

TIEA311

Tietokonegrafiikan perusteet

kevät 2019

(“Principles of Computer Graphics” – Spring 2019)

Copyright and Fair Use Notice:

The lecture videos of this course are made available for registered students only. Please, do not redistribute them for other purposes. Use of auxiliary copyrighted material (academic papers, industrial standards, web pages, videos, and other materials) as a part of this lecture is intended to happen under academic “fair use” to illustrate key points of the subject matter. The lecturer may be contacted for take-down requests or other copyright concerns (email: paavo.j.nieminen@jyu.fi).

TIEA311 Tietokonegrafiikan perusteet – kevät 2019 ("Principles of Computer Graphics" – Spring 2019)

Adapted from: *Wojciech Matusik*, and *Frédo Durand*: 6.837 Computer Graphics. Fall 2012. Massachusetts Institute of Technology: MIT OpenCourseWare, <https://ocw.mit.edu/>.

License: Creative Commons BY-NC-SA

Original license terms apply. Re-arrangement and new content copyright 2017-2019 by *Paavo Nieminen* and *Jarno Kansanaho*

Frontpage of the local course version, held during Spring 2019 at the Faculty of Information technology, University of Jyväskylä:

<http://users.jyu.fi/~nieminen/tgp19/>

TIEA311 - Today in Jyväskylä

Plan for today:

- ▶ Usual warm-up.
- ▶ Continue from yesterday
- ▶ Go through theory
- ▶ Remember to have a break!
- ▶ The teacher will try to remember and make use of the fact that we have groups of 2-3 students with pen and paper.

TIEA311 - Today in Jyväskylä

Learning by doing that which is to be learned!

Pick up your pen and paper.

The teacher will give you a live assignment. Something quite simple.

1 minute solo; 1 minute **cross-check with group**.

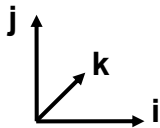
Then we **make sure everyone has it correct**.

Then we make some connections to theory.

Usual Vector Spaces

- In 3D, each vector has three components x, y, z
- But geometrically, each vector is actually the sum

$$v = x \vec{i} + y \vec{j} + z \vec{k}$$



- $\mathbf{i}, \mathbf{j}, \mathbf{k}$ are basis vectors
- Vector addition: just add components
- Scalar multiplication: just multiply components

Polynomials as a Vector Space

- Polynomials $y(t) = a_0 + a_1t + a_2t^2 + \dots + a_nt^n$
- Can be added: just add the coefficients

$$(y + z)(t) = (a_0 + b_0) + (a_1 + b_1)t + (a_2 + b_2)t^2 + \dots + (a_n + b_n)t^n$$

- Can be multiplied by a scalar: multiply the coefficients

$$s \cdot y(t) = (s \cdot a_0) + (s \cdot a_1)t + (s \cdot a_2)t^2 + \dots + (s \cdot a_n)t^n$$

Polynomials as a Vector Space

- Polynomials $y(t) = a_0 + a_1t + a_2t^2 + \dots + a_nt^n$

- In the polynomial vector space, $\{1, t, \dots, t^n\}$ are the basis vectors, a_0, a_1, \dots, a_n are the components

Subset of Polynomials: Cubic

$$y(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3$$

- Closed under addition & scalar multiplication
 - Means the result is still a cubic polynomial (verify!)
- Cubic polynomials also compose a vector space
 - A 4D **subspace** of the full space of polynomials
- The x and y coordinates of cubic Bézier curves belong to this subspace as functions of t .

Basis for Cubic Polynomials

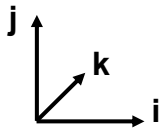
More precisely:

What's a basis?

- A set of “atomic” vectors
 - Called **basis vectors**
 - Linear combinations of basis vectors span the space
 - i.e. any cubic polynomial is a sum of those basis cubics
- Linearly independent
 - Means that no basis vector can be obtained from the others by linear combination
 - Example: $\mathbf{i}, \mathbf{j}, \mathbf{i}+\mathbf{j}$ is not a basis (missing \mathbf{k} direction!)

$$\vec{v} = x \vec{i} + y \vec{j} + z \vec{k}$$

In 3D

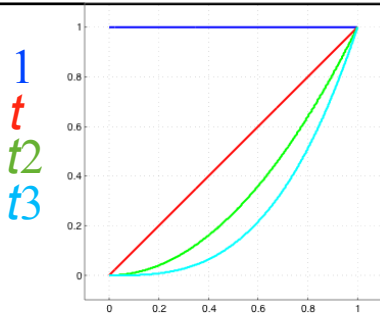


Canonical Basis for Cubics

$$\{1, t, t^2, t^3\}$$

- Any cubic polynomial is a linear combination of these:

$$a_0 + a_1 t + a_2 t^2 + a_3 t^3 = a_0 * 1 + a_1 * t + a_2 * t^2 + a_3 * t^3$$



- They are linearly independent
 - Means you cannot write any of the four monomials as a linear combination of the others. (You can try.)

TIEA311

Point of view cont'd (“independently discovered” by your teacher while thinking about this course and its primary target audience):

- ▶ Mathematics is nothing to be afraid of. Instead, we should embrace it. And, for example, in graphics, we must.
- ▶ Mathematics has been incrementally developed, peer reviewed, and **thoroughly tested** for the last 2500 years or so. (Compare with the 70 years we've had computer programming.)
- ▶ Resultingly, for many purposes, mathematics Just Works.
- ▶ We can **apply math like we apply any class library**: by learning how to read its documentation and constructing suitable instances for our current tasks!

That said, let us open the “API of mathematics”, on page “vector spaces, linear maps (\approx matrices), and their applications in cool CGI”.

6.837 Computer Graphics

Bézier Curves and Splines

Wojciech Matusik
MIT CSAIL

TIEA311 - Local plan for today

Oh, wait! Let us get one thing out of the way. . .

The teacher will now communicate to you at least three things simultaneously:

- ▶ How to pass our T2 with points 1/5, towards course grade 1/5.
- ▶ Some philosophy behind the definition of grade 1/5.
- ▶ Practically, a spoken-out definition of grade 1/5 on this course.
- ▶ One possible option of some steps that would need to be taken first in order to progress towards the higher grades, or in general “being a pro” in IT stuff.

→ Live coding and thinking aloud. See lecture video.

TIEA311

OK.

That went smoothly.

Like a smooth curve!

With score 1/5 secured, we can feel warm and happy, and stop worrying about “passing or not passing” T2 on TIEA311! That question about passing is foul in all contexts, anyway! Ugh!

This **enables us** to relax and spend some nice **focused learning time** in order to **understand more** about all this, and move towards not (just) copy-pasting, but at least copy-paste-modifying-with-some-idea-why. . .

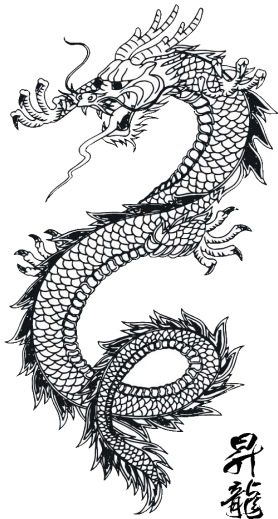
6.837 Computer Graphics

Bézier Curves and Splines

Wojciech Matusik
MIT CSAIL

Today

- Smooth curves in 2D
 - Useful in their own right
 - Provides basis for surface editing

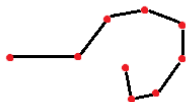


This image is in the public domain
Source: [Wikimedia Commons](#)

Modeling 1D Curves in 2D

- Polylines

- Sequence of vertices connected by straight line segments
- Useful, but not for smooth curves
- This is the representation that usually gets drawn in the end (a curve is converted into a polyline)



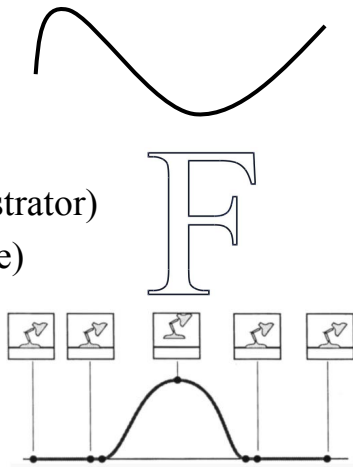
- **Smooth curves**

- How do we specify them?
- A little harder (but not too much)



Splines

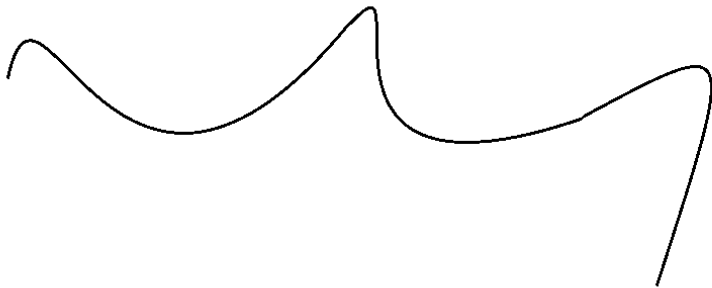
- A type of smooth curve in 2D/3D
- Many different uses
 - 2D illustration (e.g., Adobe Illustrator)
 - Fonts (e.g., PostScript, TrueType)
 - 3D modeling
 - Animation: trajectories
- In general: interpolation and approximation



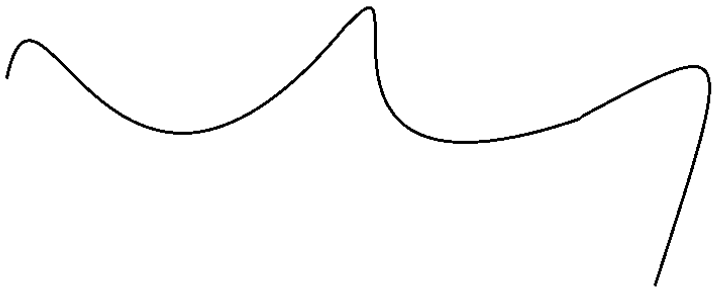
ACM © 1987 "Principles of traditional animation applied to 3D computer animation"

Demo

How Many Dimensions?

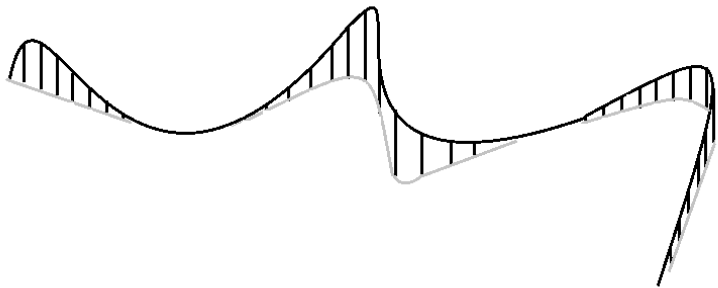


How Many Dimensions?



**This curve lies on the 2D plane,
but is itself 1D.**

How Many Dimensions?



**This curve lies on
the 2D plane,
but is itself 1D.**

**You can just as well
define 1D curves in
3D space.**

Two Definitions of a Curve

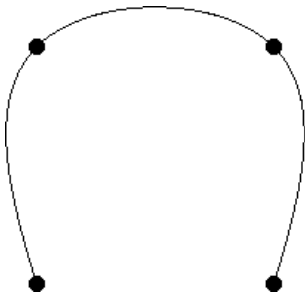
- A continuous 1D set of points in 2D (or 3D)
- A mapping from an interval S onto the plane
 - That is, $P(t)$ is the point of the curve at parameter t

$$P : \mathbb{R} \ni S \mapsto \mathbb{R}^2, \quad P(t) = \begin{pmatrix} x(t) \\ y(t) \end{pmatrix}$$

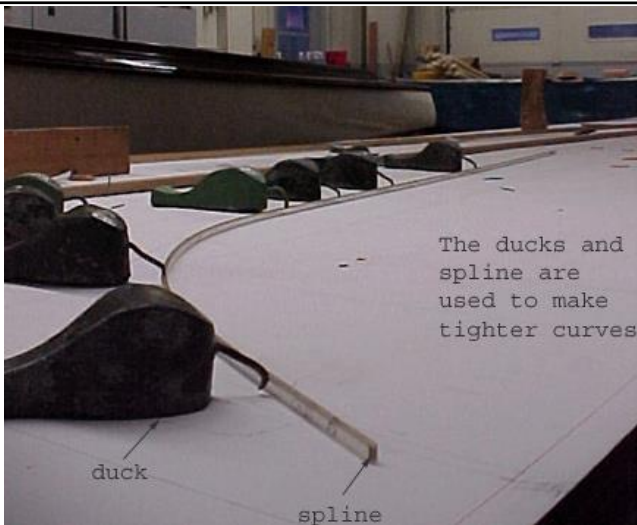
- Big differences
 - It is easy to generate points on the curve from the 2nd
 - The second definition can describe trajectories, the speed at which we move on the curve

General Principle of Splines

- User specifies **control points**
- We will interpolate the control points by a smooth curve
 - The curve is completely determined by the control points.



Physical Splines



Courtesy of The Antique Boat Museum.

See http://en.wikipedia.org/wiki/Flat_spline

Two Application Scenarios

- Approximation/interpolation
 - We have “data points”, how can we interpolate?
 - Important in many applications
- User interface/modeling
 - What is an easy way to specify a smooth curve?
 - Our main perspective today.

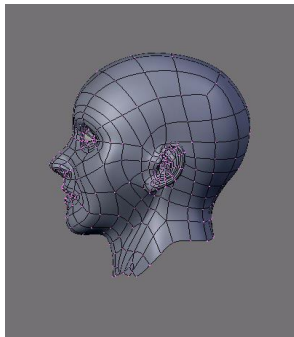


Image courtesy of [SaphireS](#) on Wikimedia Commons. License: CC-BY-SA. This content is excluded from our Creative Commons license. For more information, see <http://ocw.mit.edu/help/faq-fair-use/>.

Questions?

Splines

- Specified by a few control points
 - Good for UI
 - Good for storage

- Results in a smooth parametric curve $P(t)$
 - Just means that we specify $x(t)$ and $y(t)$
 - In practice: low-order polynomials, chained together
 - Convenient for animation, where t is time
 - Convenient for *tessellation* because we can discretize t and approximate the curve with a polyline

Tessellation

- It is easy to rasterize mathematical line segments into pixels
 - OpenGL and the graphics hardware can do it for you
- But polynomials and other parametric functions are harder

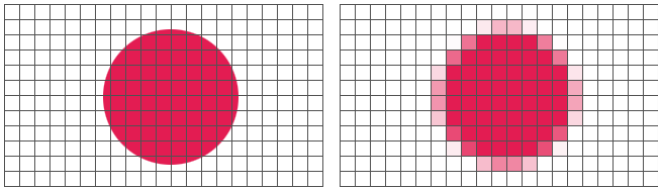
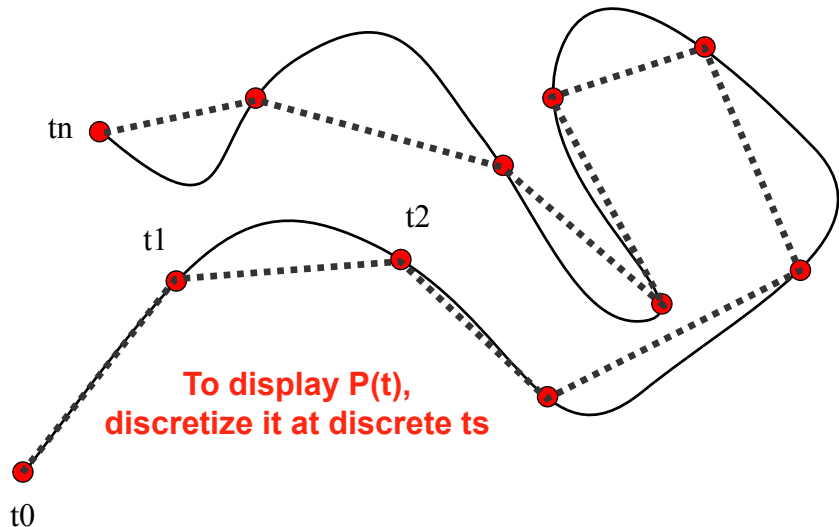
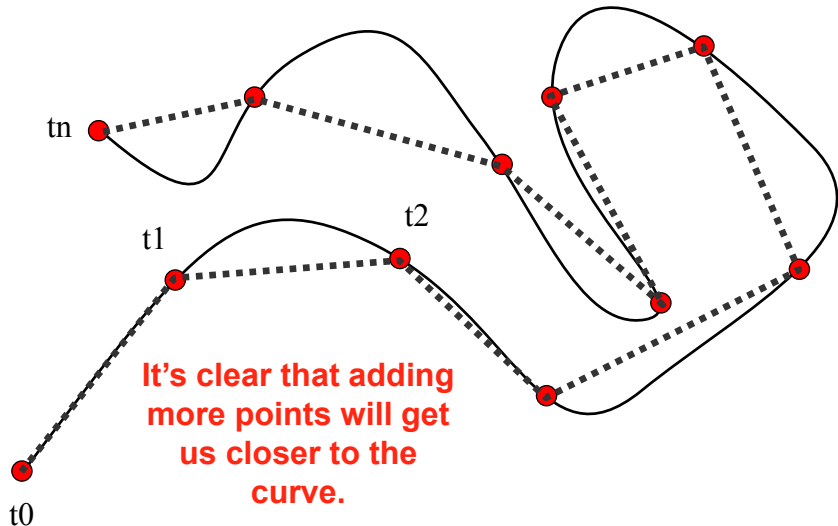


Image courtesy of [Phrood](#) on Wikimedia Commons. License: CC-BY-SA. This content is excluded from our Creative Commons license. For more information, see <http://ocw.mit.edu/help/faq-fair-use/>.

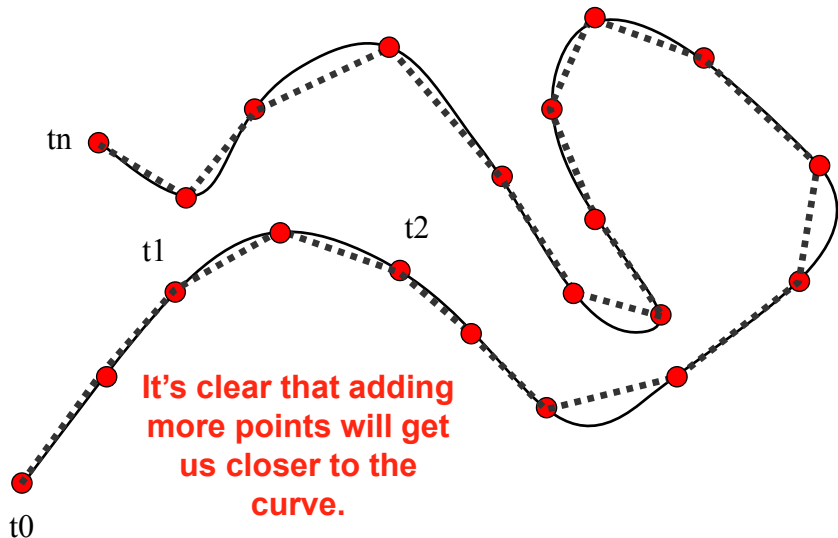
Tessellation



Tessellation

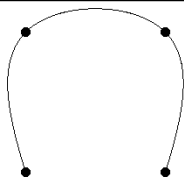


Tessellation

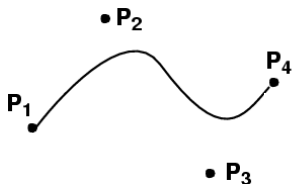


Interpolation vs. Approximation

- Interpolation
 - Goes through all specified points
 - Sounds more logical
- Approximation
 - Does not go through all points



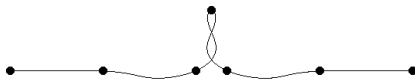
Interpolation



Approximation

Interpolation vs. Approximation

- Interpolation
 - Goes through all specified points
 - Sounds more logical
 - But can be more unstable



Interpolation

- Approximation
 - Does not go through all points
 - Turns out to be convenient



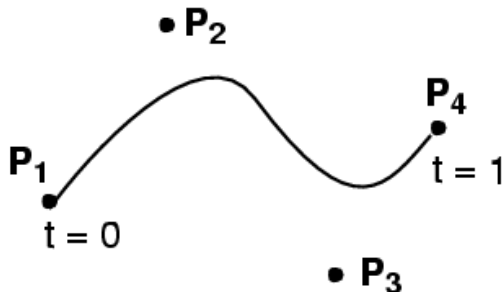
Approximation

- We will do something in between.

Questions?

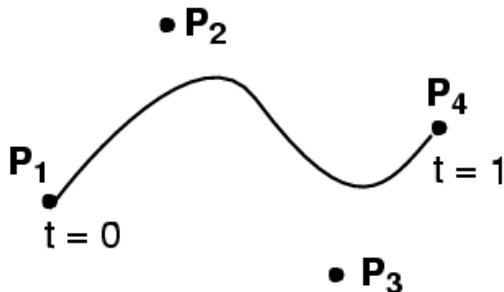
Cubic Bézier Curve

- User specifies 4 control points $P_1 \dots P_4$
- Curve goes through (interpolates) the ends P_1, P_4
- Approximates the two other ones
- Cubic polynomial



Cubic Bézier Curve

•	$P(t) = (1-t)^3$	P1
	+ $3t(1-t)^2$	P2
	+ $3t^2(1-t)$	P3
	+ t^3	P4



That is,

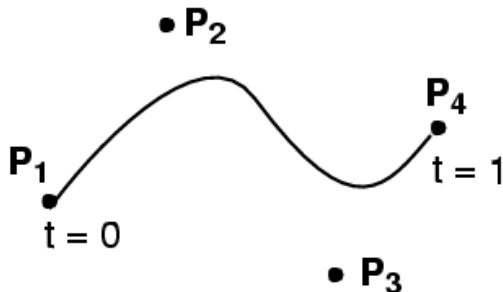
$$x(t) = (1-t)^3 x_1 + 3t(1-t)^2 x_2 + 3t^2(1-t) x_3 + t^3 x_4$$

$$y(t) = (1-t)^3 y_1 + 3t(1-t)^2 y_2 + 3t^2(1-t) y_3 + t^3 y_4$$

Cubic Bézier Curve

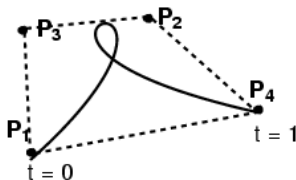
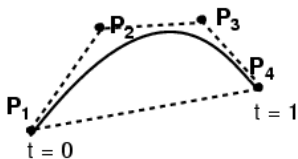
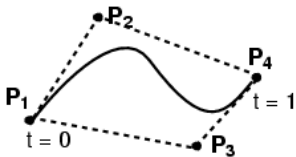
- $P(t) = (1-t)^3 \quad \mathbf{P1}$
+ $3t(1-t)^2 \quad \mathbf{P2}$
+ $3t^2(1-t) \quad \mathbf{P3}$
+ $t^3 \quad \mathbf{P4}$

Verify what happens
for $t=0$ and $t=1$



Cubic Bézier Curve

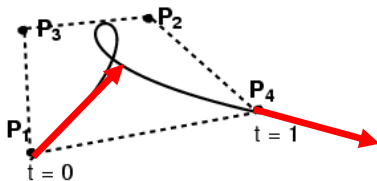
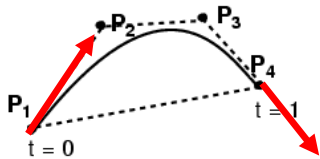
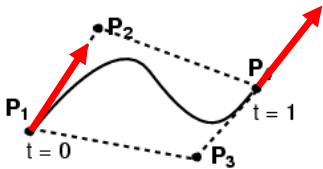
- 4 control points
- Curve passes through first & last control point



Courtesy of Seth Teller.
Used with permission.

Cubic Bézier Curve

- 4 control points
- Curve passes through first & last control point
- Curve is tangent at **P1** to **(P1-P2)** and at **P4** to **(P4-P3)**



A Bézier curve is bounded by the **convex hull** of its control points.

Questions?

Why Does the Formula Work?

- Explanation 1:
 - Magic!
- Explanation 2:
 - These are smart weights that describe the influence of each control point
- Explanation 3:
 - It is a linear combination of *basis polynomials*.

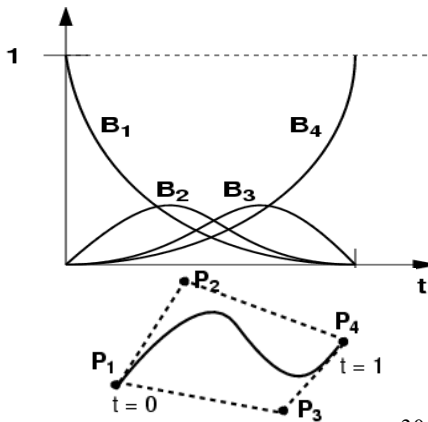
Weights

- $P(t)$ is a weighted combination of the 4 control points with weights:

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

- First, P_1 is the most influential point, then P_2 , P_3 , and P_4

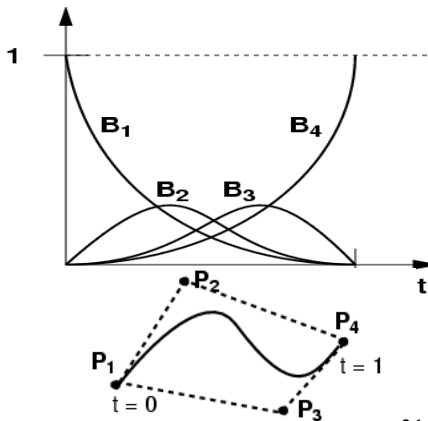
$$\begin{aligned} P(t) = & (1-t)^3 P_1 \\ & + 3t(1-t)^2 P_2 \\ & + 3t^2(1-t) P_3 \\ & + t^3 P_4 \end{aligned}$$



Weights

- P2 and P3 never have full influence
 - Not interpolated!

$$\begin{aligned} P(t) = & (1-t)^3 P_1 \\ & + 3t(1-t)^2 P_2 \\ & + 3t^2(1-t) P_3 \\ & + t^3 P_4 \end{aligned}$$



Questions?

Why Does the Formula Work?

- Explanation 1:
 - Magic!
- Explanation 2:
 - These are smart weights that describe the influence of each control point
- Explanation 3:
 - It is a linear combination of *basis polynomials*.
 - *The opposite perspective:*
control points are the weights of polynomials!!!

Why Study Splines as Vector Space?

- Understand relationships between types of splines
 - Conversion
- Express what happens when a spline curve is transformed by an affine transform (rotation, translation, etc.)
- Cool simple example of non-trivial vector space
- Important to understand for advanced methods such as finite elements

TIEA311 towards next week

- ▶ Make use of the live instruction we provide during this course!
- ▶ If you do this outside the lecture period, make use of friends who already understand this stuff!
- ▶ Start **at least looking** at Assignment 1. Our lectures have covered only part of the theory. We will cover more **next week** and some parts perhaps even later!
- ▶ **If you start coding** for Assignment 1, aim towards using the general spline formulation $GBT(t)$ with suitably constructed G , B , and $T(t)$ for curves and suitably constructed $[NBTV]$ for surfaces!
- ▶ Remember the Power Tools!
- ▶ During the weekend, sleep well and dream of vector spaces!

Assignment 1 etc: how to proceed

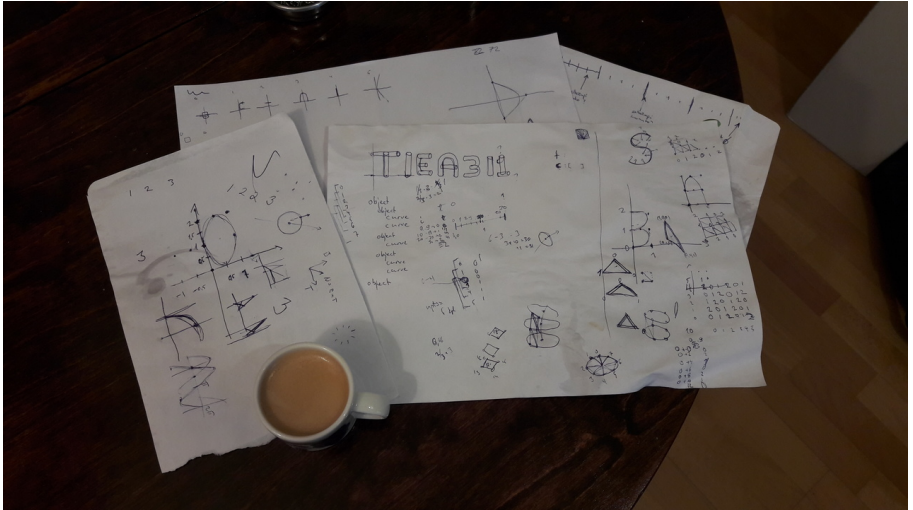
- ▶ Read instructions
- ▶ Start early
- ▶ Reflect against the theory slides
- ▶ Disregard the “start from scratch” hints! We don't have time for **that much** pain – we'll have enough, just **modifying the starter codes!**
- ▶ Ask questions when you arrive to useful ones!
- ▶ Start early
- ▶ Ask questions
- ▶ Start early
- ▶ Ask questions
- ▶ Start early
- ▶ ... [you got the point?]

Our Finnish students should revisit the material of our very first Programming course:

Vaikka ohjelmointia käytännössä tehdään suurelta osin tietokoneella, on silti kynä ja paperia syytä aina olla esillä. Ohjelmoinnin suurin vaikeus aloittelijalle onkin siinä, että ei malteta istua kynän ja paperin kanssa ja miettiä mitä ollaan tekemässä. Jos esimerkiksi pitää tehdä laivanupotuspeli, pitää ensin pelata useita kertoja peliä, jotta hahmottuu, mitä kaikkia asioita tulee aikanaan vastaan.

It is difficult for me to imagine a person who can get all the triangles and indices right in Assignment 1 without doodling stuff on paper. . . (I really want to examine your head, if you can pull that off!)

Example of pen and paper: "The making of courselogo.js"



Outcome of the adventure: <https://yousource.it.jyu.fi/tiea311-kurssimateriaalikehitys/>

[tiea311-kurssimateriaali-avoin/blobs/master/instanssi17_4k_intro_webgl/courselogo.js](https://yousource.it.jyu.fi/tiea311-kurssimateriaali-avoin/blobs/master/instanssi17_4k_intro_webgl/courselogo.js)

TIEA311 - Today in Jyväskylä

The time allotted for this week's graphics lectures is now over.

Next lecture happens in 6 days and 4 hours.

The teacher will now tell his view about what **could be useful activities** for you during that time period.

→ see lecture video.

Make notes, if you have to.

Even if he forgets to say it, **remember to rest, too!**

TIEA311

The following slides were not shown on the lecture (yet).

They are a **preview** of what we will talk about next, very soon.

Questions based on your preview will be **much appreciated** when we meet on the next lecture!

Polynomials as a Vector Space

- Polynomials $y(t) = a_0 + a_1t + a_2t^2 + \dots + a_nt^n$
- Can be added: just add the coefficients

$$(y + z)(t) = (a_0 + b_0) + (a_1 + b_1)t + (a_2 + b_2)t^2 + \dots + (a_n + b_n)t^n$$

- Can be multiplied by a scalar: multiply the coefficients

$$s \cdot y(t) = (s \cdot a_0) + (s \cdot a_1)t + (s \cdot a_2)t^2 + \dots + (s \cdot a_n)t^n$$

Subset of Polynomials: Cubic

$$y(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3$$

- Closed under addition & scalar multiplication
 - Means the result is still a cubic polynomial (verify!)
- Cubic polynomials also compose a vector space
 - A 4D **subspace** of the full space of polynomials
- The x and y coordinates of cubic Bézier curves belong to this subspace as functions of t .

Basis for Cubic Polynomials

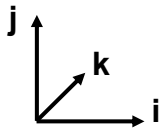
More precisely:

What's a basis?

- A set of “atomic” vectors
 - Called **basis vectors**
 - Linear combinations of basis vectors span the space
 - i.e. any cubic polynomial is a sum of those basis cubics
- Linearly independent
 - Means that no basis vector can be obtained from the others by linear combination
 - Example: $\mathbf{i}, \mathbf{j}, \mathbf{i}+\mathbf{j}$ is not a basis (missing \mathbf{k} direction!)

$$\vec{v} = x \vec{i} + y \vec{j} + z \vec{k}$$

In 3D

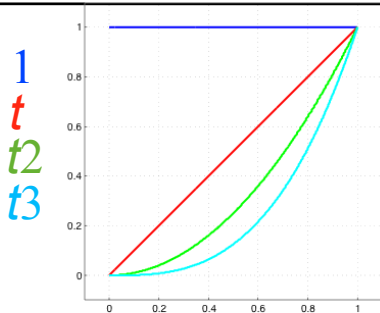


Canonical Basis for Cubics

$$\{1, t, t^2, t^3\}$$

- Any cubic polynomial is a linear combination of these:

$$a_0 + a_1 t + a_2 t^2 + a_3 t^3 = a_0 * 1 + a_1 * t + a_2 * t^2 + a_3 * t^3$$

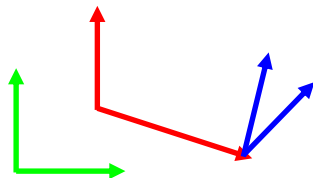


- They are linearly independent
 - Means you cannot write any of the four monomials as a linear combination of the others. (You can try.)

Different Basis

- For example:
 - $\{1, 1+t, 1+t+t^2, 1+t-t^2+t^3\}$
 - $\{t^3, t^3+t^2, t^3+t, t^3+1\}$

2D examples



- These can all be obtained from $1, t, t^2, t^3$ by linear combination
- Infinite number of possibilities, just like you have an infinite number of bases to span \mathbb{R}^2

Matrix-Vector Notation

- For example:

$1, 1+t, 1+t+t^2, 1+t-t^2+t^3$

$t^3, t^3+t^2, t^3+t, t^3+1$

Change-of-basis
matrix

“Canonical”
monomial
basis

**These
relationships
hold for each
value of t**

$$\begin{pmatrix} 1 \\ 1+t \\ 1+t+t^2 \\ 1+t-t^2+t^3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & -1 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ t \\ t^2 \\ t^3 \end{pmatrix}$$

$$\begin{pmatrix} t^3 \\ t^3+t^2 \\ t^3+t \\ t^3+1 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ t \\ t^2 \\ t^3 \end{pmatrix}$$

Matrix-Vector Notation

- For example:

1, 1+t, 1+t+t², 1+t-t²+t³

t³, t³+t², t³+t, t³+1

Change-of-basis
matrix

“Canonical”
monomial
basis

$$\begin{pmatrix} 1 \\ 1+t \\ 1+t+t^2 \\ 1+t-t^2+t^3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & -1 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ t \\ t^2 \\ t^3 \end{pmatrix}$$

Not any matrix will do!
If it's singular, the basis
set will be linearly
dependent, i.e.,
redundant and
incomplete.

$$\begin{pmatrix} t^3 \\ t^3+t^2 \\ t^3+t \\ t^3+1 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ t \\ t^2 \\ t^3 \end{pmatrix}$$

Bernstein Polynomials

- For Bézier curves, the basis polynomials/vectors are Bernstein polynomials

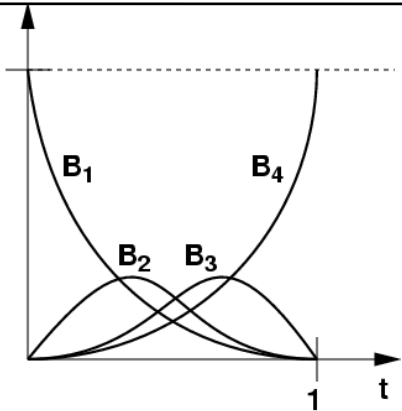
- For cubic Bezier curve:

$$B_1(t) = (1-t)^3 \quad B_2(t) = 3t(1-t)^2$$

$$B_3(t) = 3t^2(1-t) \quad B_4(t) = t^3$$

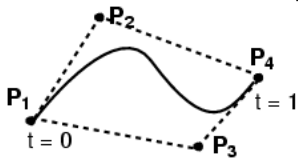
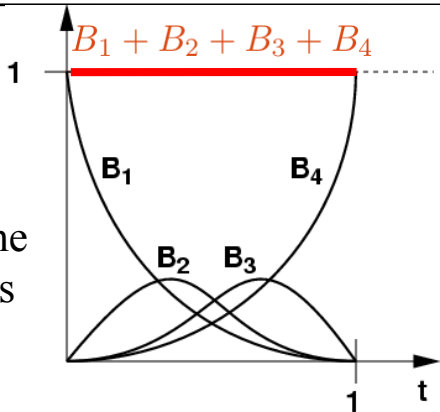
(careful with indices, many authors start at 0)

- Defined for any degree



Properties of Bernstein Polynomials

- ≥ 0 for all $0 \leq t \leq 1$
- Sum to 1 for every t
 - called *partition of unity*
- These two together are the reason why Bézier curves lie within convex hull
- $B_1(0) = 1$
 - Bezier curve interpolates P_1
- $B_4(1) = 1$
 - Bezier curve interpolates P_4



Bézier Curves in Bernstein Basis

- $P(t) = P_1B_1(t) + P_2B_2(t) + P_3B_3(t) + P_4B_4(t)$
 - P_i are 2D points (x_i, y_i)
- $P(t)$ is a linear combination of the control points with weights equal to Bernstein polynomials at t
- But at the same time, the control points (P_1, P_2, P_3, P_4) are the “coordinates” of the curve in the Bernstein basis
 - In this sense, specifying a Bézier curve with control points is exactly like specifying a 2D point with its x and y coordinates.

Two Different Vector Spaces!!!

- The plane where the curve lies, a 2D vector space
- The space of cubic polynomials, a 4D space
- Don't be confused!
- The 2D control points can be replaced by 3D points – this yields space curves.
 - The math stays the same, just add $z(t)$.
- The cubic basis can be extended to higher-order polynomials
 - Higher-dimensional vector space
 - More control points

Questions?

Change of Basis

- How do we go from Bernstein basis to the canonical monomial basis $1, t, t^2, t^3$ and back?
 - With a matrix!
- $B_1(t)=(1-t)^3$
- $B_2(t)=3t(1-t)^2$
- $B_3(t)=3t^2(1-t)$
- $B_4(t)=t^3$

$$\begin{pmatrix} B_1(t) \\ B_2(t) \\ B_3(t) \\ B_4(t) \end{pmatrix} = \begin{pmatrix} 1 & -3 & 3 & -1 \\ 0 & 3 & -6 & 3 \\ 0 & 0 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ t \\ t^2 \\ t^3 \end{pmatrix}$$


New basis vectors

How You Get the Matrix

Cubic Bernstein:

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

Expand these out
and collect powers of t .
The coefficients are the entries
in the matrix B !


$$\begin{pmatrix} B_1(t) \\ B_2(t) \\ B_3(t) \\ B_4(t) \end{pmatrix} = \overbrace{\begin{pmatrix} 1 & -3 & 3 & -1 \\ 0 & 3 & -6 & 3 \\ 0 & 0 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{pmatrix}}^B \begin{pmatrix} 1 \\ t \\ t^2 \\ t^3 \end{pmatrix}$$

Change of Basis, Other Direction

- Given $B_1 \dots B_4$, how to get back to canonical $1, t, t^2, t^3$?

$$\begin{pmatrix} B_1(t) \\ B_2(t) \\ B_3(t) \\ B_4(t) \end{pmatrix} = \overbrace{\begin{pmatrix} 1 & -3 & 3 & -1 \\ 0 & 3 & -6 & 3 \\ 0 & 0 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{pmatrix}}^B \begin{pmatrix} 1 \\ t \\ t^2 \\ t^3 \end{pmatrix}$$

Change of Basis, Other Direction

That's right, with the inverse matrix!

- Given $B_1 \dots B_4$, how to get back to canonical $1, t, t^2, t^3$?

$$\begin{pmatrix} 1 \\ t \\ t^2 \\ t^3 \end{pmatrix} = \overbrace{\begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 1/3 & 2/3 & 1 \\ 0 & 0 & 1/3 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}}^{B^{-1}} \begin{pmatrix} B_1(t) \\ B_2(t) \\ B_3(t) \\ B_4(t) \end{pmatrix}$$

Recap

- Cubic polynomials form a 4D vector space.
- Bernstein basis is canonical for Bézier.
 - Can be seen as influence function of data points
 - Or data points are coordinates of the curve in the Bernstein basis
- We can change between basis with matrices.

Questions?

More Matrix-Vector Notation

$$P(t) = \sum_{i=1}^4 P_i B_i(t) = \sum_{i=1}^4 \left[\begin{pmatrix} x_i \\ y_i \end{pmatrix} B_i(t) \right]$$

Bernstein polynomials
(4x1 vector)

$$P(t) = \begin{pmatrix} x(t) \\ y(t) \end{pmatrix} = \begin{pmatrix} x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \end{pmatrix} \begin{pmatrix} B_1(t) \\ B_2(t) \\ B_3(t) \\ B_4(t) \end{pmatrix}$$

point on curve
(2x1 vector)

matrix of
control points (2 x 4)

Flashback

$$\begin{pmatrix} B_1(t) \\ B_2(t) \\ B_3(t) \\ B_4(t) \end{pmatrix} = \overbrace{\begin{pmatrix} 1 & -3 & 3 & -1 \\ 0 & 3 & -6 & 3 \\ 0 & 0 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{pmatrix}}^B \begin{pmatrix} 1 \\ t \\ t^2 \\ t^3 \end{pmatrix}$$

Cubic Bézier in Matrix Notation

point on curve

(2x1 vector)

$$P(t) = \begin{pmatrix} x(t) \\ y(t) \end{pmatrix} =$$

Canonical
monomial basis

$$\begin{pmatrix} x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \end{pmatrix} \begin{pmatrix} 1 & -3 & 3 & -1 \\ 0 & 3 & -6 & 3 \\ 0 & 0 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ t \\ t^2 \\ t^3 \end{pmatrix}$$

“Geometry matrix”
of control points P1..P4
(2 x 4)

“Spline matrix”
(Bernstein)

General Spline Formulation

$$Q(t) = \mathbf{GBT}(\mathbf{t}) = \text{Geometry } \mathbf{G} \cdot \text{Spline Basis } \mathbf{B} \cdot \text{Power Basis } \mathbf{T}(\mathbf{t})$$

- Geometry: control points coordinates assembled into a matrix $(P_1, P_2, \dots, P_{n+1})$
- Spline matrix: defines the type of spline
 - Bernstein for Bézier
- Power basis: the monomials $(1, t, \dots, t^n)$
- Advantage of general formulation
 - Compact expression
 - Easy to convert between types of splines
 - Dimensionality (plane or space) does not really matter