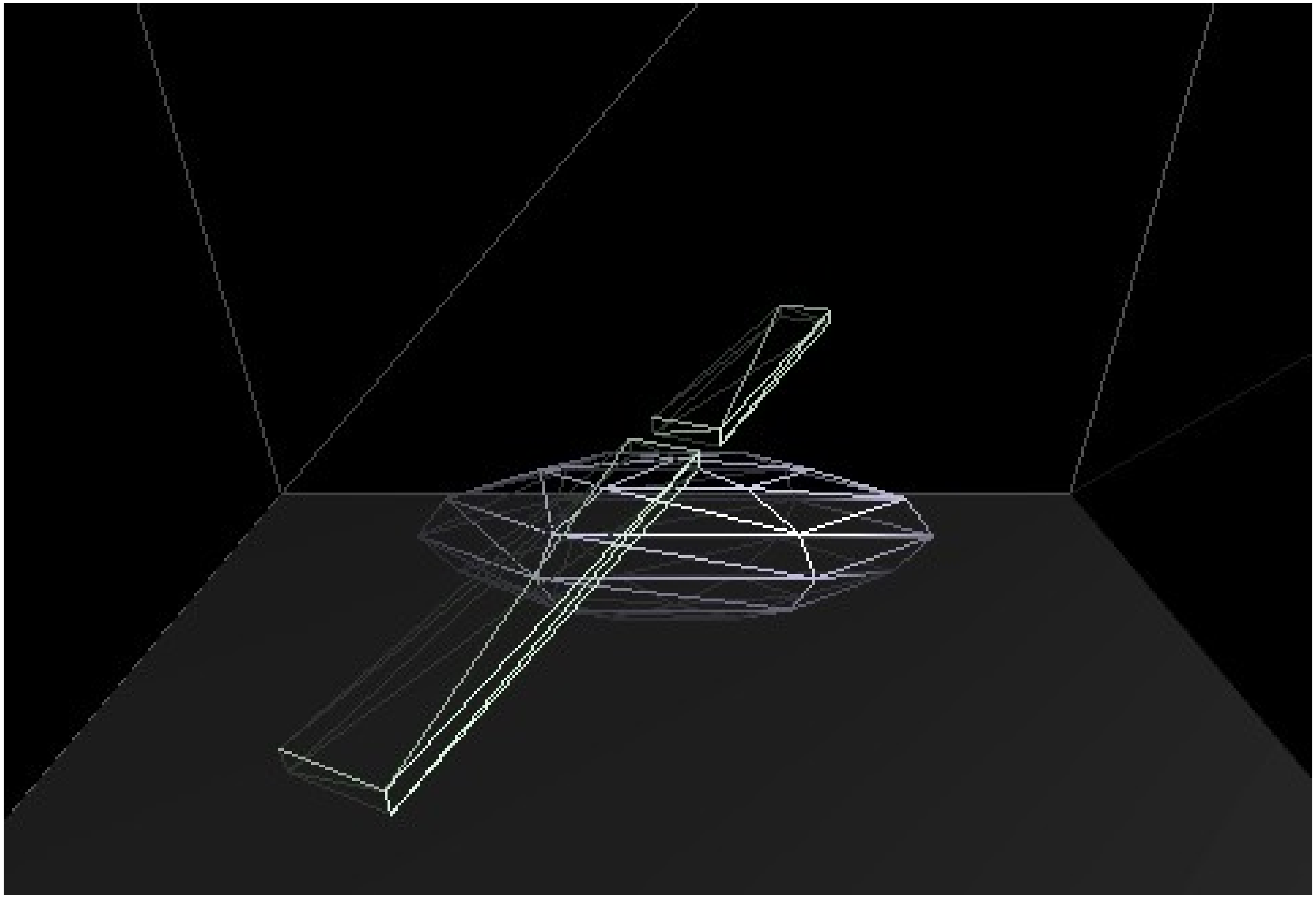
Computer Graphics

MTAT.03.015

Raimond Tunnel

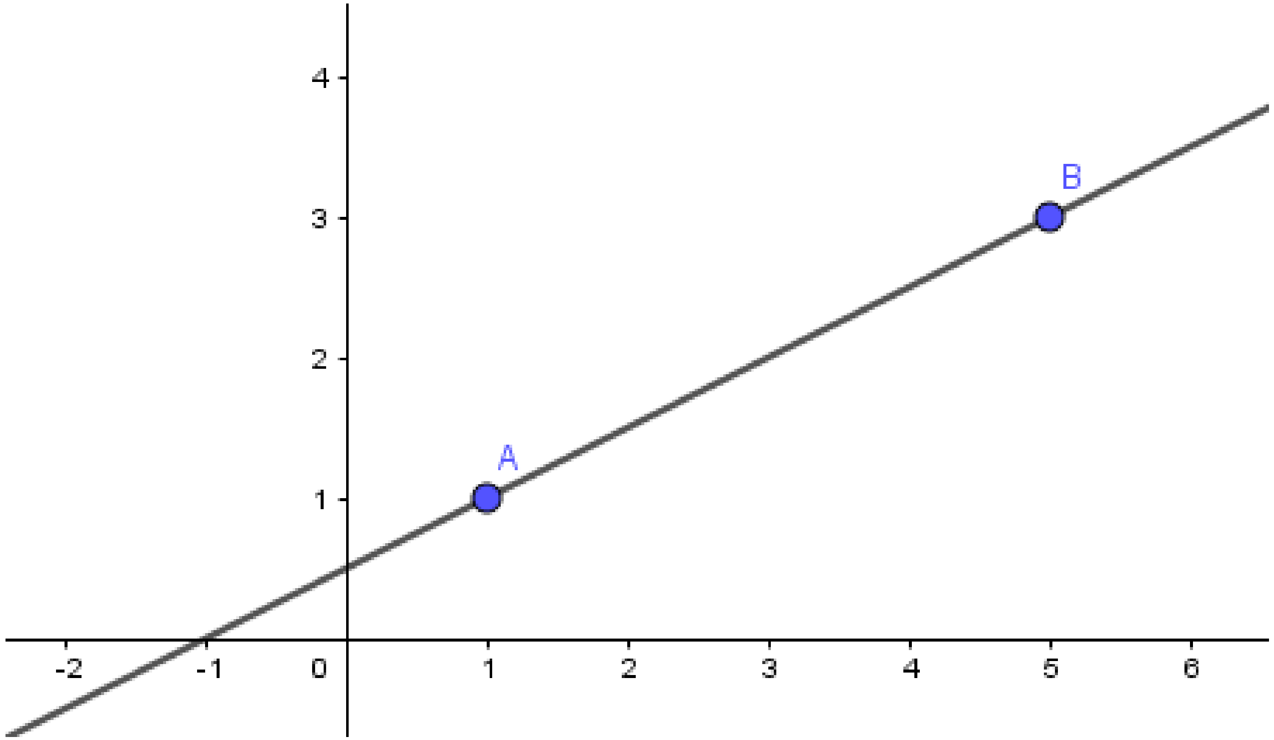
Study IT in .ee

The Road So Far...



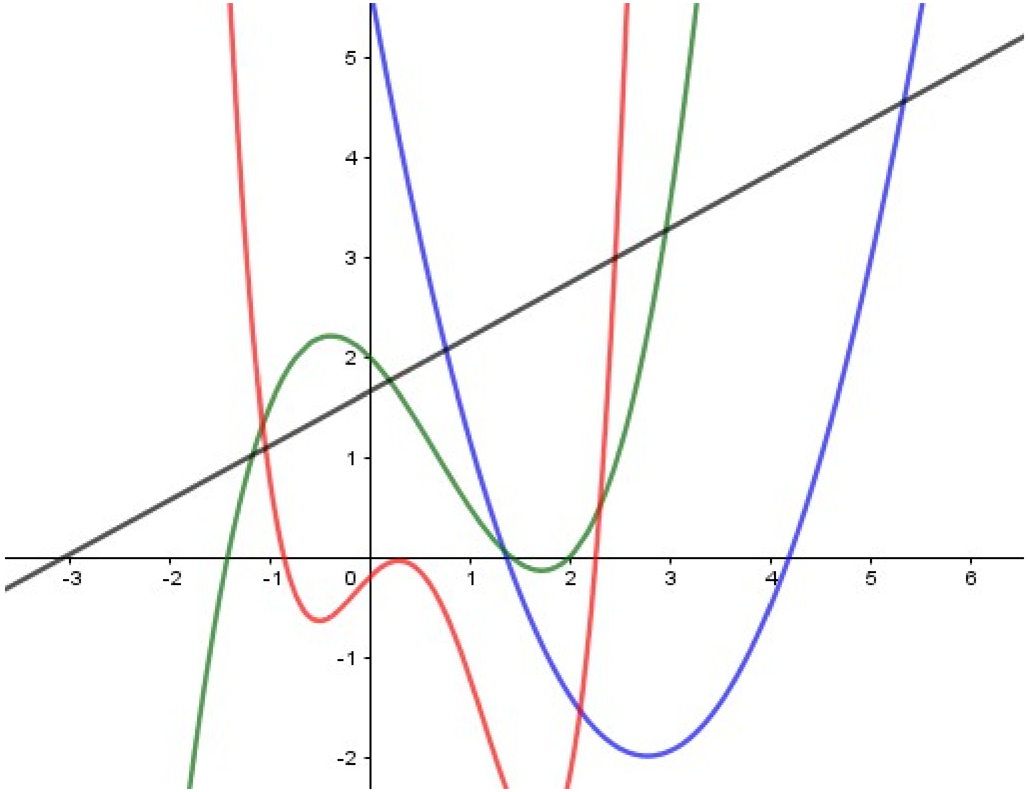
Curves

- Line interpolates between 2 points.



Curves

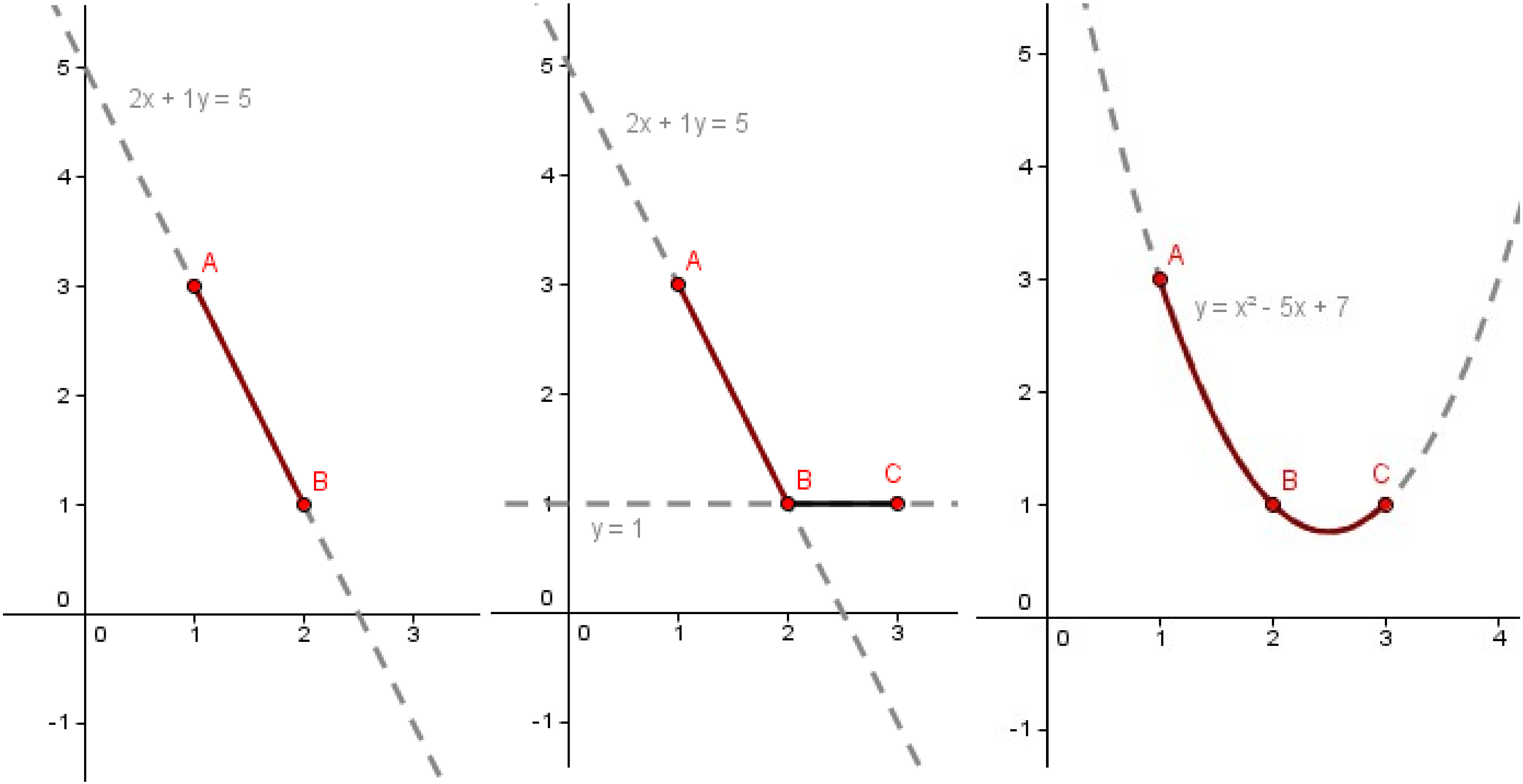
- Line interpolates between 2 points.
- Mathematically there are higher polynomials to interpolate between more points



Curves

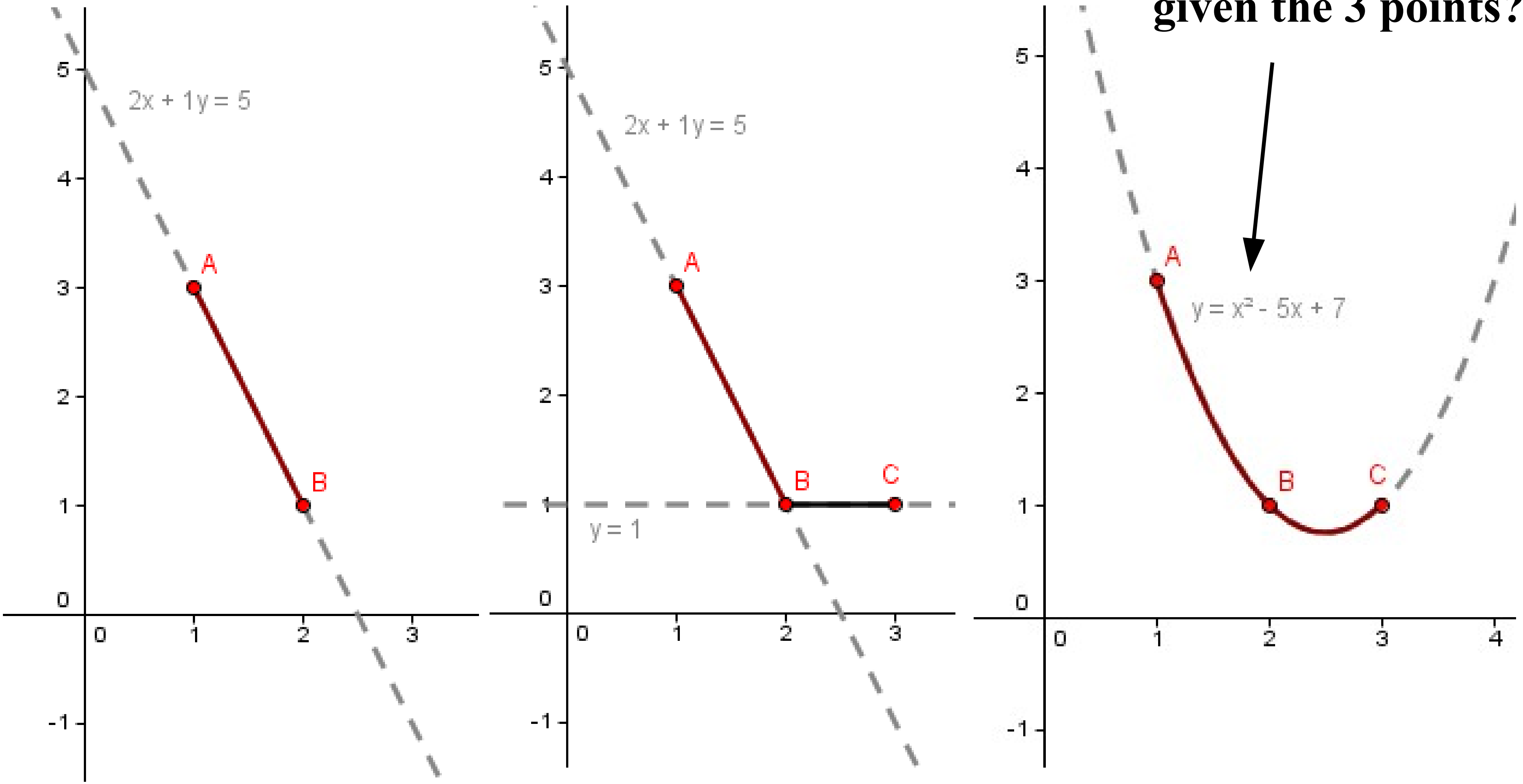
- Line interpolates between 2 points.
- Mathematically there are higher polynomials to interpolate between more points
- How many points you need, to construct a *n*-th degree polynomial through it?

Curves



Curves

How to find that parabola given the 3 points?



Curves

- Constructing a parabola through 3 points:

$$f(x) = a_2 \cdot x^2 + a_1 \cdot x + a_0$$

Just like we found a line equation in the first lectures

Curves

- Constructing a parabola through 3 points:

$$f(x) = a_2 \cdot x^2 + a_1 \cdot x + a_0$$

- We are looking for: $a_{2,}\ \ a_{1,}\ \ a_0.$

Curves

- Constructing a parabola through 3 points:

$$f(x) = a_2 \cdot x^2 + a_1 \cdot x + a_0$$

- We are looking for: $a_{2,}\ \ a_{1,}\ \ a_0$.
- We know: $f(1) = 3,\ \ f(2) = 1,\ \ f(3) = 1$

Curves

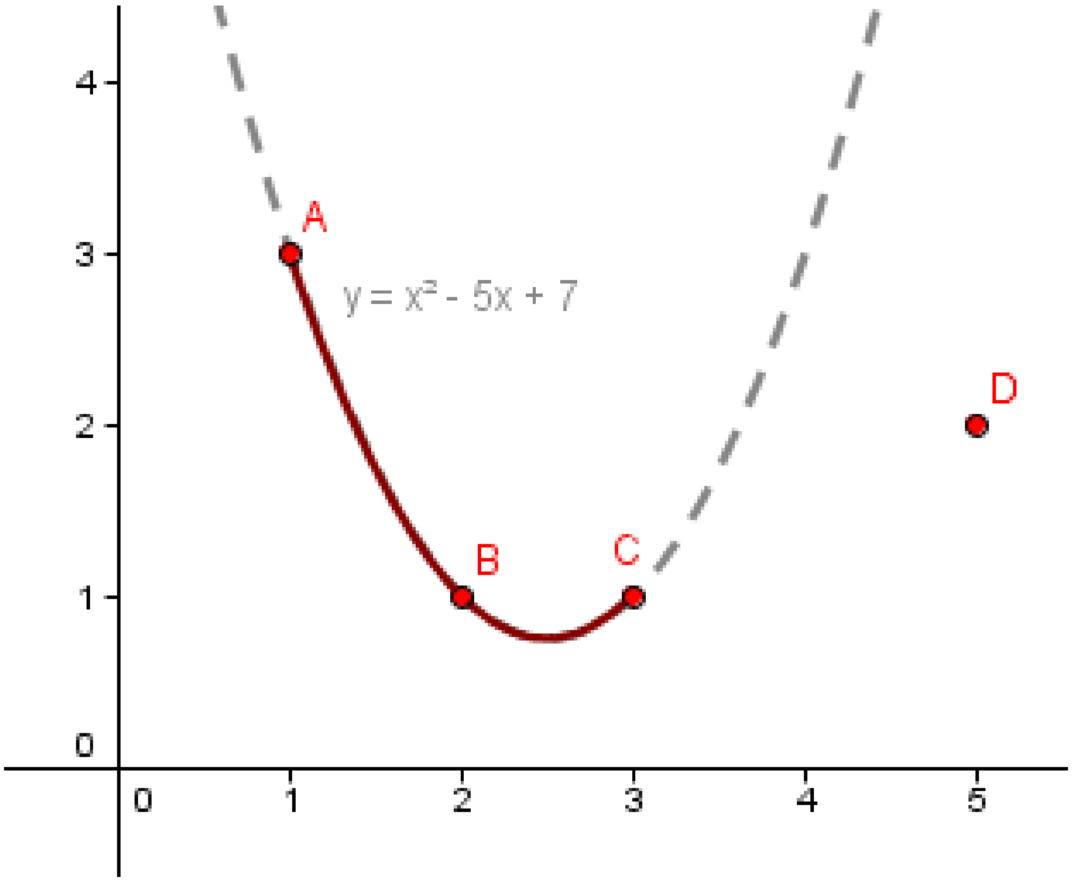
- Constructing a parabola through 3 points:

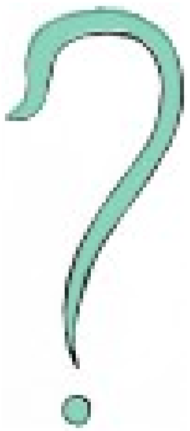
$$f(x) = a_2 \cdot x^2 + a_1 \cdot x + a_0$$

- We are looking for: $a_{2,} \; a_{1,} \; a_0.$
- We know: $f(1) = 3, \; f(2) = 1, \; f(3) = 1$
- **3 unknowns, 3 constraints, we can solve it.**
- http://www.wolframalpha.com/input/?i=a*1+%2B+b*1+%2B+c+%3D+3%2C+a*4+%2B+b*2+%2B+c+%3D+1%2C+a*9+%2B+b*3+%2B+c+%3D+1

Curves

- What choices we have with 4 points?





One additional point meant another line, could we have 2 parabolas here?

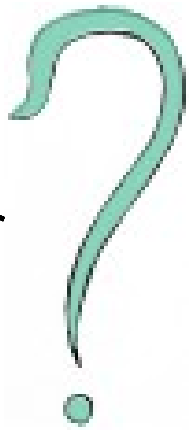
Curves

- Contstraints do not have to be on the function.

Curves

- Contstraints do not have to be on the function.
- They can also be on the derivative of it.

$$g(x) = a_2 \cdot x^2 + a_1 \cdot x + a_0 \qquad g'(x) = ?$$



Curves

- Contstraints do not have to be on the function.
- They can also be on the derivative of it.

$$g(x) = a_2 \cdot x^2 + a_1 \cdot x + a_0 \qquad g'(x) = ?$$

- Constraints:

$$g(3) = 1$$

$$g(5) = 2$$

$$g'(3) = f'(3) = ?$$

Curves

- Contstraints do not have to be on the function.
- They can also be on the derivative of it.

$$g(x) = a_2 \cdot x^2 + a_1 \cdot x + a_0 \qquad g'(x) = ?$$

- Constraints:

$$g(3) = 1$$

$$g(5) = 2$$

$$g'(3) = f'(3) = 1$$

Curves

- Contstraints do not have to be on the function.
- They can also be on the derivative of it.

$$g(x) = a_2 \cdot x^2 + a_1 \cdot x + a_0 \qquad g'(x) = ?$$

- Constraints:

$$g(3) = 1$$

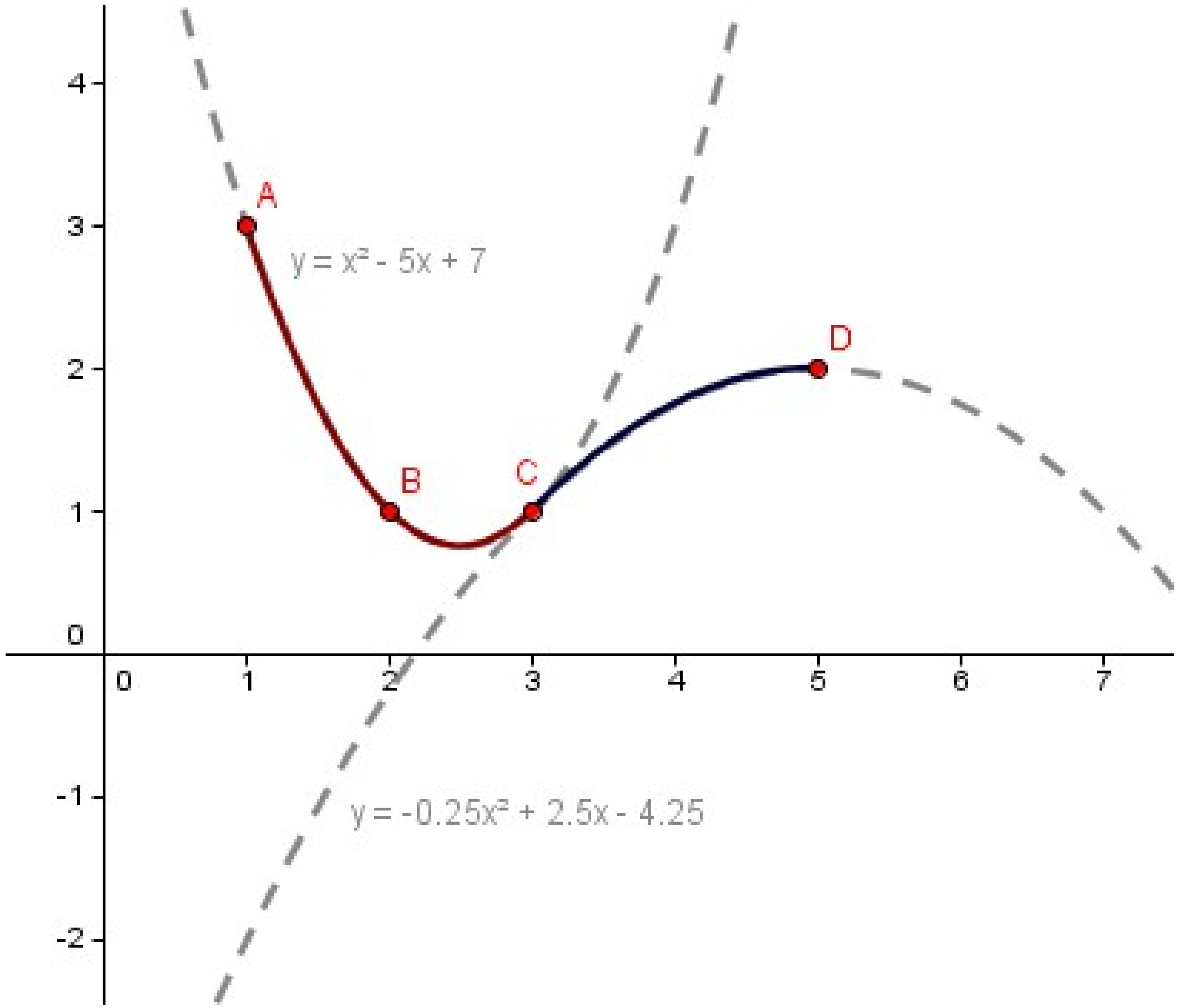
$$g(5) = 2$$

$$g'(3) = f'(3) = 1$$

http://www.wolframalpha.com/input/?i=9a%2B3b+%2B+c%3D1%2C+25a%2B5b%2Bc%3D2%2C+6a%2Bb%3D1

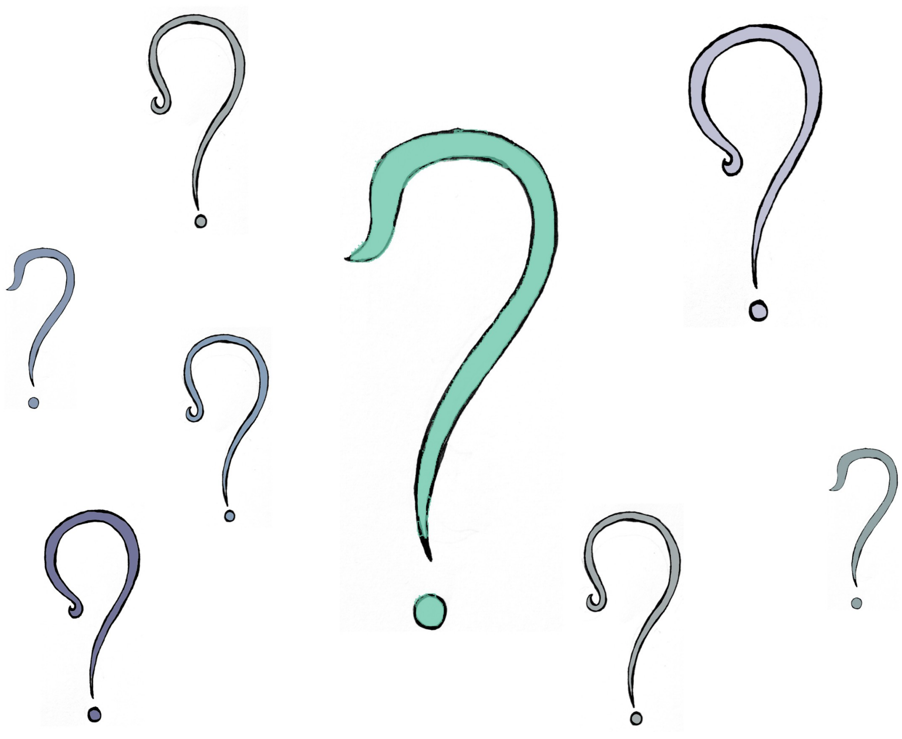
- **3 unknowns, 3 constraints, we can solve it.**

Curves



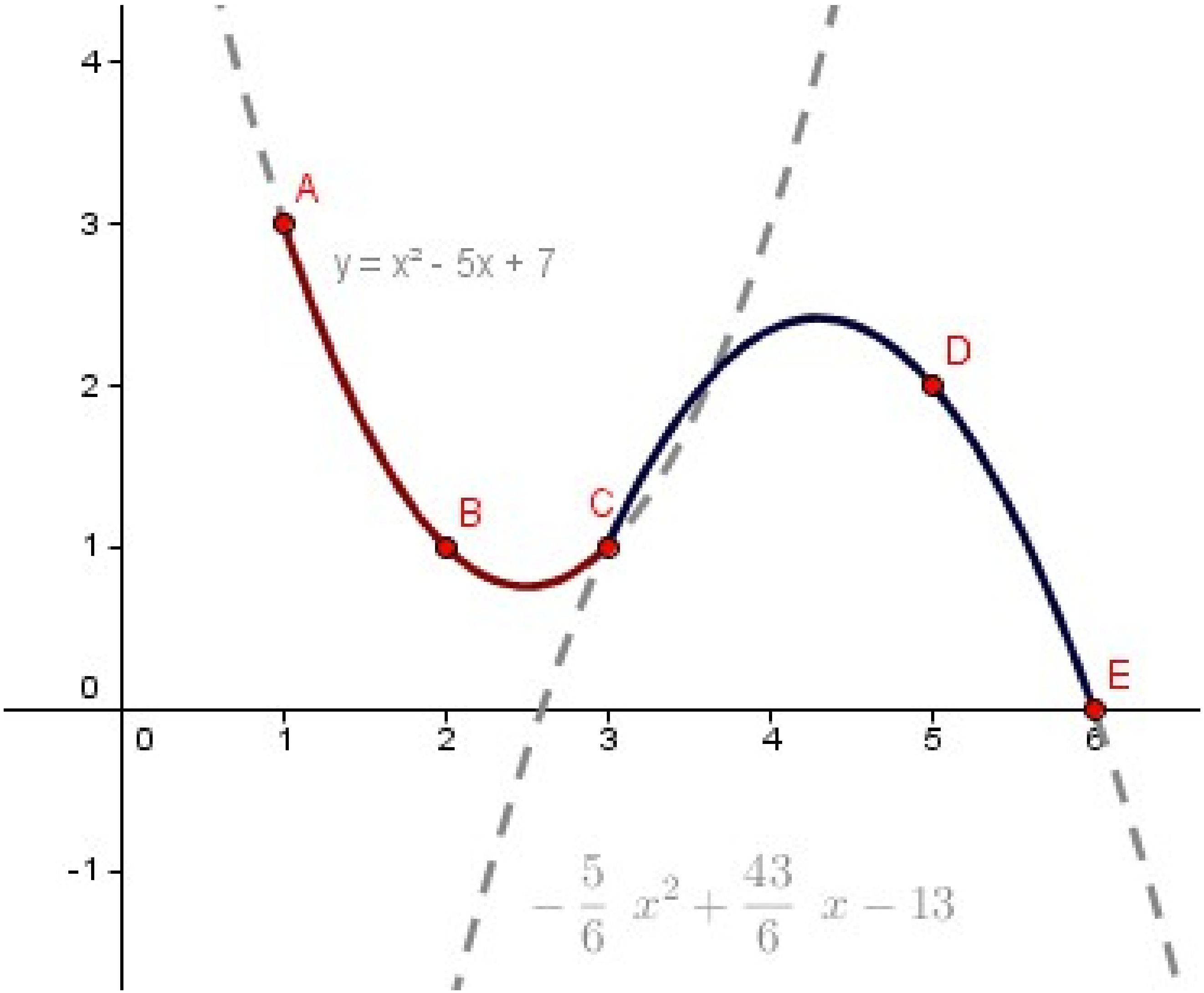
Smoothness

- What if we have 5 points and we put two parabolas through them without accounting for the derivative?



Smoothness

- That spline is not C^1 smooth.



Smoothness (continuity)

- **Spline** – one or many connected curves.

Smoothness (continuity)

- **Spline** – one or many connected curves.
- C^n **smoothness** – the *n*-th derivative is continuous everywhere along the object and the object is also C^{n-1} smooth.

The differentiability class

Smoothness (continuity)

- **Spline** – one or many connected curves.
- C^n **smoothness** – the *n*-th derivative is continuous everywhere along the object and the object is also C^{n-1} smooth.

The differentiability class

The continuty class?

Smoothness (continuity)

- **Spline** – one or many connected curves.
- C^n **smoothness** – the *n*-th derivative is continuous everywhere along the object and the object is also C^{n-1} smooth.

- For **parametric curves**, we can also talk about:
  - G^n smoothness (geometric smoothness) – the *n*-th derivative can have sudden jumps in magnitude, but not the direction.

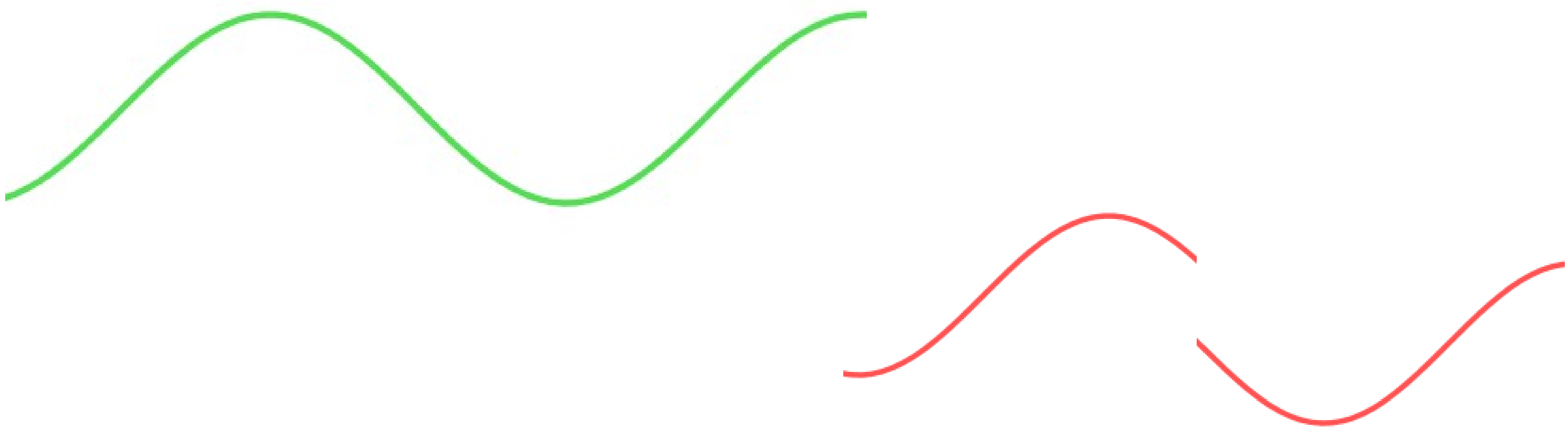
Smoothness (continuity)

- **Spline** – one or many connected curves.
- C^n **smoothness** – the *n*-th derivative is continuous everywhere along the object and the object is also C^{n-1} smooth.

- For **parametric curves**, we can also talk about:
  - G^n smoothness (geometric smoothness) – the *n*-th derivative can have sudden jumps in magnitude, but not the direction.
    And the object is of G^{n-1}

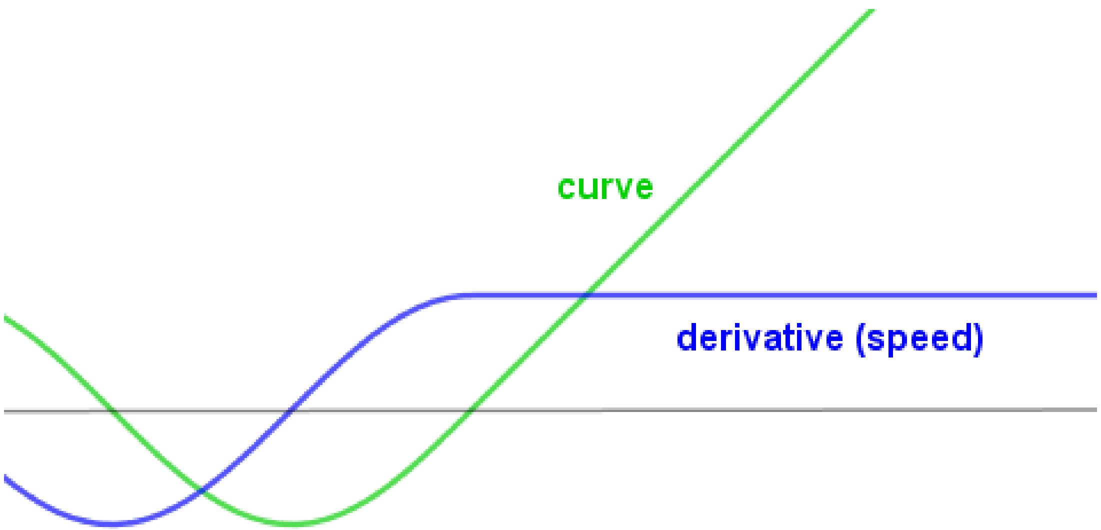
Smoothness (continuity)

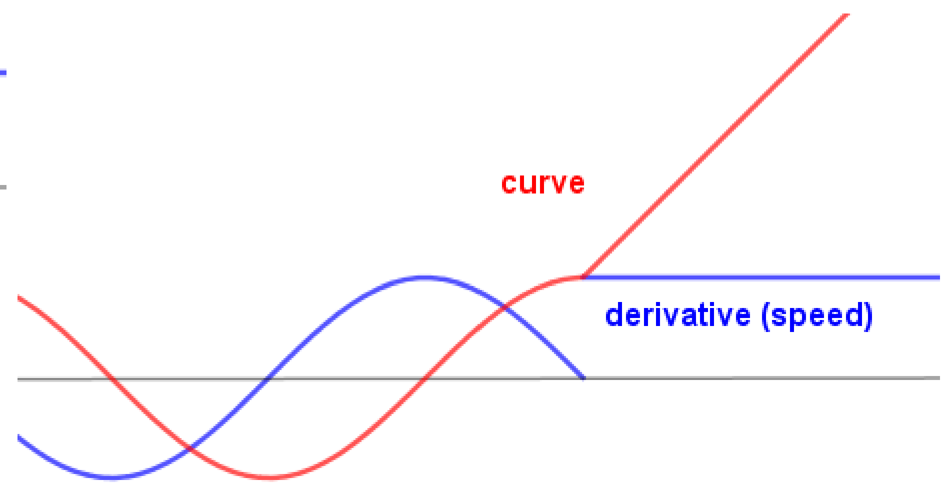
- Different levels of smoothness:
  - C^0 – Curve itself is continuous



Smoothness (continuity)

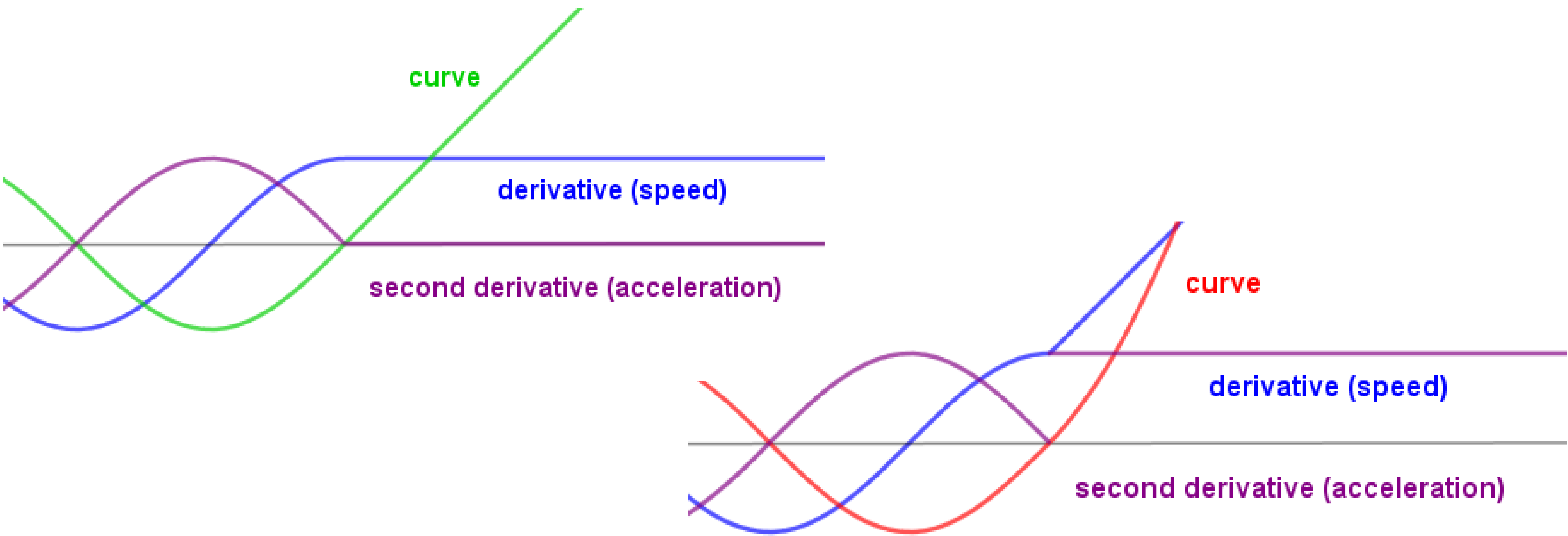
- Different levels of smoothness:
  - C^0 – Curve itself is continuous
  - C^1 – First derivative (speed) is continuous





Smoothness (continuity)

- Different levels of smoothness:
  - C^0 – Curve itself is continuous
  - C^1 – First derivative (speed) is continuous
  - C^2 – Second derivative (acceleration) is continuous



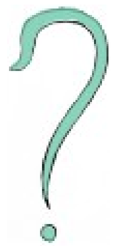
Smoothness (continuity)

- Different levels of smoothness:
  - C^0 – Curve itself is continuous
  - C^1 – First derivative (speed) is continuous
  - C^2 – Second derivative (acceleration) is continuous
- Often times C^1 or C^2 smooth curves are enough in computer graphics.

Smoothness (continuity)

- Different levels of smoothness:
  - C^0 – Curve itself is continuous
  - C^1 – First derivative (speed) is continuous
  - C^2 – Second derivative (acceleration) is continuous
- Often times C^1 or C^2 smooth curves are enough in computer graphics.
- If we put quadratic curves together, so that the spline is C^1 smooth, how to get C^2 smoothness?

Find the second derivatives of our previous example...



Parametric Curves

- Implicit form: $f(x) = a_2 \cdot x^2 + a_1 \cdot x + a_0$
  - Good for testing points in a curve
  - Finding collisions

Parametric Curves

- Implicit form: $f(x) = a_2 \cdot x^2 + a_1 \cdot x + a_0$
  - Good for testing points in a curve
  - Finding collisions

For your regular mathematical quadratic fun.

Parametric Curves

- Implicit form: $f(x)=a_2 \cdot x^2+a_1 \cdot x+a_0$
  - Good for testing points in a curve
  - Finding collisions
- Parametric form: $g(t)=(t+x_0,\ a_2 \cdot t^2+y_0)=(x,\ y)$

$$x_0=\frac{-a_1}{2 \cdot a_2},\quad y_0=f(x_0)$$

  - Good for **generating points on the curve**

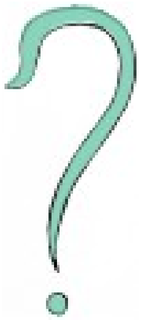
Parametric form of a quadratic polynomial: http://www.nabla.hr/PC-ParametricEqu2.htm

Parametric Curves

- Implicit form: $f(x) = a_2 \cdot x^2 + a_1 \cdot x + a_0$
  - Good for testing points in a curve
  - Finding collisions
- Parametric form: $g(t) = (t + x_0,\ a_2 \cdot t^2 + y_0) = (x,\ y)$

$$x_0 = \frac{-a_1}{2 \cdot a_2},\quad y_0 = f(x_0)$$

  - Good for generating points on the curve
  - **What other parametric equations you know?**



Parametric Curve Construction

- We want to find the **vector coefficients** a_i for a **function of *t* (time)**, where $t \in [0..1]$.

Quadratic: $curve(t) = a_0 + a_1 \cdot t + a_2 \cdot t^2$

Cubic: $curve(t) = a_0 + a_1 \cdot t + a_2 \cdot t^2 + a_3 \cdot t^3$

Parametric Curve Construction

- We want to find the vector coefficients a_i for a function of t (time), where $t \in [0..1]$.

Quadratic: $$curve(t) = a_0 + a_1 \cdot t + a_2 \cdot t^2$$

Cubic: $$curve(t) = a_0 + a_1 \cdot t + a_2 \cdot t^2 + a_3 \cdot t^3$$

- We need to have constraints. For example, the curve must **interpolate a number of 2D points**.

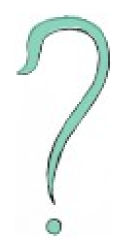
Parametric Curve Construction

- We want to find the vector coefficients a_i for a function of t (time), where $t \in [0..1]$.

Quadratic: $$curve(t) = a_0 + a_1 \cdot t + a_2 \cdot t^2$$

Cubic: $$curve(t) = a_0 + a_1 \cdot t + a_2 \cdot t^2 + a_3 \cdot t^3$$

- We need to have constraints. For example, the curve must **interpolate a number of 2D points**.
- **How many points we need?**

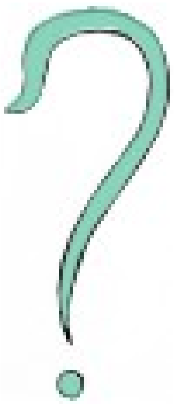


Parametric Curve Construction

- For a quadratic curve in 2D we need 3 points. Each 2D point gives 2 1D constraints.

Parametric Curve Construction

- For a quadratic curve in 2D we need 3 points. Each 2D point gives 2 1D constraints.
- What about in 3D? Cubic?



Parametric Curve Construction

- For a quadratic curve in 2D we need 3 points. Each 2D point gives 2 1D constraints.
- What about in 3D? Cubic?

$$curve(0) = (1,\ 3) = \boldsymbol{p_0}$$

$$curve(0.5) = (2,\ 1) = \boldsymbol{p_1}$$

$$curve(1) = (3,\ 1) = \boldsymbol{p_2}$$

Control points

Parametric Curve Construction

- For a quadratic curve in 2D we need 3 points. Each 2D point gives 2 1D constraints.
- What about in 3D? Cubic?

$$curve(0) = (1,\ 3) = p_0$$
$$curve(0.5) = (2,\ 1) = p_1$$
$$curve(1) = (3,\ 1) = p_2$$

- Usually the system of constraints is written in a **constraint matrix**.

Parametric Curve Construction

- Constraint matrix

$$\begin{pmatrix} 1 & 0 & 0 \\ 1 & 0.5 & 0.25 \\ 1 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix} = \begin{pmatrix} p_0 \\ p_1 \\ p_2 \end{pmatrix}$$

Parametric Curve Construction

- Constraint matrix

In short: $C \cdot a = p$

$$\begin{pmatrix} 1 & 0 & 0 \\ 1 & 0.5 & 0.25 \\ 1 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix} = \begin{pmatrix} p_0 \\ p_1 \\ p_2 \end{pmatrix}$$

- Write out the equations to see, that this is exactly what we did before with the implicit eq.

Parametric Curve Construction

- Constraint matrix

In short: $C \cdot a = p$

$$\begin{pmatrix} 1 & 0 & 0 \\ 1 & 0.5 & 0.25 \\ 1 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix} = \begin{pmatrix} p_0 \\ p_1 \\ p_2 \end{pmatrix}$$

- Write out the equations to see, that this is exactly what we did before with the implicit eq.
- Only now the a_i and p_i are vectors.

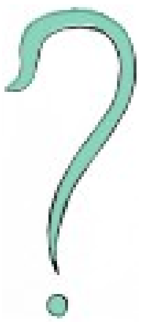
Parametric Curve Construction

- Constraint matrix

In short: $C \cdot \boldsymbol{a} = p$

$$\begin{pmatrix} 1 & 0 & 0 \\ 1 & 0.5 & 0.25 \\ 1 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} \boldsymbol{a_0} \\ \boldsymbol{a_1} \\ \boldsymbol{a_2} \end{pmatrix} = \begin{pmatrix} p_0 \\ p_1 \\ p_2 \end{pmatrix}$$

How to find $\boldsymbol{a}$?



- Write out the equations to see, that this is exactly what we did before.
- Only now the a_i and p_i are vectors.

Parametric Curve Construction

- We can find $a = C^{-1} p$, the inverse constraint matrix C^{-1} is often denoted B and called the **basis / blending matrix**.

Parametric Curve Construction

- We can find $a = C^{-1} p$, the inverse constraint matrix C^{-1} is often denoted B and called the **basis / blending matrix**.

In our example:

$$B = \begin{pmatrix} 1 & 0 & 0 \\ -3 & 4 & -1 \\ 2 & -4 & 2 \end{pmatrix}$$

Inverse (and other stuff) matrix calculator: http://www.bluebit.gr/matrix-calculator/

Parametric Curve Construction

- We can find $a = C^{-1} p$, the inverse constraint matrix C^{-1} is often denoted B and called the **basis / blending matrix**.
- Now we know the coefficients in:

$$curve(t) = a_0 + a_1 \cdot t + a_2 \cdot t^2$$

In our example:

$$B = \begin{pmatrix} 1 & 0 & 0 \\ -3 & 4 & -1 \\ 2 & -4 & 2 \end{pmatrix}$$

Parametric Curve Construction

- We can find $a = C^{-1} p$, the inverse constraint matrix C^{-1} is often denoted B and called the **basis / blending matrix**.
- Now we know the coefficients in:

$$curve(t) = a_0 + a_1 \cdot t + a_2 \cdot t^2$$

$$a_0 = b_{0,0} p_0 + b_{0,1} p_1 + b_{0,2} p_2$$
$$a_1 = b_{1,0} p_0 + b_{1,1} p_1 + b_{1,2} p_2$$
$$a_2 = b_{2,0} p_0 + b_{2,1} p_1 + b_{2,2} p_2$$

In our example:

$$B = \begin{pmatrix} 1 & 0 & 0 \\ -3 & 4 & -1 \\ 2 & -4 & 2 \end{pmatrix}$$

Parametric Curve Construction

- We can find $a = C^{-1} p$, the inverse constraint matrix C^{-1} is often denoted B and called the **basis / blending matrix**.
- Now we know the coefficients in:

$$curve(t) = a_0 + a_1 \cdot t + a_2 \cdot t^2$$

$$a_0 = \boldsymbol{b}_{\mathbf{0,0}}\, p_0 + \boldsymbol{b}_{\mathbf{0,1}}\, p_1 + \boldsymbol{b}_{\mathbf{0,2}}\, p_2$$

$$a_1 = \boldsymbol{b}_{\mathbf{1,0}}\, p_0 + \boldsymbol{b}_{\mathbf{1,1}}\, p_1 + \boldsymbol{b}_{\mathbf{1,2}}\, p_2$$

$$a_2 = \boldsymbol{b}_{\mathbf{2,0}}\, p_0 + \boldsymbol{b}_{\mathbf{2,1}}\, p_1 + \boldsymbol{b}_{\mathbf{2,2}}\, p_2$$

In our example:

$$\boldsymbol{B} = \begin{pmatrix} 1 & 0 & 0 \\ -3 & 4 & -1 \\ 2 & -4 & 2 \end{pmatrix}$$

Parametric Curve Construction

- We can find $a = C^{-1} p$, the inverse constraint matrix C^{-1} is often denoted B and called the **basis / blending matrix**.
- Now we know the coefficients in:

$$curve(t) = a_0 + a_1 \cdot t + a_2 \cdot t^2$$

$$a_0 = b_{0,0}\,\boldsymbol{p_0} + b_{0,1}\,\boldsymbol{p_1} + b_{0,2}\,\boldsymbol{p_2}$$

$$a_1 = b_{1,0}\,\boldsymbol{p_0} + b_{1,1}\,\boldsymbol{p_1} + b_{1,2}\,\boldsymbol{p_2}$$

$$a_2 = b_{2,0}\,\boldsymbol{p_0} + b_{2,1}\,\boldsymbol{p_1} + b_{2,2}\,\boldsymbol{p_2}$$

In our example:

$$B = \begin{pmatrix} 1 & 0 & 0 \\ -3 & 4 & -1 \\ 2 & -4 & 2 \end{pmatrix}$$

Parametric Curve Construction

- Let us look at the entire curve:

$$curve(t) =$$
$$= b_{0,0}\,p_0 + b_{0,1}\,p_1 + b_{0,2}\,p_2 +$$
$$+ b_{1,0}\,p_0\,t + b_{1,1}\,p_1\,t + b_{1,2}\,p_2\,t +$$
$$+ b_{2,0}\,p_0\,t^2 + b_{2,1}\,p_1\,t^2 + b_{2,2}\,p_2\,t^2$$

Parametric Curve Construction

- Let us look at the entire curve:

$$curve(t) =$$
$$= b_{0,0}\, p_0 + b_{0,1}\, p_1 + b_{0,2}\, p_2 +$$
$$+ b_{1,0}\, p_0 t + b_{1,1}\, p_1 t + b_{1,2}\, p_2 t +$$
$$+ b_{2,0}\, p_0 t^2 + b_{2,1}\, p_1 t^2 + b_{2,2}\, p_2 t^2$$

- We can rewrite it as:

$$curve(t) = b_0(t) \cdot p_0 + b_1(t) \cdot p_1 + b_2(t) \cdot p_2$$

Parametric Curve Construction

- Let us look at the entire curve:

$$
\begin{aligned}
&curve(t) = \\
&\quad = b_{0,0}\,\boldsymbol{p_0} + b_{0,1}\,\boldsymbol{p_1} + b_{0,2}\,\boldsymbol{p_2} + \\
&\quad + b_{1,0}\,\boldsymbol{p_0}\,t + b_{1,1}\,\boldsymbol{p_1}\,t + b_{1,2}\,\boldsymbol{p_2}\,t + \\
&\quad + b_{2,0}\,\boldsymbol{p_0}\,t^2 + b_{2,1}\,\boldsymbol{p_1}\,t^2 + b_{2,2}\,\boldsymbol{p_2}\,t^2
\end{aligned}
$$

- We can rewrite it as:

$$curve(t) = b_0(t)\cdot\boldsymbol{p_0} + b_1(t)\cdot\boldsymbol{p_1} + b_2(t)\cdot\boldsymbol{p_2}$$

Parametric Curve Construction

- Let us look at the entire curve:

$$curve(t) =$$
$$= \boldsymbol{b_{0,0}}\, p_0 + b_{0,1}\, p_1 + b_{0,2}\, p_2 +$$
$$+ \boldsymbol{b_{1,0}}\, p_0 \boldsymbol{t} + b_{1,1}\, p_1 t + b_{1,2}\, p_2 t +$$
$$+ \boldsymbol{b_{2,0}}\, p_0 \boldsymbol{t^2} + b_{2,1}\, p_1 t^2 + b_{2,2}\, p_2 t^2$$

- We can rewrite it as:

$$curve(t) = \boldsymbol{b_0(t)} \cdot p_0 + b_1(t) \cdot p_1 + b_2(t) \cdot p_2$$

Parametric Curve Construction

- Let us look at the entire curve:

$$curve(t) =$$
$$= b_{0,0}\, p_0 + \boldsymbol{b_{0,1}}\, p_1 + b_{0,2}\, p_2 +$$
$$+ b_{1,0}\, p_0 t + \boldsymbol{b_{1,1}}\, p_1 \boldsymbol{t} + b_{1,2}\, p_2 t +$$
$$+ b_{2,0}\, p_0 t^2 + \boldsymbol{b_{2,1}}\, p_1 \boldsymbol{t^2} + b_{2,2}\, p_2 t^2$$

- We can rewrite it as:

$$curve(t) = b_0(t) \cdot p_0 + \boldsymbol{b_1(t)} \cdot p_1 + b_2(t) \cdot p_2$$

Parametric Curve Construction

- Let us look at the entire curve:

$$curve(t) =$$
$$= b_{0,0}\, p_0 + b_{0,1}\, p_1 + \boldsymbol{b_{0,2}}\, p_2 +$$
$$+ b_{1,0}\, p_0 t + b_{1,1}\, p_1 t + \boldsymbol{b_{1,2}}\, p_2 \boldsymbol{t} +$$
$$+ b_{2,0}\, p_0 t^2 + b_{2,1}\, p_1 t^2 + \boldsymbol{b_{2,2}}\, p_2 \boldsymbol{t^2}$$

- We can rewrite it as:

$$curve(t) = b_0(t) \cdot p_0 + b_1(t) \cdot p_1 + \boldsymbol{b_2(t)} \cdot p_2$$

Parametric Curve Construction

- Let us look at the entire curve:

$$curve(t) = \\ = b_{0,0}\, p_0 + b_{0,1}\, p_1 + b_{0,2}\, p_2 + \\ + b_{1,0}\, p_0 t + b_{1,1}\, p_1 t + b_{1,2}\, p_2 t + \\ + b_{2,0}\, p_0 t^2 + b_{2,1}\, p_1 t^2 + b_{2,2}\, p_2 t^2$$

- We can rewrite it as:

$$curve(t) = \boldsymbol{b_0}(\boldsymbol{t}) \cdot p_0 + \boldsymbol{b_1}(\boldsymbol{t}) \cdot p_1 + \boldsymbol{b_2}(\boldsymbol{t}) \cdot p_2$$

$$\boldsymbol{b_i}(\boldsymbol{t}) = b_{0,i} + b_{1,i} \cdot t + b_{2,i} \cdot t^2$$

← **Coefficients from one (*i*-th) column of the matrix B.**

Parametric Curve Construction

- Let us look at the entire curve:

$$curve(t) =$$
$$= b_{0,0}\, p_0 + b_{0,1}\, p_1 + b_{0,2}\, p_2 +$$
$$+ b_{1,0}\, p_0 t + b_{1,1}\, p_1 t + b_{1,2}\, p_2 t +$$
$$+ b_{2,0}\, p_0 t^2 + b_{2,1}\, p_1 t^2 + b_{2,2}\, p_2 t^2$$

- We can rewrite it as:

$$curve(t) = b_0(t) \cdot p_0 + b_1(t) \cdot p_1 + b_2(t) \cdot p_2$$

$$b_i(t) = b_{0,i} + b_{1,i} \cdot t + b_{2,i} \cdot t^2$$

The functions b_i are called **basis / blending functions**.

Parametric Curve Construction

- We have constructed a quadratic equation of time to interpolate our control points!

Parametric Curve Construction

- We have constructed a quadratic equation of time to interpolate our control points!
- Similar construction can be done for cubic equations and different other constraints (besides interpolation).

Parametric Curve Construction

- We have constructed a quadratic equation of time to interpolate our control points!
- Similar construction can be done for cubic equations and different other constraints (besides interpolation).

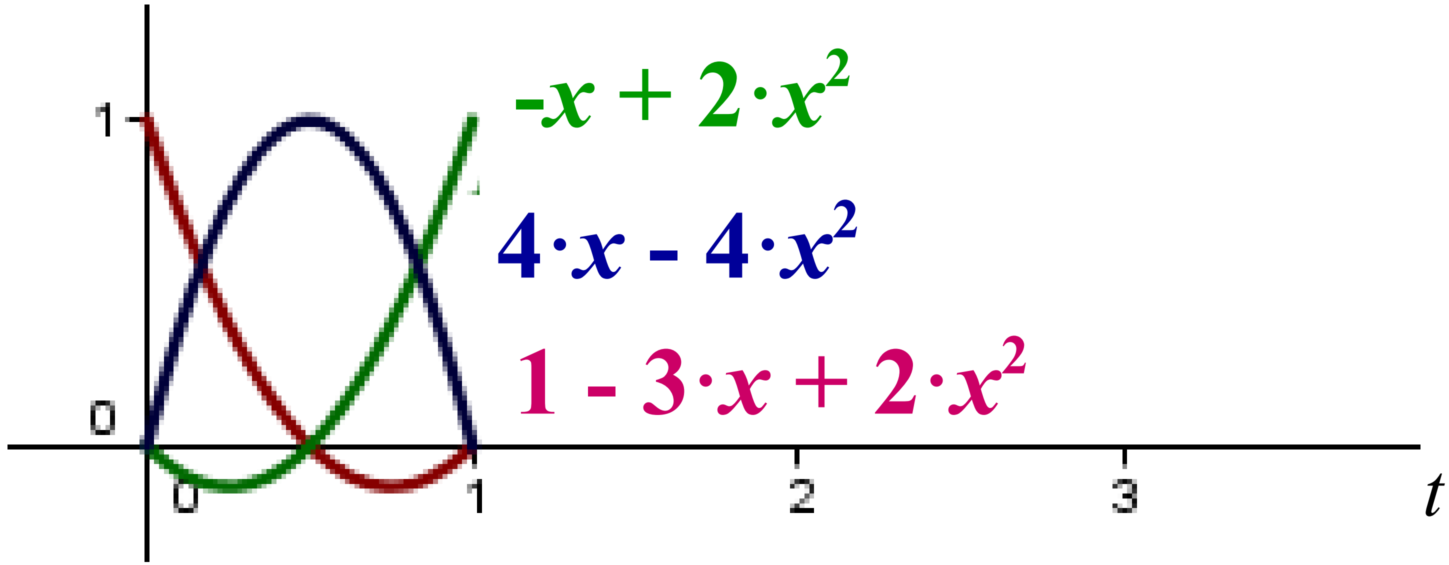
1) Pick a degree of the curve
2) Fix the parameters (*incl* control points)
3) Create the constraint matrix C
4) Find the basis matrix $B = C^{-1}$
5) Read the blending functions from the basis matrix

Blending Functions

- Used to interpolate between the parameters.

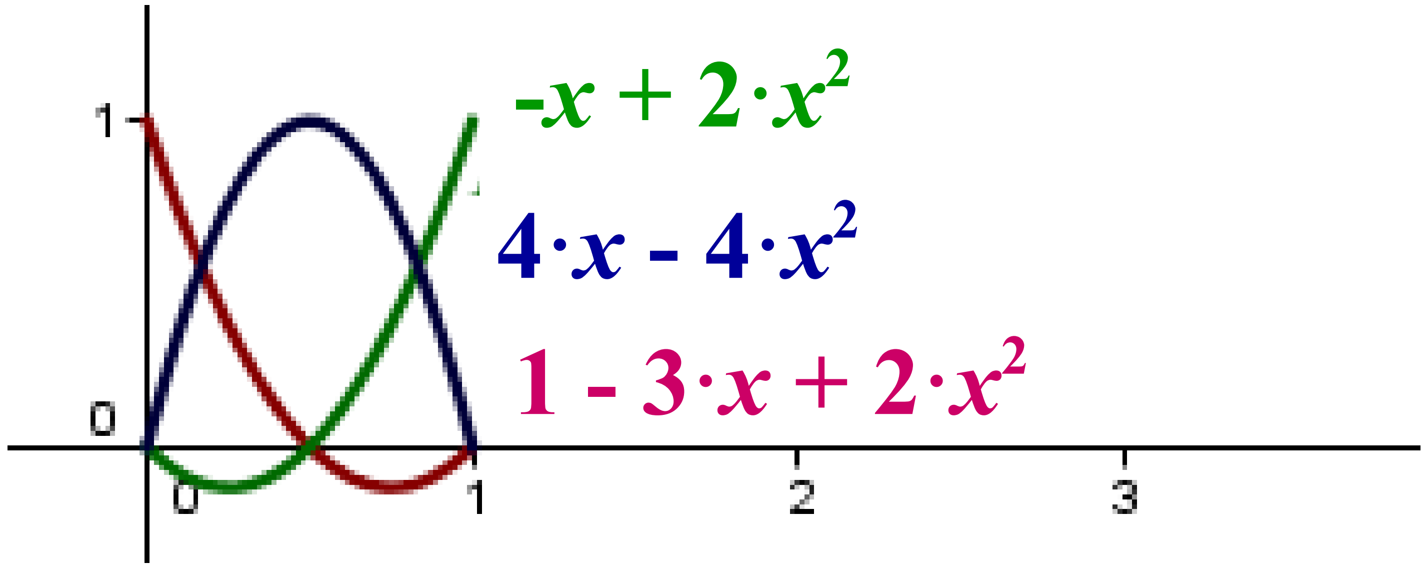
Blending Functions

- Used to interpolate between the parameters.
- Here are the found blending functions for interpolating a quadratic curve between 3 points:



Blending Functions

- Used to interpolate between the parameters.
- Here are the found blending functions for interpolating a quadratic curve between 3 points:



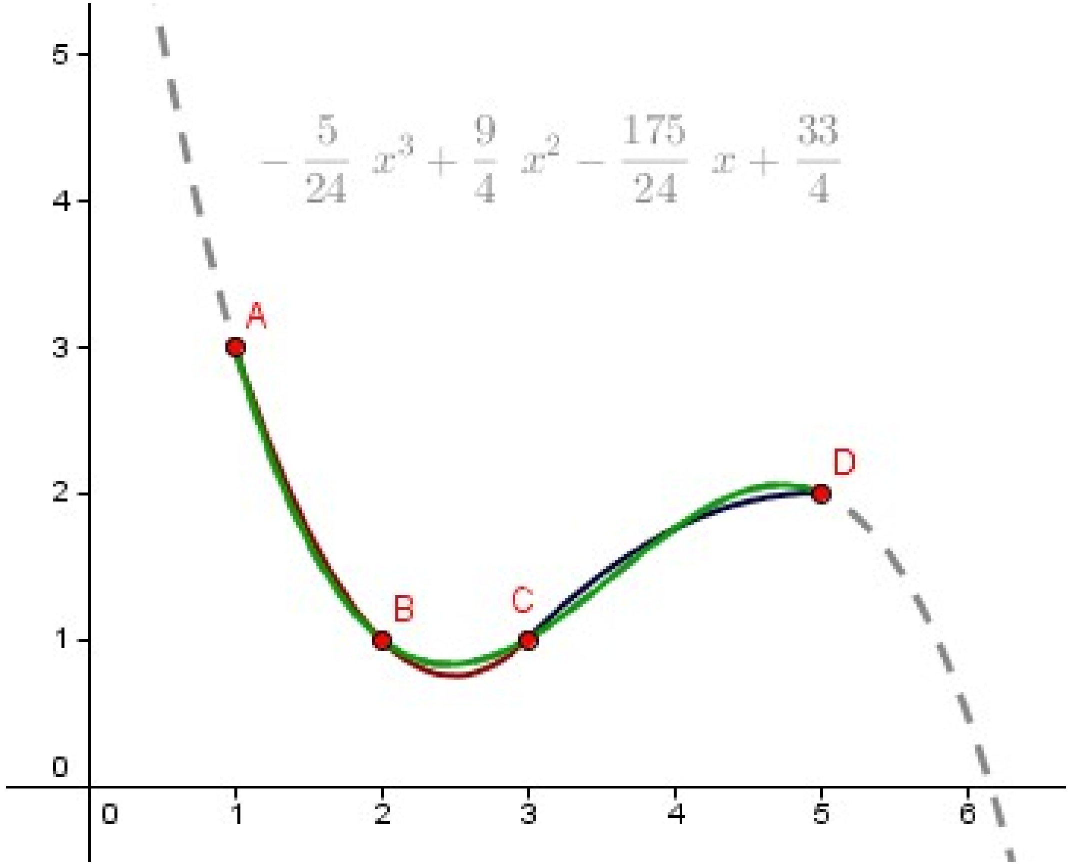
- Different constraints give different functions

Cubic not Quadratic

- In computer graphics, we usually want to use cubic polynomials, not quadratics.

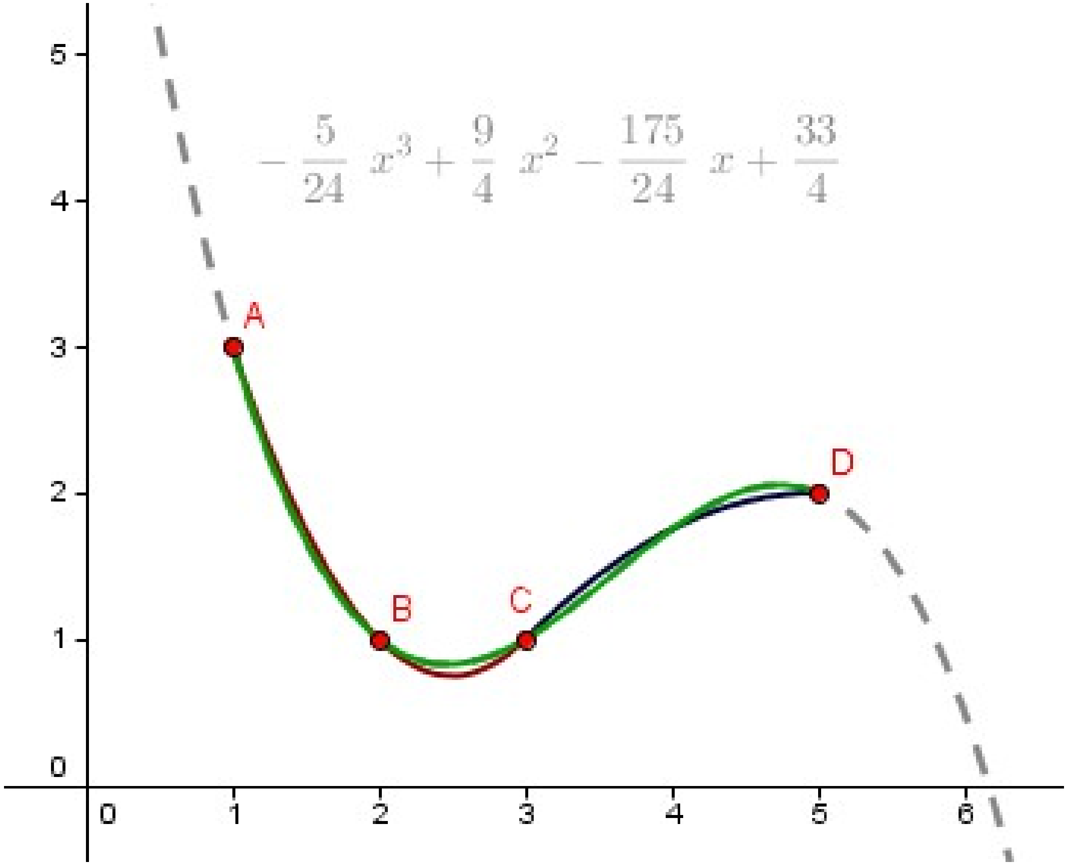
Cubic not Quadratic

- In computer graphics, we usually want to use cubic polynomials, not quadratics.
- Cubic polynomials provide us with 4 possible constraints.



Cubic not Quadratic

- In computer graphics, we usually want to use cubic polynomials, not quadratics.
- Cubic polynomials provide us with 4 possible constraints.
- Splines can achieve C^2 smoothness.



Hermite Spline

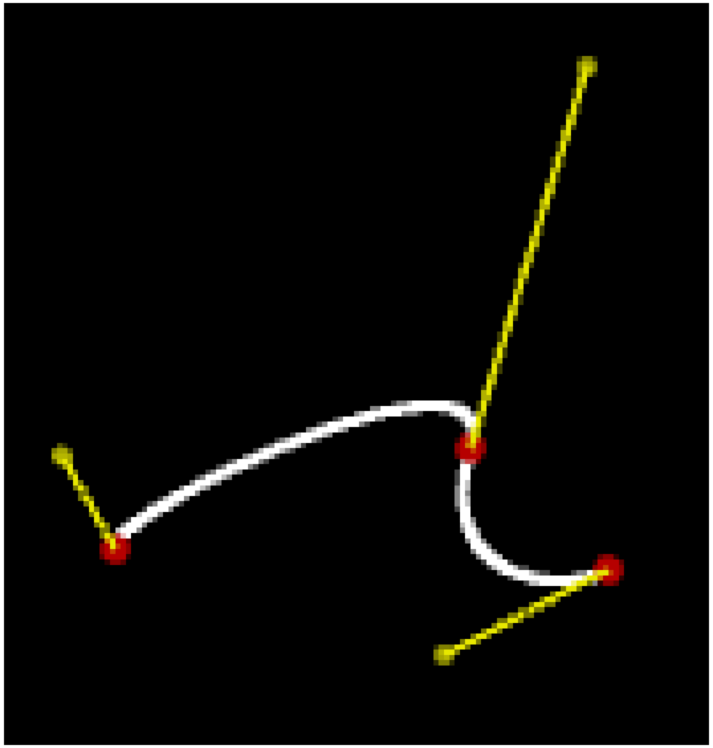
- The derivatives at the endpoints are parameters.
- Segments share the endpoints and derivatives.

$$curve(0) = p_0$$

$$curve'(0) = p_1$$

$$curve(1) = p_2$$

$$curve'(1) = p_3$$



Hermite Spline

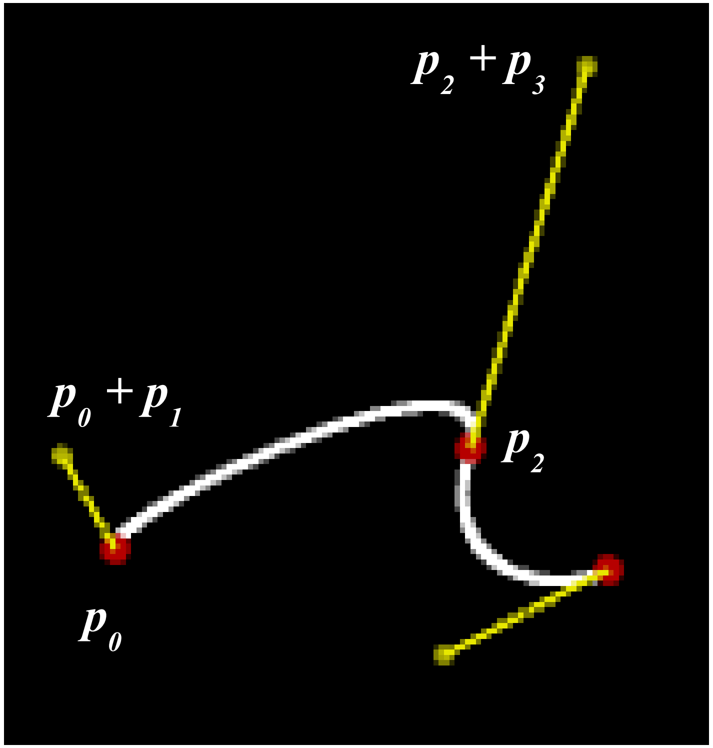
- The derivatives at the endpoints are parameters.
- Segments share the endpoints and derivatives.

$$curve(0) = p_0$$

$$curve'(0) = p_1$$

$$curve(1) = p_2$$

$$curve'(1) = p_3$$



Catmull-Rom Spline

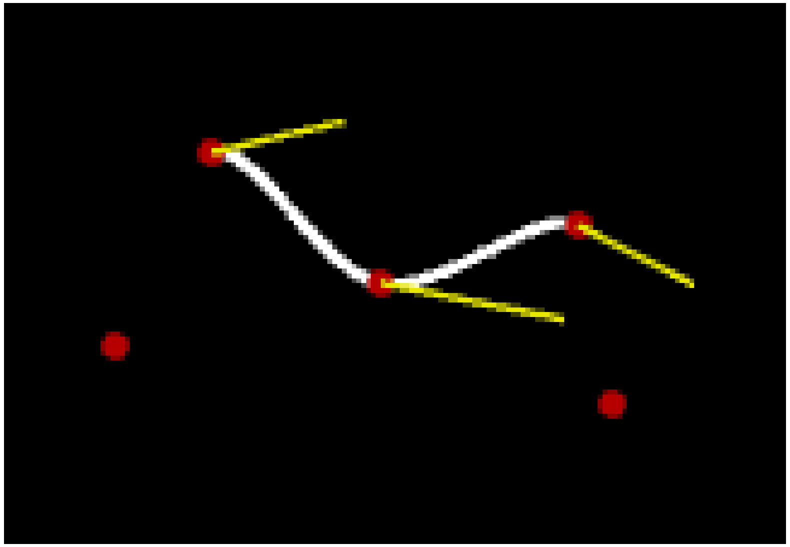
- We interpolate the p_1 and p_2.
- Dervatives are calculated using the other points.

$$curve'(0) = 0.5 \cdot (p_2 - p_0)$$

$$curve(0) = p_1$$

$$curve(1) = p_2$$

$$curve'(1) = 0.5 \cdot (p_3 - p_1)$$



- Only specify start and end derivatives, others are calculated.

Catmull-Rom Spline

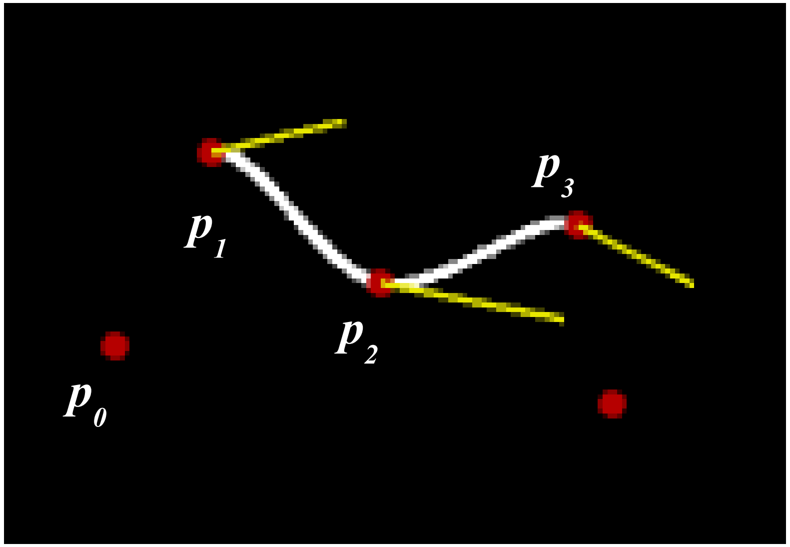
- We interpolate the p_1 and p_2.
- Dervatives are calculated using the other points.

$$curve'(0) = 0.5 \cdot (p_2 - p_0)$$

$$curve(0) = p_1$$

$$curve(1) = p_2$$

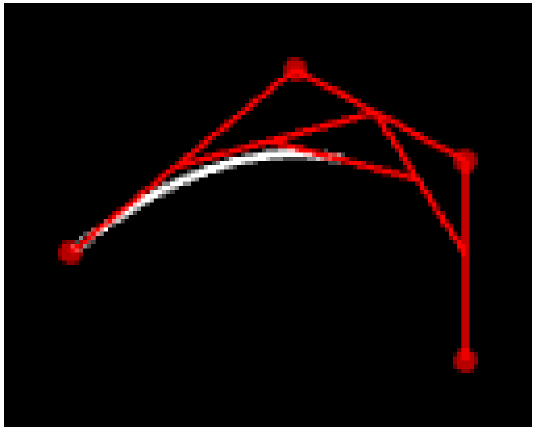
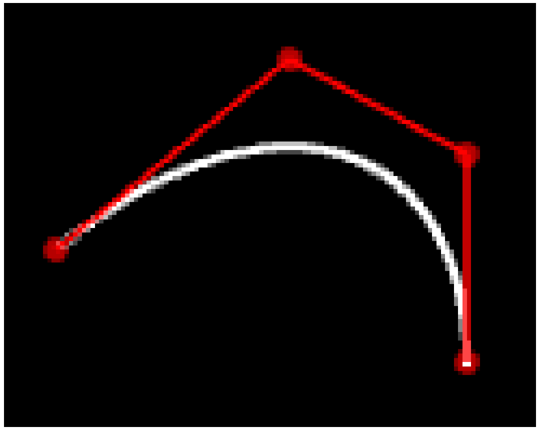
$$curve'(1) = 0.5 \cdot (p_3 - p_1)$$



- Only specify start and end derivatives, others are calculated.

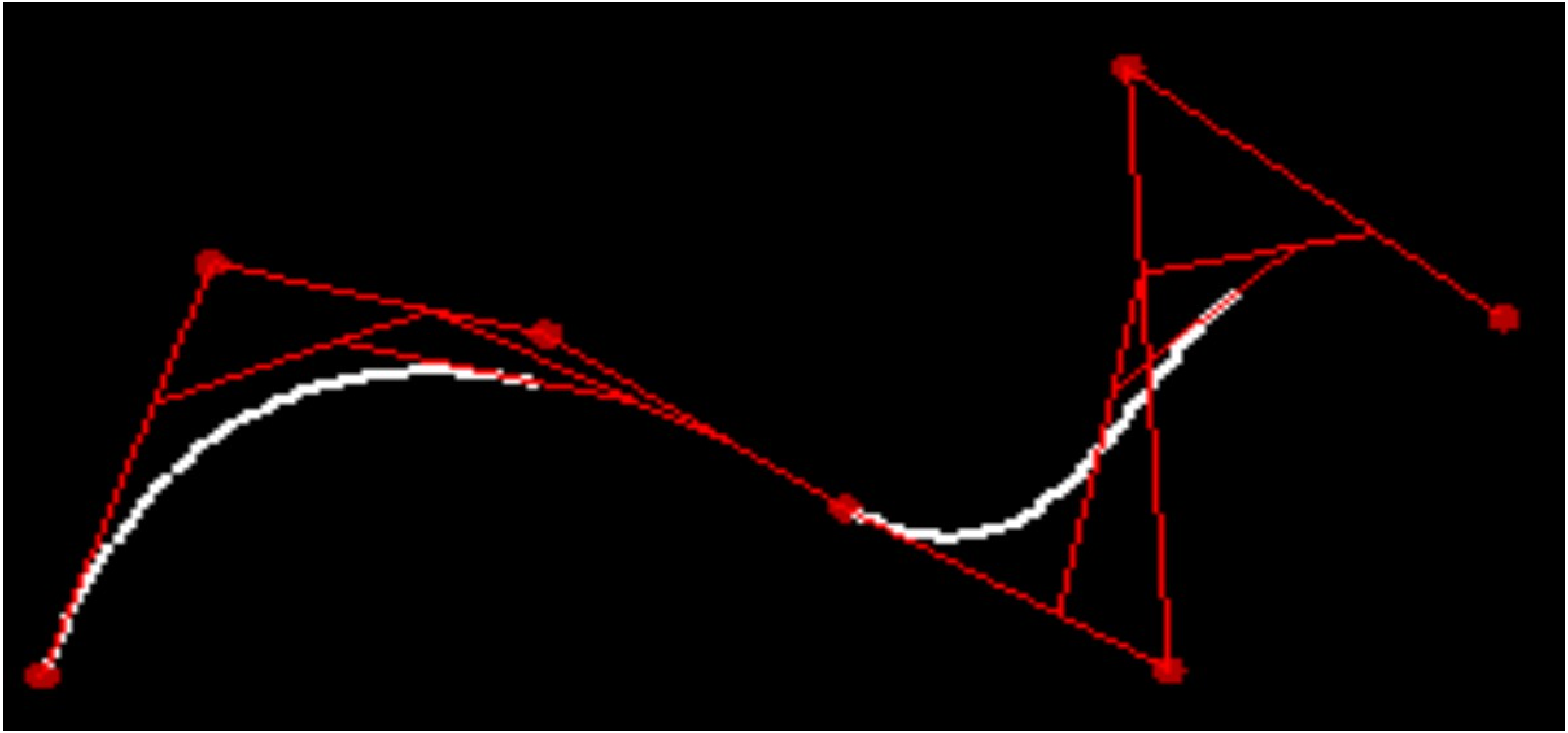
Bezier Curve

- Could be constructed using the constraints and finding the blending functions.
- Could also be constructed in a procedural way:
  - Subdivide the lines connecting the control points, into proportions t and $(1-t)$.
  - Do it recusively until at last subdivision, which will give a point on the curve.

Bezier Curve

- That procedure is called De Casteljau's algorithm.



Bezier Curve

- That procedure is called De Casteljau's algorithm.
- The corresponding blending functions are called Bernstein basis polynomials.

$$b_{0,0}(t) = 1$$

$$b_{0,1}(t) = 1 - t\,, \quad b_{1,1}(t) = t$$

$$b_{0,2}(t) = (1-t)^2\,, \quad b_{1,2}(t) = 2 \cdot t \cdot (1-t)\,, \quad b_{2,2}(t) = t^2$$

$$b_{i,\,degree}(t) = \binom{degree}{i} \cdot t^i \cdot (1-t)^{degree-i}$$

Bezier Curve

- That procedure is called De Casteljau's algorithm.
- The corresponding blending functions are called Bernstein basis polynomials.

Already familiar?

$$b_{0,0}(t) = 1$$

$$b_{0,1}(t) = 1 - t, \quad b_{1,1}(t) = t$$

$$b_{0,2}(t) = (1-t)^2, \quad b_{1,2}(t) = 2 \cdot t \cdot (1-t), \quad b_{2,2}(t) = t^2$$

$$b_{i,degree}(t) = \binom{degree}{i} \cdot t^i \cdot (1-t)^{degree-i}$$

Bezier Curve

- That procedure is called De Casteljau's algorithm.
- The corresponding blending functions are called Bernstein basis polynomials.

Those you already used in the practice session.

$$b_{0,0}(t)=1$$

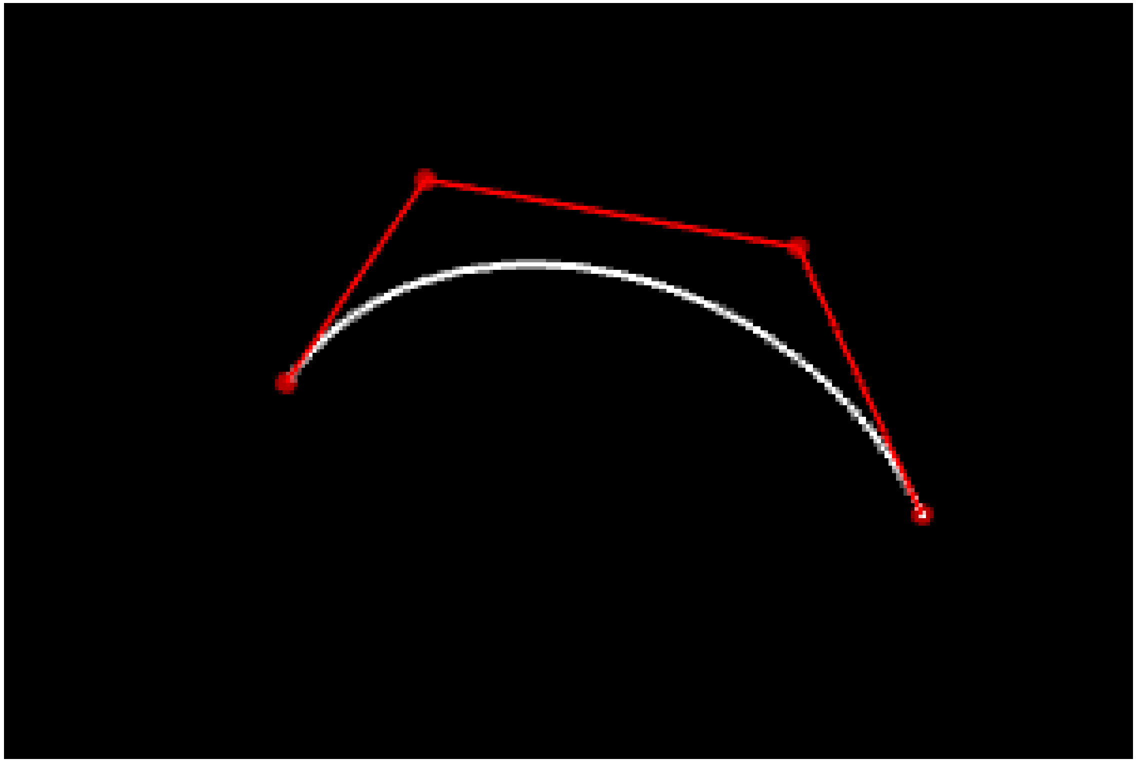
$$b_{0,1}(t)=1-t\,,\quad b_{1,1}(t)=t$$

$$b_{0,2}(t)=(1-t)^2,\quad b_{1,2}(t)=2\cdot t\cdot(1-t),\quad b_{2,2}(t)=t^2$$

$$b_{i,degree}(t)=\binom{degree}{i}\cdot t^i\cdot(1-t)^{degree-i}$$

Bezier Curve

- Always inside the convex hull of the control points.



Bezier Curve

- Always inside the convex hull of the control points.
- Affine invariance – affine transformations on the control points, transform the curve itself correctly too.

Bezier Curve

- Always inside the convex hull of the control points.
- Affine invariance – affine transformations on the control points, transform the curve itself correctly too.
- Sufficiently smooth splines can be constructed (Stärk's construction, we will see in the practice)

Bezier Curve

- Always inside the convex hull of the control points.
- Affine invariance – affine transformations on the control points, transform the curve itself correctly too.
- Sufficiently smooth splines can be constructed (Stärk's construction, we will see in the practice)
- Very widely used (*eg* font rendering)

Cubic Splines

- When constructing cubic splines, only 2 of the following properties can be satisfied at once:

Cubic Splines

- When constructing cubic splines, only 2 of the following properties can be satisfied at once:

  a) Spline is C^2 smooth.

Cubic Splines

- When constructing cubic splines, only 2 of the following properties can be satisfied at once:
  a) Spline is C^2 smooth.
  b) Spline interpolates the control points.

Cubic Splines

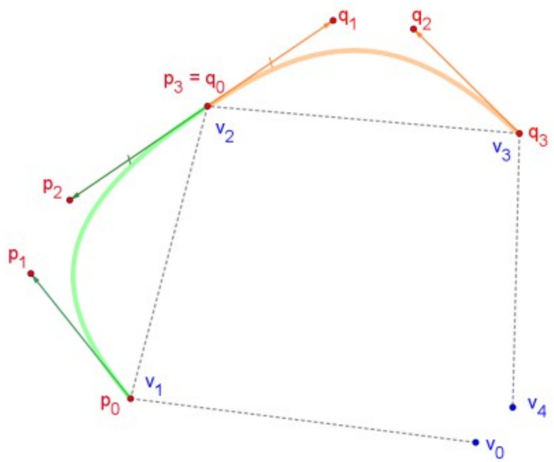
- When constructing cubic splines, only 2 of the following properties can be satisfied at once:
  a) Spline is C^2 smooth.
  b) Spline interpolates the control points.
  c) Spline has local control (changes in control points do not generally affect the entire curve)..

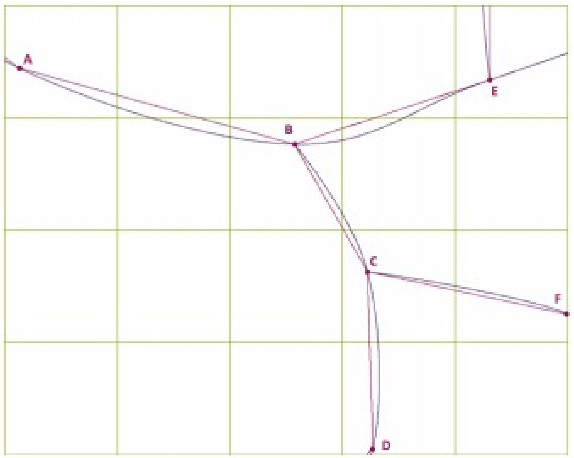
Cubic Splines

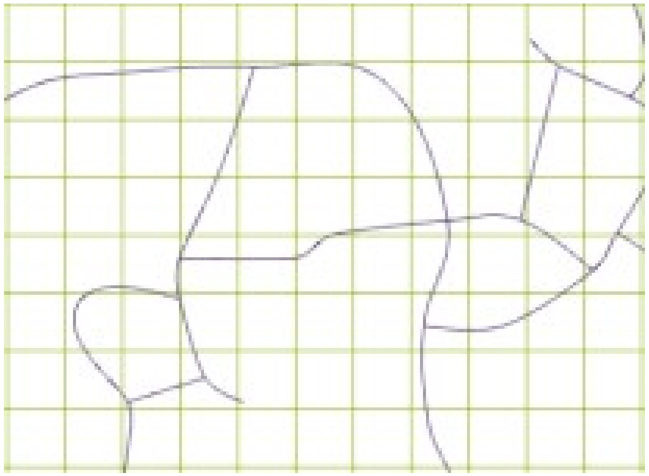
- When constructing cubic splines, only 2 of the following properties can be satisfied at once:
  a) Spline is C^2 smooth.
  b) Spline interpolates the control points.
  c) Spline has local control (changes in control points do not generally affect the entire curve).

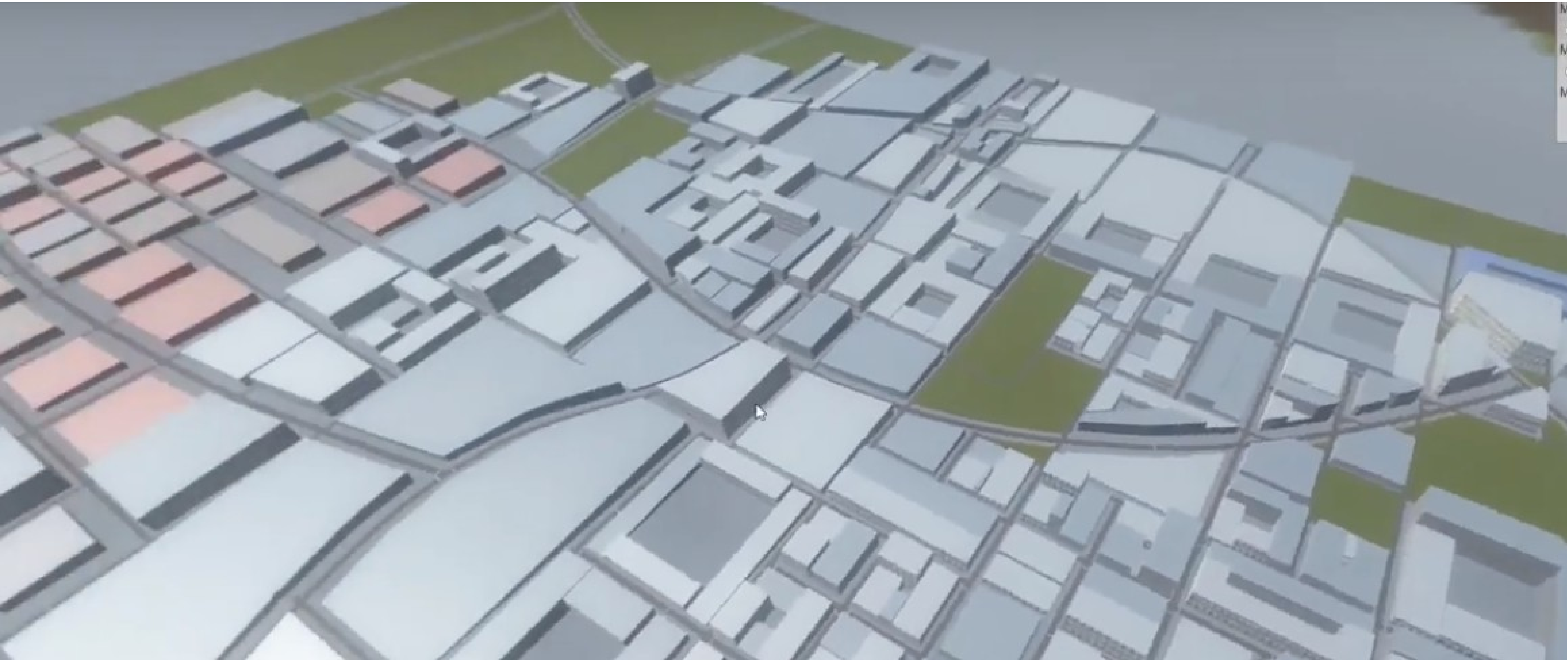
- Hermite and Catmull-Rom – are not C^2 smooth.
- Bezier – does not interpolate the control points.

Infinite Procedural Infrastuctured World Generation by Andreas Sepp









What did you find exciting today?

What more would you like to know?

Next time

Procedural Generation – *Jaanus Jaggo*