications
  - Computer graphics for smooth curves (quadratic and cubic Bézier curves are the most common); used by Adobe Illustrator, CorelDraw, Inkscape, and Microsoft Excel (smooth curve feature for charts).
  - Animation for outlining movement; used by Adobe Flash and Synfig.
  - TrueType fonts are composed of quadratic Bézier curves (used by font engines like FreeType).

Linear Bézier Curves

- A Bézier curve (BC) is defined by a set of control points P_0 through P_n, where n is the order of the (BC). For a linear BC, n=1 and for a quadratic BC, n=2. The first and last control points are always the endpoints of the curve and the intermediate control points usually do not lie on the curve.
- For the case n=1 with points P_0 and P_1, the BC is simply a straight line between those points:

$$B(t) = P_0 + t(P_1 - P_0) = (1 - t)P_0 + tP_1, t \in [0,1].$$

Quadratic Bézier Curves

- A quadratic BC is the path traced by the function $B(t)$, given control points P_0, P_1, and P_2:

$$B(t) = (1-t)\left[(1-t)P_0 + tP_1\right] + t\left[(1-t)P_1 + tP_2\right], t \in [0,1].$$

- The function $B(t)$ above linearly interpolates the points that lie on all the linear BCs defined from P_0 to P_1 and all the linear BCs defined from P_1 to P_2.
- Rearranging the right-hand-side for $B(t)$ we obtain:

$$B(t) = (1-t)^2 P_0 + 2(1-t)tP_1 + t^2 P_2, t \in [0,1].$$

Quadratic Bézier Curves

- Since we have $B(t) = (1-t)^2 P_0 + 2(1-t)tP_1 + t^2 P_2, t \in [0,1]$. Suppose we compute its derivative $B'(t)$:

$$B'(t) = 2(1-t)(P_1 - P_0) + 2t(P_2 - P_1).$$

It can be shown that the tangents to the curve $B(t)$ at P_0 and P_2 intersect at _____. As t increases from 0 to 1, the curve departs from P_0 in the direction of _____, and then bends to arrive at P_2 in the direction of _____.

Cubic Bézier Curves

- A cubic BC is the path traced by the function $B(t)$, given control points P_0, P_1, P_2, and P_3.
- The curve $B(t)$ starts at P_0 going toward P_1 and later arrives at P_3 from the direction of P_2. The curve typically does not pass through the intermediate control points (P_1 and P_2) – they are used for directions.
- Distance between P_0 and P_1 determines how long $B(t)$ moves into direction P_2 before turning towards P_3.

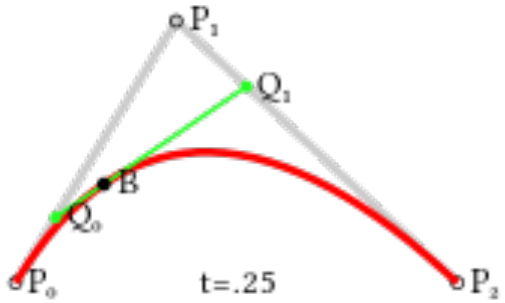
THE UNIVERSITY of TENNESSEE KNOXVILLE
Big Orange. Big Ideas.

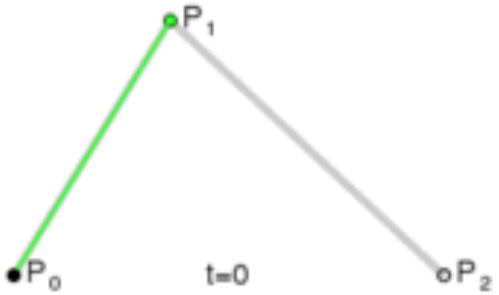
$$B(t) = (1-t)^3 P_0 + 3(1-t)^2 t P_1 + 3(1-t) t^2 P_2 + t^3 P_3, t \in [0,1].$$

Bézier Curve Animations

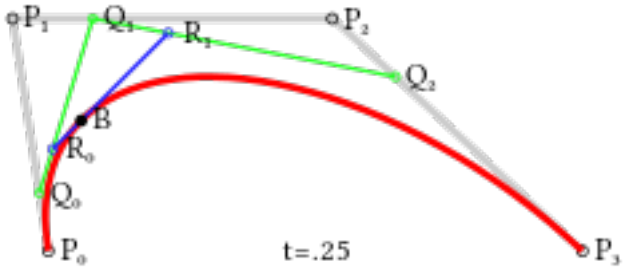
- Images courtesy of Wikipedia:

Quadratic





Cubic



THE UNIVERSITY of TENNESSEE KNOXVILLE
Big Orange. Big Ideas.

matplotlib.patch

- The **patch** function uses a **Path** object to generate Bézier curves via standard curve commands (**MOVETO**, **LINETO**, **CLOSEPOLY**).
- Typically specify a path via

```
path = Path(verts,codes)
```

  where `verts` is a list of 2D data points and codes is a list of curve commands.
- See **bezier.py** script file for demo of a cubic Bézier curve using the **patch** function.