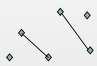
## Introduction to Curves

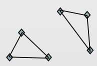
## Modelling

- Points
  - Defined by 2D or 3D coordinates

- Lines
  - Defined by a set of 2 points

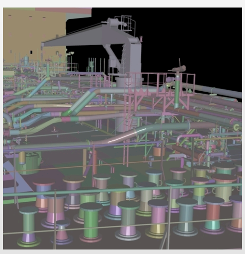


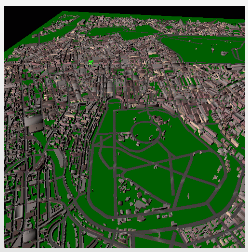
- Polygons
  - Defined by a sequence of lines
  - Defined by a list of ordered points



## 3D Models

Triangular mesh

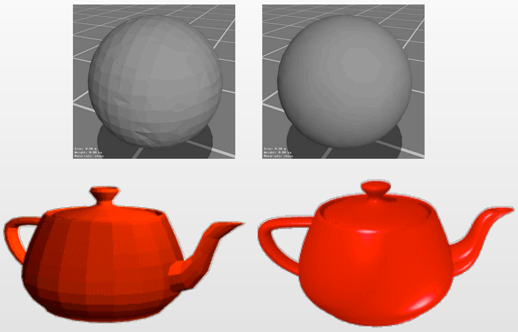




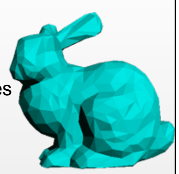
## Limitations of Polygonal Meshes

- Planar facets (& silhouettes)
- Fixed resolution
- Deformation is difficult
- No natural parameterization (for texture mapping)

## Need to Disguise the Facets



## Continuity Definitions



- $C^0$ continuous
  - curve/surface has no breaks/gaps/holes
- $G^1$ continuous
  - tangent at joint has same direction
- $C^1$ continuous
  - curve/surface derivative is continuous
  - tangent at join has same direction *and* magnitude
- $C^n$ continuous
  - curve/surface through $n^{th}$ derivative is continuous
  - important for shading

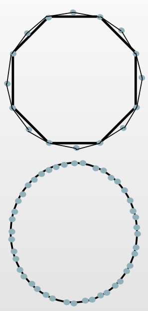
## Let's start with curves

## What if you want to have curves?

- Curves are often described with an analytic equation
- It's different from the discrete description of polygons
- How do you deal with it in Computer Graphics?

## First Solution

- Refine the number of points
  - Can become extremely complex!
  - How do we interpolate?
- Draw freehand ---
  - Too much data!

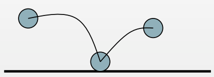


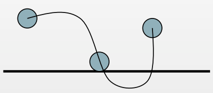
## And for more complex curves?

Can I approximate this with line segments?

## Interpolation

- How to interpolate between points?





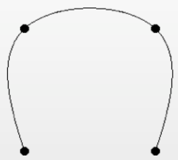
- Which one corresponds to what we want?

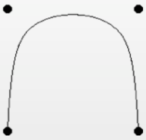
## Using curves for 3D modelling

- Modelled with curved surfaces, displayed with polygons


UCL

## Interpolation vs. Approximation Curves





**Interpolation**

curve must pass through control points

**Approximation**

curve is influenced by control points

UCL

## Interpolation vs. Approximation Curves

- Interpolation curve – over constrained
  $\rightarrow$ lots of (undesirable?) oscillations

- Approximation curve – more reasonable?

UCL

## Math background

- Polynomials
  - n-th degree polynomial:
    $p(t) = a_0 + a_1\, t + a_2\, t^2 + a_3\, t^3 + \ldots + a_n\, t^n$
- Affine map
  - In one variable, defined as:
    $f(t) = a + b\, t$
  - A mapping is affine:
    $f(\alpha_0 {*} t_0 + \alpha_1 {*} t_1) = \alpha_0 {*} f(t_0) + \alpha_1 {*} f(t_1)$ if $\alpha_0 + \alpha_1 = 1$

## Interpolation

- On affine maps
  - For t in $[t_1,t_2]$

$$t = \frac{t_2 - t}{t_2 - t_1} t_1 + \frac{t - t_1}{t_2 - t_1} t_2$$

  - Hence, for any affine f():

$$f(t) = \frac{t_2 - t}{t_2 - t_1} f(t_1) + \frac{t - t_1}{t_2 - t_1} f(t_2)$$

since $\frac{t_2 - t}{t_2 - t_1} + \frac{t - t_1}{t_2 - t_1} = 1$

## Example of Affine Mapping

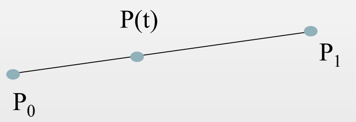
- Given $g(x) = 3x + 2$ and $\lambda_0 + \lambda_1 = 1$
- We can show:

$$\begin{aligned} g(\lambda_0 x_0 + \lambda_1 x_1) &= 3(\lambda_0 x_0 + \lambda_1 x_1) + 2 \\ &= 3\lambda_0 x_0 + 3\lambda_1 x_1 + 2(\lambda_0 + \lambda_1) \\ &= \lambda_0(3x_0 + 2) + \lambda_1(3x_1 + 2) \\ &= \lambda_0 g(x_0) + \lambda_1 g(x_1), \end{aligned}$$

## Parameterised line segment

t between [0,1]

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




We can generalise to t in the range $t_1$ to $t_2$ (prev. slide)

How to generalize to non-linear interpolation?

## Symmetric Multi-Affine Maps

- 2-parameter version defined as:
  - $f(t_1,t_2) = c_0 + c_1 t_1 + c_2 t_2 + c_3 t_1 t_2$ (symmetry: $c_1=c_2$)
- Properties
  - Affine *separately* on each of its arguments

    $f(\alpha_a * t_{1a} + \alpha_b * t_{1b}, t_2) = \alpha_a * f(t_{1a},t_2) + \alpha_b * f(t_{1b},t_2)$

    $f(t_1,\alpha_a * t_{2a} + \alpha_b * t_{2b}) = \alpha_a * f(t_1,t_{2a}) + \alpha_b * f(t_1,t_{2b})$
  - *Symmetry*: Any permutation of the arguments results in the same value

    $f(t_1,t_2) = f(t_2,t_1)$
- Can be extended to more parameters!

## Example

- Given $f(x,y) = xy + 3$
- Symmetry: $f(x,y) = xy + 3 = yx + 3 = f(y,x)$
- Affine:
$$\begin{aligned} f(ax_0 + bx_1, y) &= (ax_0 + bx_1)y + 3 \\ &= ax_0 y + bx_1 y + (a+b)3 \\ &= af(x_0,y) + bf(x_1,y), \end{aligned}$$

  provided $a + b = 1.$

## Diagonalization of Symmetric Multi-Affine Maps

- Multi-Affine Map defined
  - on hyper-cube $[0,1]^n$, for n-variables
- In 2-parameter case:
  - on square-domain $[0,1]^2$
- Diagonalization: all arguments take same value
  - $F(t) := f(t,t) = c0 + (c_1+c_2)\,t + c_3\,t^2$
- New function F(t)
  - Defined on *diagonal* of original domain

## Blossoming theorem

- Strong connection between multi-affine maps and polynomials
- Every n-argument multi-affine map has a *unique* n-th degree polynomial as its diagonal
- Every n-th degree polynomial corresponds to a unique symmetric n-argument multi-affine map, that has this polynomial as its diagonal
- The multi-affine map is called blossom (or polar form)

## Interpolation (through Diagonal)

- Recall interpolation on affine maps

$$f(t) = \frac{t_2 - t}{t_2 - t_1} f(t_1) + \frac{t - t_1}{t_2 - t_1} f(t_2)$$

- Consider t ∈ [r,s] in the 2-parameter case

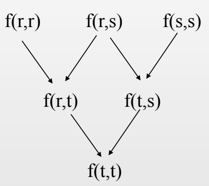
$$f(r,t) = \frac{s - t}{s - r} f(r,r) + \frac{t - r}{s - r} f(r,s)$$

$$f(t,s) = \frac{s - t}{s - r} f(r,s) + \frac{t - r}{s - r} f(s,s)$$

$$f(t,t) = \frac{s - t}{s - r} f(r,t) + \frac{t - r}{s - r} f(s,t)$$

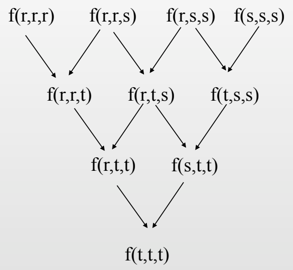
## De Casteljau triangles

- Solution for f(t,t)



## Can be extend to degree n

- Solution for f(t,t,t)




## Bézier curves

- Use it to define Bézier curves:

  $f(t_1,t_2) = (x(t_1,t_2), y(t_1,t_2))$ [= two f, one for each dim.]

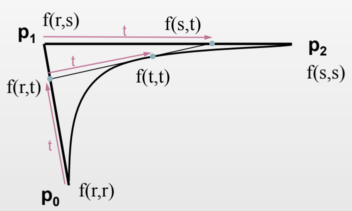
- Given 3 points:

  $p_0 = f(r,r)$ $p_1 = f(r,s)$ $p_2 = f(s,s)$

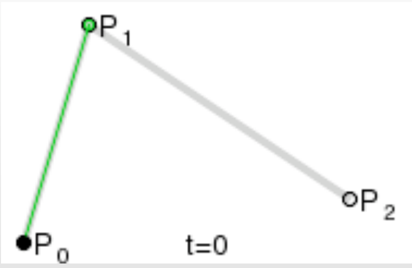
- Interpolation by f(t,t) for any value of t.
- All points given by f(t,t) will lie on a curve ($2^{nd}$ degree Bézier curve)

## Three control points (quadratic Bézier)

- $p_0$, $p_1$, $p_2$

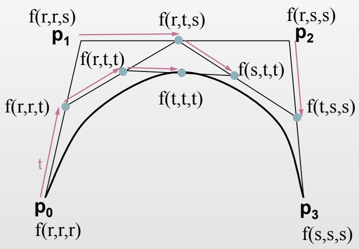


## Three control points (quadratic Bézier)

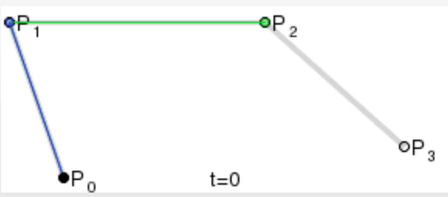



## With four control points (cubic Bézier)

- $p_0$, $p_1$, $p_2$, $p_3$




## With four control points (cubic Bézier)



Demo…

## Properties for 3rd- degree Bézier curves

- End-points $p_0$ = P(r) and $p_3$ = P(s)
- Invariance of shape when change on the parametric interval (affine transformation)
- The curve is bounded by the Convex hull given by the control points
- An affine transformation of the control points is the same as an affine transformation of any points of the curve

## Properties for 3rd- degree Bézier curves

- If the control points are on a straight line, the Bézier curve is a straight line
- The tangent vector of the curve at end points are (for t=0 and t=1)
  - $P'(r) = 3(p_1-p_0)$
  - $P'(s) = 3(p_3-p_2)$

## Polynomial form of a Bézier curve

- Restriction to interval [0,1]
- 2nd degree
  - $f(0,t) = (1-t)\ f(0,0) + t\ f(0,1) = (1-t)\ P_0 + t\ P_1$
  - $f(t,1) = (t-1)\ f(0,1) + t\ f(1,1) = (1-t)\ P_1 + t\ P_2$
  - $F(t) = f(t,t) = (1-t)\ f(0,t) + t\ f(t,1)$
    $= (1-t)^2\ P_0 + 2t(1-t)\ P_1 + t^2\ P_2$
  - Using the equation for interpolation

## Polynomial form of a Bézier curve

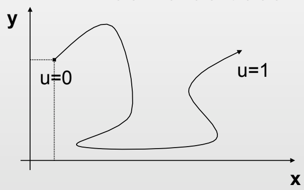
- 3rd degree
  - $f(t,t,t) = (1-t)^3\ f(0,0,0) + 3t(1-t)^2 f(0,0,1)$
    $+ 3(1-t)t^2\ f(0,1,1) + t^3\ f(1,1,1)$
    $= (1-t)^3\ P_0 + 3t(1-t)^2\ P_1 + 3(1-t)t^2\ P_2 + t^3\ P_3$

## Basis Functions

## Other way of seeing a Bézier curve

- We drive the position on a curve using a parameter *u* (for 2D or 3D):

$$\mathbf{Q}(u) = ( x(u), y(u) )$$



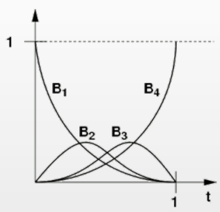

## Four control points and [0,1]

- If we restrict t between [0,1], polynomial form:
  - $f(t,t,t) = (1-t)^3 f(0,0,0) + 3t(1-t)^2 f(0,0,1) + 3(1-t)t^2 f(0,1,1) + t^3 f(1,1,1)$
    $= (1-t)^3 P_0 + 3t(1-t)^2 P_1 + 3(1-t)t^2 P_2 + t^3 P_3$
- Which can be re-written in a more general case:
  - $Q(u) = ( x(u), y(u) ) = \Sigma P_i B_i(u)$ , where

$$B_i(u) = \binom{n}{i} u^i (1-u)^{n-i} \qquad \binom{n}{i} = \frac{n!}{i!(n-i)!}$$

## Bernstein basis

- $B_0(u) = (1-u)^3$
- $B_1(u) = 3\,u\,(1-u)^2$
- $B_2(u) = 3\,u^2\,(1-u)$
- $B_3(u) = u^3$




$$\mathbf{Q}(u) = \mathbf{U}\,\mathbf{M_B}\,\mathbf{P} = [u^3\ u^2\ u\ 1] \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{p_0} \\ \mathbf{p_1} \\ \mathbf{p_2} \\ \mathbf{p_3} \end{bmatrix}$$

## Bernstein basis

- $B_i(u)$ are called Bernstein basis functions

$$B_i(u) = \binom{n}{i} u^i (1-u)^{n-i}$$

- For 3rd degree:

$$B_1(t) = (1-t)^3;\ B_2(t) = 3t(1-t)^2;\ B_3(t) = 3t^2(1-t);\ B_4(t) = t^3$$

- Property
  - Any polynomial can be expressed uniquely as a linear combination of these basis functions

## Higher-Order Bézier Curves

- > 4 control points
- Bernstein Polynomials as the basis functions

$$B_i^n(t) = \frac{n!}{i!(n-i)!} t^i (1-t)^{n-i}, \qquad 0 \le i \le n$$

- Every control point affects the entire curve
  - Not simply a local effect
  - More difficult to control for modeling

## Tangent vectors

- For t in [0,1], consider
  - $F(t) = f(t,t,t) = (1-t)^3 P_0 + 3t(1-t)^2 P_1 + 3(1-t)t^2 P_2 + t^3 P_3$
  - $F'(0) = 3 (P_1-P_0)$
  - $F'(1) = 3 (P_3-P_2)$
- In general, for a Bézier curve of degree n:
  - $F'(0) = n (P_1-P_0)$
  - $F'(1) = n (P_n-P_{n-1})$

## From polynomials, to blossoms, to control points

## Problem

- $F(t) = (X(t), Y(t)) = (1+3t^2-t^3, 1+3t-t^3)$
- Degree?
- Control points?
- $F(t) = (x(t,t,t), y(t,t,t))$
  - $x(t_1,t_2,t_3) = ?$
  - $y(t_1,t_2,t_3) = ?$
  - $P_0 = (\quad , \quad)$ $P_1 = (\quad , \quad)$
    $P_2 = (\quad , \quad)$ $P_3 = (\quad , \quad)$

## Blossoming

- Easy, when paying attention to symmetry
- For our example:
  - $x(t,t,t) = 1+3t^2-t^3$
  $\rightarrow x(t_1,t_2,t_3) = 1 + 3*(t_0t_1 + t_1t_2 + t_0t_2)/3 – t_0*t_1*t_2$
  - $y(t,t,t) = 1+3t-t^3$
  $\rightarrow y(t_1,t_2,t_3) = 1 + 3*(t_0 + t_1 + t_2)/3 - t_0*t_1*t_2$

## Deriving control points

- In general
  - $P_0$ = ( x(0,0,0), y(0,0,0) )
  - $P_1$ = ( x(0,0,1), y(0,0,1) )
  - $P_2$ = ( x(0,1,1), y(0,1,1) )
  - $P_3$ = ( x(1,1,1), y(1,1,1) )
- For our example
  - $P_0$ = (1,1)
  - $P_1$ = (1,2)
  - $P_2$ = (2,3)
  - $P_3$ = (3,3)

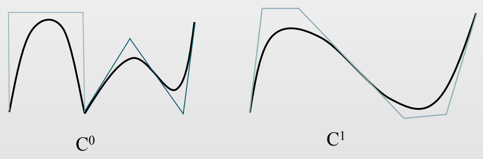
## Example

- How to derive $2^{nd}$-degree Bézier control points for parabola: $y=x^2$ ?

- Let's try it!

## Joining Bézier curves

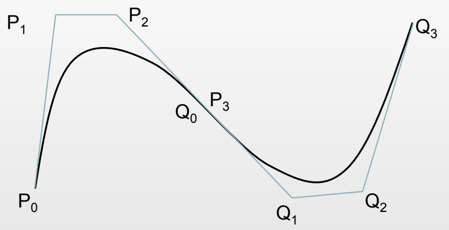
- Better to join curves than raise the number of controls points
  - Avoid numerical instability
  - Local control of the overall shape



## Joining Bézier curves

- $P_0P_1$ defines a tangent to the curve at $P_0$
- Tangent is common requirement to join two Bézier curves together (with control points $P_{0\text{-}3}$, $Q_{0\text{-}3}$)
- This requires:
  - The points $P_3$ equals $Q_0$
  - Tangents to be equal
    - I.e., $P_3$ ($=Q_0$), $P_2$, $Q_1$ are collinear
  - Called $C_1$ continuity (1st derivative is continuous)
  - $C_0$: only positions are continuous (i.e. $P_3 = Q_0$)

## Joining Bézier curves




Asymmetric: Curve goes through some control points but misses others

## Conclusions

- It is possible to define and draw a curve with a discrete representation
- All is needed are control points and interpolation strategy
- We have scene Bézier curves
  - From the DeCasteljau representation
  - From the Bernstein basis

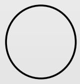
## Rational Bézier Curves

- Bézier curves cannot represent many shapes
  - E.g., no matter how high the degree a Bézier curve cannot represent a quadrant of a circle.
- Rational Bézier curves provide a more powerful tool
- The example shows how a circle can be exactly represented by the ratio of polynomials
- (Ex – find the corresponding Bézier control points for numerator and denominator!)

$$x(t) = \frac{1-t^2}{1+t^2}$$

$$y(t) = \frac{2t}{1+t^2}$$

$$t \in [0, 1]$$



## Rational Bézier Curves

- To define a rational BC we attach a 'weight' $w_i$>0 to each control point.
- Note if all the weights are equal then this is the same as a normal Bézier curve.
- The weights act as 'atractors' – the greater the weight the more the curve is pulled towards the corresponding point.

$$p(t) = \frac{\sum_{i=0}^{n} p_i w_i B_{n,i}(t)}{\sum_{i=0}^{n} w_i B_{n,i}(t)}$$

## Conclusions

- Rational Bézier Curves more powerful!