Preliminary concepts

desired curve	**linear** approximation $ax+by+c=0$	**quadratic** approximation $y=ax^2+bx+c$	**piecewise linear** approximation *polyline*

A polyline is a *piecewise linear approximation* to a curve. Many pieces are required for a close approximation, making it tedious to change shape (and notice that where the pieces join it is not smooth). What we're looking for is a more compact, more easily manipulable representation for piecewise smooth curves.

Parametric form is very attractive: Define $Q(t)=\begin{bmatrix}x(t) & y(t)\end{bmatrix}$, a mapping from parameter space, $t \in R$, to model space, (x, y). $x(t)$, $y(t)$ are usually polynomial functions, with $t \in [0, 1]$.

Example: parameteric form of a line from P_0 to P_1: $Q(t)=(1-t)P_0+(t)P_1,\ t\in[0,1]$

Parameteric form can represent multi-valued functions.

Example: 2D circle: $\begin{bmatrix}r\cos\theta & r\sin\theta\end{bmatrix}, \theta\in[0,2\pi]$ (note that this is not a polynomial function)

Tangent is given by: $Q'(t)=\dfrac{d}{dt}Q(t)$ … which is never infinite (!)

Suppose we want to find the curve $Q(t)=\begin{bmatrix}x(t) & y(t)\end{bmatrix}$ that passes though (*interpolates*) a set of points. We call these points the *"geometric constraints"*. One approach is to use *Lagrange interpolation*.

Lagrange interpolation

A set of $n+1$ points, $P_0, P_1, \cdots, P_n$, can be interpolated $Q(t) = P_0L_0^n(t) + P_1L_1^n(t) + \cdots + P_nL_n^n(t)$ using the *Lagrange polynomials of degree n*, $L_i^n(t)$:

$$L_i^n(t) = \prod_{j=0}^{n} \frac{(t - t_j)}{(t_i - t_j)}, (j \neq i) = \frac{(t - t_0)(t - t_1)\cdots(t - t_n)}{(t_i - t_0)(t_i - t_1)\cdots(t_i - t_n)}, (j \neq i)$$

For n = 3, the 3rd degree Lagrange polynomials would be given by:

$$L_i^3(t) = \prod_{j=0}^{3} \frac{(t - t_j)}{(t_i - t_j)}, (j \neq i) = \frac{(t - t_0)(t - t_1)(t - t_2)(t - t_3)}{(t_i - t_0)(t_i - t_1)(t_i - t_2)(t_i - t_3)}, (j \neq i)$$

$$L_i^3(t) = \prod_{j=0}^{3} \frac{(t - t_j)}{(t_i - t_j)}, (j \neq i) = \frac{(t - t_0)(t - t_1)(t - t_2)(t - t_3)}{(t_i - t_0)(t_i - t_1)(t_i - t_2)(t_i - t_3)}, (j \neq i)$$

We drop the term where $j = i$, which gives us:

$$L_0^3(t) = \frac{(t-t_0)(t-t_1)(t-t_2)(t-t_3)}{(t_0-t_0)(t_0-t_1)(t_0-t_2)(t_0-t_3)} = \frac{(t-\frac{1}{4})(t-\frac{3}{4})(t-1)}{(0-\frac{1}{4})(0-\frac{3}{4})(0-1)} = -\frac{16}{3}(t-\tfrac{1}{4})(t-\tfrac{3}{4})(t-1)$$

$$L_1^3(t) = \frac{(t-t_0)(t-t_1)(t-t_2)(t-t_3)}{(t_1-t_0)(t_1-t_1)(t_1-t_2)(t_1-t_3)} = \frac{(t-0)(t-\frac{3}{4})(t-1)}{(\frac{1}{4}-0)(\frac{1}{4}-\frac{3}{4})(\frac{1}{4}-1)} = \frac{32}{3}t(t-\tfrac{3}{4})(t-1)$$

$$L_2^3(t) = \frac{(t-t_0)(t-t_1)(t-t_2)(t-t_3)}{(t_2-t_0)(t_2-t_1)(t_2-t_2)(t_2-t_3)} = \frac{(t-0)(t-\frac{1}{4})(t-1)}{(\frac{3}{4}-0)(\frac{3}{4}-\frac{1}{4})(\frac{3}{4}-1)} = -\frac{32}{3}t(t-\tfrac{1}{4})(t-1)$$

$$L_3^3(t) = \frac{(t-t_0)(t-t_1)(t-t_2)(t-t_3)}{(t_3-t_0)(t_3-t_1)(t_3-t_2)(t_3-t_3)} = \frac{(t-0)(t-\frac{1}{4})(t-\frac{3}{4})}{(1-0)(1-\frac{1}{4})(1-\frac{3}{4})} = \frac{16}{3}t(t-\tfrac{1}{4})(t-\tfrac{3}{4})$$

Example. Use Lagrange Interpolation to interpolate the points, `{ (0,0),(2,2),(0,3),2,4) }`.

The simplest approach for selecting the values t_k is to use equally spaced values in $[0 \quad 1]$, although this is not required. For example we could use $t =$ `( 0, 0.25, 0.75, 1.0 )`.

For $t = 0.25$, evaluating the Lagrange polynomials results in:

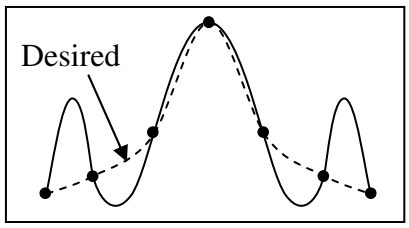
$$L_0^3(\tfrac{1}{4}) = -\tfrac{16}{3}(\tfrac{1}{4} - \tfrac{1}{4})(\tfrac{1}{4} - \tfrac{3}{4})(\tfrac{1}{4} - 1) = 0 \qquad L_1^3(\tfrac{1}{4}) = \tfrac{32}{3}(\tfrac{1}{4})(\tfrac{1}{4} - \tfrac{3}{4})(\tfrac{1}{4} - 1) = \tfrac{32}{3}(\tfrac{1}{4})(-\tfrac{1}{2})(-\tfrac{3}{4}) = 1$$
$$L_2^3(\tfrac{1}{4}) = -\tfrac{32}{3}(\tfrac{1}{4})(\tfrac{1}{4} - \tfrac{1}{4})(\tfrac{1}{4} - 1) = 0 \qquad L_3^3(\tfrac{1}{4}) = \tfrac{16}{3}(\tfrac{1}{4})(\tfrac{1}{4} - \tfrac{1}{4})(\tfrac{1}{4} - \tfrac{3}{4}) = 0$$

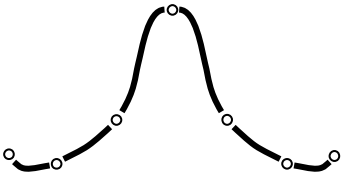
Substituting back into $Q(t) = P_0 L_0^n(t) + P_1 L_1^n(t) + \cdots + P_n L_n^n(t)$, we should get a point on the curve:

$$\begin{aligned} Q_x(\tfrac{1}{4}) &= P_{0_x} L_0^3(\tfrac{1}{4}) + P_{1_x} L_1^3(\tfrac{1}{4}) + P_{2_x} L_2^3(\tfrac{1}{4}) + P_{3_x} L_3^3(\tfrac{1}{4}) \\ &= (0 \cdot 0) + (2 \cdot 1) + (0 \cdot 0) + (2 \cdot 0) \\ &= 2 \end{aligned} \qquad \begin{aligned} Q_y(\tfrac{1}{4}) &= P_{0_y} L_0^3(\tfrac{1}{4}) + P_{1_y} L_1^3(\tfrac{1}{4}) + P_{2_y} L_2^3(\tfrac{1}{4}) + P_{3_y} L_3^3(\tfrac{1}{4}) \\ &= (0 \cdot 0) + (2 \cdot 1) + (3 \cdot 0) + (4 \cdot 0) \\ &= 2 \end{aligned}$$

$$Q(0.25) = (2,2)$$

Polynomial interpolation gives smooth curves, but the high degree polynomials required to interpolate many points results in curves with many oscillations.



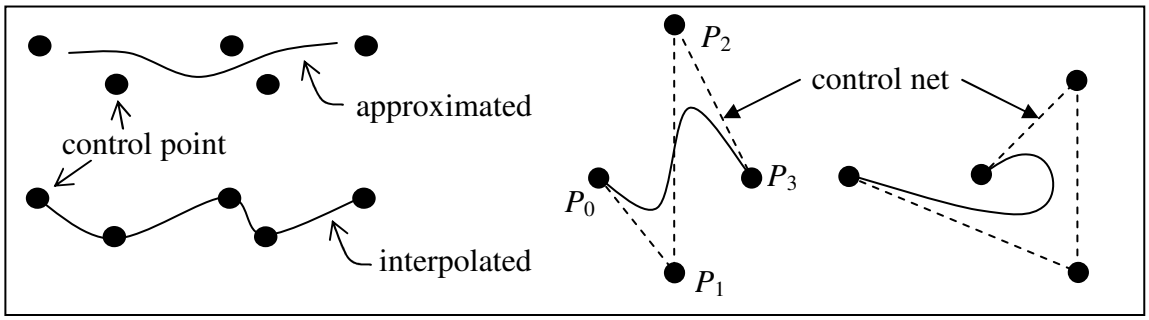


Piecewise continous parametric polynomial curves

This is a curve "pieced" together using short segments of polynomial curves defined parametrically. It is more commonly called a *spline curve* (or just "spline").

Physical splines are thin, flexible metal or wood strips traditionally used by draftsmen, ship modelers, and automobile designers. Physical splines were held in shape by weights called *ducks*. Mathematical splines are "held in place" by a set of *control points* analagous to the ducks.

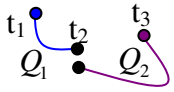
A spline curve may *approximate* or *interpolate* its control points. The *control net* is the polyline connecting the control points for an approximating spline.



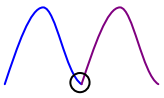
Continuity at the join point between segments

Defined as *parametric* (C^0, C^1, C^2, ...) and *geometric* (G^0, G^1, G^2, ...) continuity. Consider two polynomial curve segments:

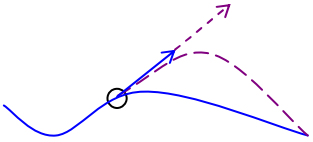
$$Q_1(t) \quad t \in [t_1, t_2]$$
$$Q_2(t)' \quad t \in [t_2, t_3]$$



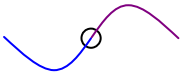
C^0, G^0: $Q_1(t_2) = Q_2(t_2)$: the coordinates are equal at the join point.



$G^{1:}$ $Q_1'(t_2) = kQ_2'(t_2)$: parametric first derivatives are *proportional* at the join point (the direction of the tangents are equal, but the magnitude may differ).



C^1: $Q_1'(t_2) = Q_2'(t_2)$: Parametric first derivatives (tangents) are equal at the join point.



G^2: $Q_1''(t_2) = kQ_2''(t_2)$: Parametric second derivatives (rate of change of the tangents) are proportional at the join point.

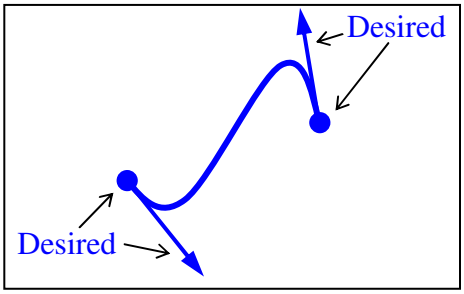
C^2: $Q_1''(t_2) = Q_2''(t_2)$: Parametric second derivatives are equal at the join point.

C^1 continuity is generally adequate for drawing/rendering. C^2 continuity is desirable if the spline defines a motion path (for example, the path of a camera through a scene) to ensure there is no abrupt change in acceleration through the join points.

Spline curves are often specified using **cubic polynomials**: $x(t) = a_x t^3 + b_x t^2 + c_x t + d_x$

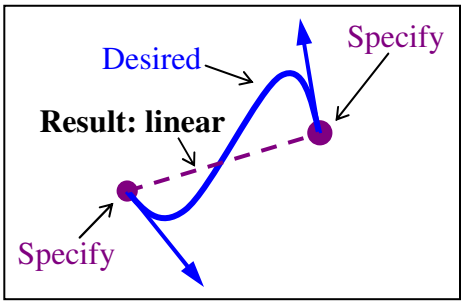
Cubic polynomials are used because lower degree polynomials are not flexible enough in controlling shape, and higher degree polynomials are generally too flexible. Cubics are "just right", as illustrated below:

Suppose we want a curve segment to interpolate two endpoints. There are an infinite number of such curves, but we want ours to have a particular shape, in fact with a certain **tangent** (derivative) **at the endpoints**. Linear or quadratic polynomials do not allow a curve segment to pass through two specified endpoints, such that the curve also has specified derivatives at the endpoints.



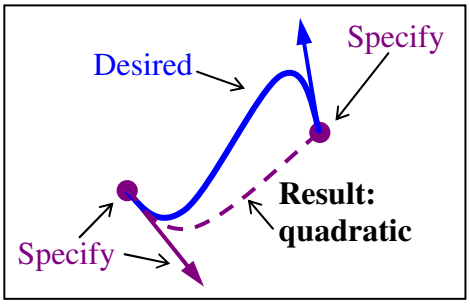
- Linear: $x(t) = a_x t + b_x$

The two coefficients, a_x and b_x, give us two degrees of freedom, i.e., two things we can specify. These two things are the endpoint themselves (i.e., their coordinates). Once we specify the endpoints we want interpolated, *we have determined the derivative at each endpoint also*: it is the slope of the line between the endpoints (and is constant over the entire curve)



- Quadratic: $x(t) = a_x t^2 + b_x t + c_x$

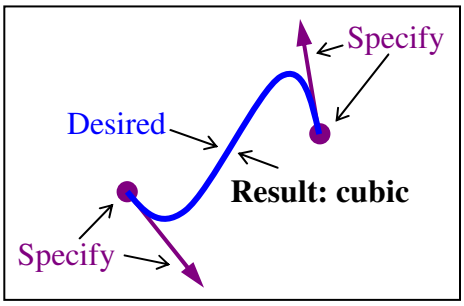
Three coefficients ⇒ three degrees of freedom (we can specify two endpoints and one derivative). Better, but we still have control over the shape at one endpoint only.



- **Cubic**: $x(t) = a_x t^3 + b_x t^2 + c_x t + d$

Four coefficients ⇒ four degrees of freedom … allowing us to specify two endpoint positions and two endpoint derivatives.

… giving us control over the shape at both endpoints.

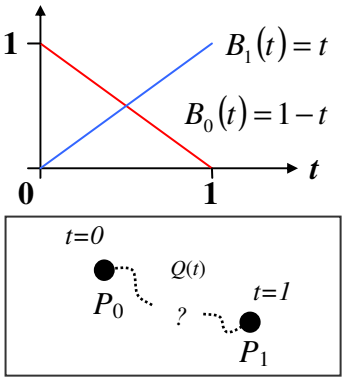


- Higher degree: need to specify many more conditions (too many to be manageable)

Parametric polynomial curves: (the general idea introduced using the linear case)

Let's use *linear parametric polynomials* to represent a curve interpolating two control points, P_0 and P_1. We will be speaking in general terms about the "shape" of that curve, knowing beforehand that in this case it's just a line from one point to the other. Although we know it's a line let's look at the parameteric representation and see how this "shape" – a straight line - is determined.

Consider the two linear polynomials $B_0(t), B_1(t)$ shown to the right. We call these *blending functions*. Control points P_0 and P_1 are called our *geometric constraints*.



A blending function tells us how much a geometric constraint influences the shape of the curve $Q(t)$ at a given value of the parameter *t*:

$$Q(t) = B_0(t)P_0 + B_1(t)P_1$$

A point on a parametric polynomial curve is the sum of the geometric constraints weighted by the blending functions.

Using two control points as our geometric contraints and our two linear blending functions we have:

$$Q(t) = (1-t)P_0 + (t)P_1$$

This is just the familiar parametric equation of a line through endpoints P_0 and P_1.

As we stated, a set of geometric constraints and a set of blending functions determine the shape of a parametric polynomial curve. And as we saw, using linear polynomial blending functions gives us a line. Knowing that a line must interpolate its endpoints, the question is: does the relationship

$$Q(t) = B_0(t)P_0 + B_1(t)P_1$$

using blending functions $B_0(t) = 1-t$, $B_1(t) = t$ and control points P_0, P_1 show that the curve interpolates these two points? Observe:

$t = 0$:	$B_1(0) = 0$	$B_1(t)P_1 = 0$	P_1 has no influence on the shape of the curve
	$B_0(0) = 1$	$B_0(t)P_0 = P_0$	$Q(0) = P_0$: the curve interpolates P_0.
$t = 1$:	$B_0(1) = 0$	$B_0(t)P_0 = 0$	P_0 has no influence on the shape of the curve
	$B_1(1) = 1$	$B_1(t)P_1 = P_1$	$Q(1) = P_1$: the curve interpolates P_1.

The above discussion illustrated curve representation using parametric linear polynomials. Note that using *parametric cubic polynomials* is just a generalization of linear case:

A point on a parametric polynomial curve is the sum of the geometric constraints weighted by the blending functions:

$$Q(t) = \sum_i B_i(t)P_i$$

Parametric cubic curves (for clarity we will show the math in 2D, but it extends to 3D)

Cubic polynomials defining a curve segment $Q(t)=\begin{bmatrix}x(t)\\y(t)\end{bmatrix}$ are of the form:

$$x(t)=a_x t^3+b_x t^2+c_x t+d_x$$
$$y(t)=a_y t^3+b_y t^2+c_y t+d_y,\quad 0\le t\le 1$$

Rewriting the above system of equations:

$$Q(t)=\begin{bmatrix}x(t)\\y(t)\end{bmatrix}=\begin{bmatrix}a_x t^3+b_x t^2+c_x t+d_x\\a_y t^3+b_y t^2+c_y t+d_y\end{bmatrix}=\begin{bmatrix}t^3 & t^2 & t & 1\end{bmatrix}\begin{bmatrix}a_x & a_y\\b_x & b_y\\c_x & c_y\\d_x & d_y\end{bmatrix}=T\cdot C$$

$$\text{Let } C=\begin{bmatrix}m_{11} & m_{12} & m_{13} & m_{14}\\m_{21} & m_{22} & m_{23} & m_{24}\\m_{31} & m_{32} & m_{33} & m_{34}\\m_{41} & m_{42} & m_{43} & m_{44}\end{bmatrix}\begin{bmatrix}G_1\\G_2\\G_3\\G_4\end{bmatrix}=M\cdot G$$

Then $Q(t)=\begin{bmatrix}t^3 & t^2 & t & 1\end{bmatrix}\begin{bmatrix}m_{11} & m_{12} & m_{13} & m_{14}\\m_{21} & m_{22} & m_{23} & m_{24}\\m_{31} & m_{32} & m_{33} & m_{34}\\m_{41} & m_{42} & m_{43} & m_{44}\end{bmatrix}\begin{bmatrix}G_1\\G_2\\G_3\\G_4\end{bmatrix}=T\cdot M\cdot G$ M is called a *basis matrix.*

G contains our four *geometric constraints*: $G=\begin{bmatrix}G_1\\G_2\\G_3\\G_4\end{bmatrix}=\begin{bmatrix}g_{1x} & g_{1y}\\g_{2x} & g_{2y}\\g_{3x} & g_{3y}\\g_{4x} & g_{4y}\end{bmatrix}$.

Let $G_x=\begin{bmatrix}g_{1x}\\g_{2x}\\g_{3x}\\g_{4x}\end{bmatrix}$, $G_y=\begin{bmatrix}g_{1y}\\g_{2y}\\g_{3y}\\g_{4y}\end{bmatrix}$. Then $x(t)=T\cdot M\cdot G_x$, and $y(t)=T\cdot M\cdot G_y$, or

$$\begin{aligned}x(t)=&\left(t^3m_{11}+t^2m_{21}+tm_{31}+m_{41}\right)g_{1x}+\\&\left(t^3m_{12}+t^2m_{22}+tm_{32}+m_{42}\right)g_{2x}+\\&\left(t^3m_{13}+t^2m_{23}+tm_{33}+m_{43}\right)g_{3x}+\\&\left(t^3m_{14}+t^2m_{24}+tm_{34}+m_{44}\right)g_{4x}\end{aligned}\qquad\begin{aligned}y(t)=&\left(t^3m_{11}+t^2m_{21}+tm_{31}+m_{41}\right)g_{1y}+\\&\left(t^3m_{12}+t^2m_{22}+tm_{32}+m_{42}\right)g_{2y}+\\&\left(t^3m_{13}+t^2m_{23}+tm_{33}+m_{43}\right)g_{3y}+\\&\left(t^3m_{14}+t^2m_{24}+tm_{34}+m_{44}\right)g_{4y}\end{aligned}$$

$\Rightarrow$ A point on the curve is computed as a *sum of the geometric constraints weighted by the blending functions*:

$$Q(t)=\sum_i B_i(t)G_i$$

The weights $B_i(t)$ are cubic polynomials blending functions, $B=T\cdot M$.

For *parametric cubic polynomial curves* the method of specifying the geometric constraints gives the particular basis matrix, *M*.

Curve	Geometric constraints given by
Hermite	2 control points (endpoints of the curve) 2 tangent vectors at the control points (directly specified)
Cardinal	2 control points (endpoints of the curve) 2 points not on the curve that determine the endpoint tangent vectors
Bezier	2 control points (endpoints of the curve) 2 intermediate points not on the curve that determine the endpoint tangent vectors
B-Spline	4 control points (not on the curve)

Cubic Hermite spline (C^0, G^0)

Shape is determined by two control points, P_1 and P_4 (endpoints of the curve), and two tangent vectors at the control points, R_1 and R_4 (subscripts 1 and 4 are used for consistency with later notation).

The geometric constraints are therefore:

$$G_{\mathrm{H}} = \begin{bmatrix} P_{1_x} & P_{1_y} \\ P_{4_x} & P_{4_y} \\ R_{1_x} & R_{1_y} \\ R_{4_x} & R_{4_y} \end{bmatrix} \qquad G_{\mathrm{H}_x} = \begin{bmatrix} P_1 \\ P_4 \\ R_1 \\ R_4 \end{bmatrix}_x \qquad G_{\mathrm{H}_y} = \begin{bmatrix} P_1 \\ P_4 \\ R_1 \\ R_4 \end{bmatrix}_y$$

$$Q(t) = \begin{bmatrix} x(t) \\ y(t) \end{bmatrix} = \begin{bmatrix} a_x t^3 + b_x t^2 + c_x t + d_x \\ a_y t^3 + b_y t^2 + c_y t + d_y \end{bmatrix} = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} a_x & a_y \\ b_x & b_y \\ c_x & c_y \\ d_x & d_y \end{bmatrix} = T \cdot M_H \cdot G_H = \sum_i B_i(t) G_i \ .$$

We know G_{H}, but what is M_H ? To find the *Hermite basis matrix*, M_{H}, that relates the *Hermite geometric constraints*, G_{H}, to the polynomial coefficients $(a_x, b_x, \ldots)$, we write an equation in the four unknown coefficients, then solve for the unknowns using each geometric constraint:

$$x(t) = a_x t^3 + b_x t^2 + c_x t + d_x = T \cdot M_H \cdot G_{H_x}$$
$$x(t) = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \cdot M_H \cdot G_{H_x}$$
$$x'(t) = \begin{bmatrix} 3t^2 & 2t & 1 & 0 \end{bmatrix} \cdot M_H \cdot G_{H_x}$$

The endpoint constraints are given by:

$$x(0) = P_{1_x} = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix} \cdot M_H \cdot G_{H_x}$$
$$x(1) = P_{4_x} = \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix} \cdot M_H \cdot G_{H_x}$$

The tangent vector constraints are given by:

$$x'(0) = R_{1_x} = \begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix} \cdot M_H \cdot G_{H_x}$$
$$x'(1) = R_{4_x} = \begin{bmatrix} 3 & 2 & 1 & 0 \end{bmatrix} \cdot M_H \cdot G_{H_x}$$

Rewriting as a matrix equation:

$$G_{\mathrm{H}_x} = \begin{bmatrix} P_1 \\ P_4 \\ R_1 \\ R_4 \end{bmatrix}_x = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix} \cdot M_{\mathrm{H}} \cdot G_{\mathrm{H}_x}$$

Solving for M_H:

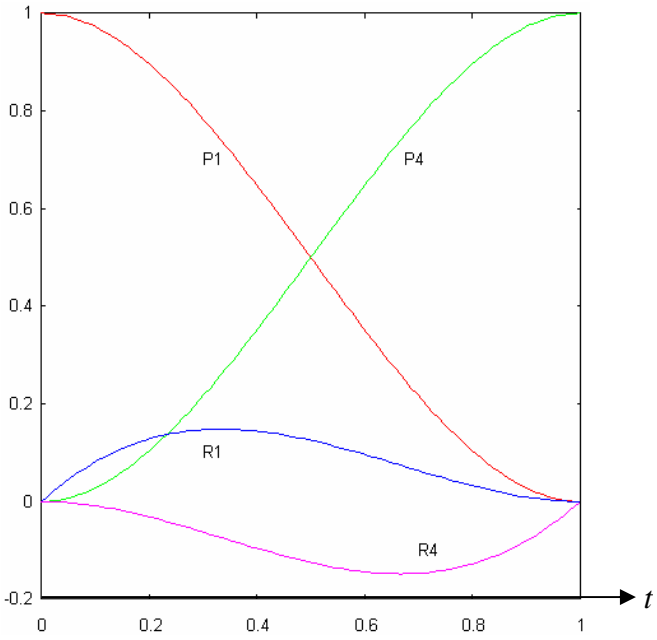
$$G_{H_x} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix} \cdot M_{\mathrm{H}} \cdot G_{H_x} \Rightarrow M_{\mathrm{H}} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix}^{-1} = \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}.$$

Since we now have G_H and M_H, we can solve $Q(t) = [x(t) \quad y(t)]$

$$Q(t) = T \cdot M_H \cdot G_H = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \cdot \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} P_1 \\ P_4 \\ R_1 \\ R_4 \end{bmatrix}, \quad \begin{matrix} x(t) = T \cdot M_H \cdot G_{H_x} \\ y(t) = T \cdot M_H \cdot G_{H_y} \end{matrix}$$

Carrying out the multiplication $B_H = (T \cdot M_H)$ gives the cubic *Hermite blending functions*, $B_H = T \cdot M_H$, the polynomials weighting the geometric constraints:

$$Q(t) = \underbrace{(2t^3 - 3t^2 + 1)}_{B_{H_0}(t)} P_1 + \underbrace{(-2t^3 + 3t^2)}_{B_{H_1}(t)} P_4 + \underbrace{(t^3 - 2t^2 + t)}_{B_{H_2}(t)} R_1 + \underbrace{(t^3 - t^2)}_{B_{H_3}(t)} R_4 = \sum_{i=0}^{3} B_{H_i}(t) P_i$$



Cubic Hermite blending functions

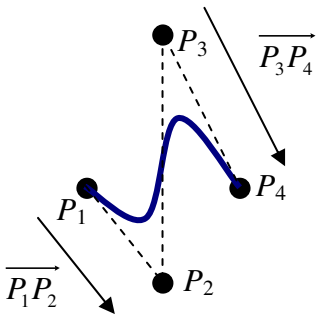
At $t = 0$, only $B_{H_0}(t)$ is nonzero, so the curve interpolates P_1. Likewise, at $t = 1$, only $B_{H_4}(t)$ is nonzero, so the curve interpolates P_4.

Cubic Bezier spline (C^0, G^0)

Whereas the cubic Hermite geometric constraints, G_H, consists of two points and two tangent vectors (P_1, P_4, R_1, R_4), the cubic Bezier geometric constraints, G_B, consists of four points (P_1, P_2, P_3, P_4), with endpoint tangent vectors specified indirectly: $\overrightarrow{P_1P_2}$ and $\overrightarrow{P_3P_4}$.

A Bezier curve defined by four points *interpolates* the two endpoints and *approximates* the intermediate two points:

$$G_B = \begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ P_4 \end{bmatrix}$$



The cubic Bezier endpoint tangents are related to the cubic Hermite endpoint tangents:

$$R_1 = 3(P_2 - P_1)$$
$$R_4 = 3(P_4 - P_3)$$

Using this relationship, we can write a matrix, M_{HB}, relating Hermite geometric constraints to Bezier geometric constraints:

$$G_H = \begin{bmatrix} P_1 \\ P_4 \\ R_1 \\ R_4 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ -3 & 3 & 0 & 0 \\ 0 & 0 & -3 & 3 \end{bmatrix} \begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ P_4 \end{bmatrix} = M_{HB} \cdot G_B$$

Starting as we did for the Hermite basis, $Q(t) = T \cdot M_H \cdot G_H$,
we substitute $M_{HB} \cdot G_B$ for G_H: $\quad Q(t) = T \cdot M_H \cdot (M_{HB} \cdot G_B)$.

Carrying out the multiplication, $M_H \cdot M_{HB}$, gives the cubic *Bezier basis matrix*, M_B:

$$M_H \cdot M_{HB} = \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ -3 & 3 & 0 & 0 \\ 0 & 0 & -3 & 3 \end{bmatrix} = \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} = M_B$$

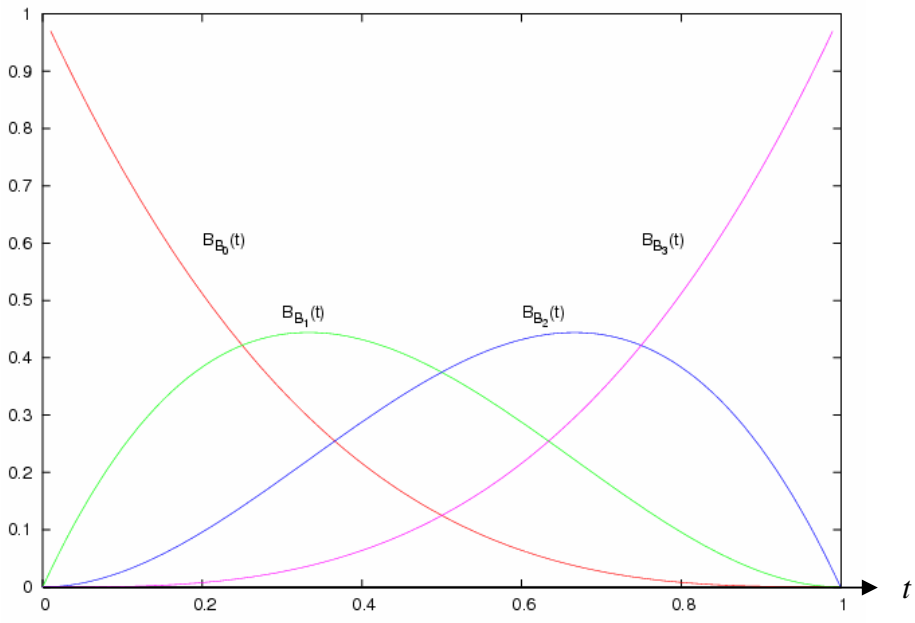
$$Q(t) = T \cdot M_B \cdot G_B = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \cdot \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ P_4 \end{bmatrix}.$$

The cubic *Bezier blending functions*, $B_B = T \cdot M_B$, are the polynomials weighting the geometric constraints:

$$Q(t) = (T \cdot M_B) \cdot G_B = B_B \cdot G_B$$

Carrying out the multiplication $B_B = (T \cdot M_B)$:

$$Q(t) = \underbrace{(1-t)^3}_{B_{B_0}(t)} P_1 + \underbrace{3t(1-t)^2}_{B_{B_1}(t)} P_2 + \underbrace{3t^2(1-t)}_{B_{B_2}(t)} P_3 + \underbrace{(t^3)}_{B_{B_3}(t)} P_4 = \sum_{i=0}^{3} B_{B_i}(t) P_i$$



Cubic Bezier blending functions.

At $t = 0$, only $B_{B_0}(t)$ is nonzero, so the curve interpolates P_1. Likewise, at $t = 1$, only $B_{B_4}(t)$ is nonzero, so the curve interpolates P_4.

Cubic B-spline: (C^2)

A B-spline approximates $n+1$ control points with a curve composed of $n-2$ segments, where $n \geq 3$. Each segment is defined by four control points (with $n \geq 3$, the minimum number of control points for a cubic B-spline is four, resulting in one segment).

The geometric constraints for a cubic B-spline with 4 control points, (P_1, P_2, P_3, P_4) is

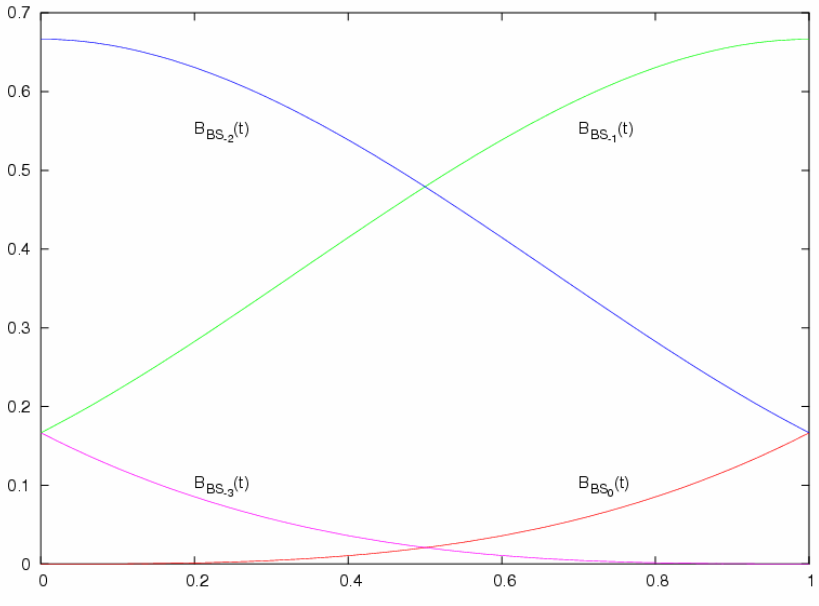
$$G_{BS} = \begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ P_4 \end{bmatrix}$$

The basis matrix is shown (but not derived):

$$Q(t) = T \cdot M_{BS} \cdot G_{BS} = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \cdot \frac{1}{6} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ P_4 \end{bmatrix}$$

Carrying out the multiplication $B_{\mathrm{BS}} = (T \cdot M_{\mathrm{BS}})$:

$$Q(t) = \underbrace{\tfrac{1}{6}(1-t)^3}_{B_{BS_0}(t)} P_1 + \underbrace{\tfrac{1}{6}(3t^3 - 6t^2 + 4)}_{B_{BS_1}(t)} P_2 + \underbrace{\tfrac{1}{6}(-3t^3 + 3t^2 + 3t + 1)}_{B_{BS_2}(t)} P_3 + \underbrace{\tfrac{1}{6}(t^3)}_{B_{BS_3}(t)} P_4 = \sum_{i=0}^{3} B_{\mathrm{BS}_i}(t) P_i$$



Cubic B-spline blending functions

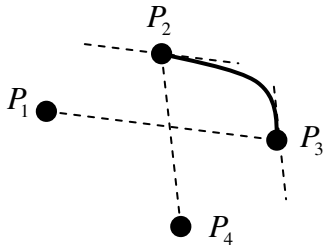
B-splines exhibit *local control*: a point on the curve is influenced by only four control points. Hermite and Bezier curves exhibit *global control*: moving any control point affects the entire curve.

Cubic Cardinal spline

Shape is determined by: two control points, P_2 and P_3, the endpoints of the curve, two points not on the curve, P_1 and P_4, that determine the endpoint tangent vectors, and a *tension* parameter

The tangent at endpoint P_i is proportional to the chord between the adjacent points:

Tangent at P_2: $\frac{1}{2}(1-a)(P_3 - P_1)$
Tangent at P_3: $\frac{1}{2}(1-a)(P_4 - P_2)$



The parameter *a* is called the *tension*.
Defining $\tau = \frac{1}{2}(1-a)$, the following Cardinal spline basis matrix and blending functions can be derived:

$$M_{CD} = \begin{bmatrix} -\tau & 2-\tau & \tau-2 & \tau \\ 2\tau & \tau-3 & 3-2\tau & -\tau \\ -\tau & 0 & \tau & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}, \quad B_i(t) = T \cdot M_{CD}$$

For the case where $a = 0$ $(\tau = \frac{1}{2})$, the spline is called a *Catmull-Rom* spline (or *Overhauser* spline), with the following basis matrix:

$$M_{CR} = \frac{1}{2}\begin{bmatrix} -1 & 3 & -3 & 1 \\ 2 & -5 & 4 & -1 \\ -1 & 0 & 1 & 0 \\ 0 & 2 & 0 & 0 \end{bmatrix}$$

(Note that when $a = 1$ the spline degenerates to a straight line).

Catmull-Rom: Edwin Catmull, *A Subdivision Algorithm for Computer Display of Curved Surfaces.*. PhD dissertation, Department of Computer Science, University of Utah, Salt Lake City, Dec 1974.

Overhauser: A.W. Overhauser, *Analytic Definition of Curves and Surfaces by Parabolic blending*. Scientific Research Staff Publication, Ford Motor Company, May 1968.

Bezier: Bezier, P.E., *How Renault Uses Numerical Control for Car Body Design and Tooling*, SAE Paper 680010, Society of Automotive Engineers Congress, Detroit, MI, 1968.