TIEA311
Tietokonegrafiikan perusteet
kevät 2018

(“Principles of Computer Graphics” – Spring 2018)

TIEA311 Tietokonegrafiikan perusteet – kevät 2018 (“Principles of Computer Graphics” – Spring 2018)

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

Frontpage of the local course version, held during Spring 2018 at the Faculty of Information technology, University of Jyväskylä:
`http://users.jyu.fi/~nieminen/tgp18/`

Slides are in English - OMG, why? How do I survive?

- Face it: Population of the world is 7.5 billion. 5.5 million Finns account for 0.07 percent.
- Most of science and technology is documented in English as of 2018.
- Very nice English course material available from MIT Open CourseWare with an open license!
- You just **need to survive**, now and in your future career!
- And. . . lectures and instruction happens in Finnish during the Spring 2018 course!

Spring 2018 course

What do we have?

- Korppi page
- 1800+ slides and 6 programming assignments from MIT OpenCourseware (but this was reduced to approx. 900 slides and approx. 3.5 assignments in 2017; we'll calibrate 2018 as we do it)
- Open git repository for our local course adaptation
- Website for a static export version of the local course
- WWW full of tutorials :)

What we build along the way?

- A local course version, likely to be shorter than the OCW one. (but perhaps deeper than 2017; we calibrate)
- English-Finnish translations of terminology
- Some additional material about math and programming

Spring 2018 grading and passing the course

- All exercises must be made
- We aim at 5 ECTS credit points worth of workload, i.e., 135 hours of active studying. The course will be built along the way, so we can be “adaptive”.
- Grading will be based on the level achieved in the exercises (more information later)

What else?

Let’s dive into the OCW materials and Assignment 0 . . .

(The following slides are reproduced verbatim from “Lecture 0” of the MIT course. I showed and discussed these on our Finnish lecture.)

What you will learn in 6.837

- Fundamentals of computer graphics algorithms
  - Will give a pretty good idea of how to implement lots of the things just shown
- We will concentrate on 3D, not 2D illustration or image processing
- Basics of real-time rendering and graphics hardware
- Basic OpenGL
  - Not the focus, though: Means, not the end.
- You will get C++ programming experience

What you will NOT learn in 6.837

- OpenGL and DirectX hacks
  - Most become obsolete every 18 months anyway!
  - Does not really matter either: Graphics is becoming all software again (OpenCL, Larrabee, etc.)
- Software packages
  - CAD-CAM, 3D Studio MAX, Maya
  - Photoshop and other painting tools
- Artistic skills
- Game design

How much Math?

- Lots of simple linear algebra
  - Get it right, it will help you a lot!
- Some more advanced concepts
  - Homogeneous coordinates
  - Ordinary differential equations (ODEs) and their numerical solution
  - Sampling, antialiasing (some gentle Fourier analysis)
  - Monte-Carlo integration
- Always in a concrete and visual context

Beyond computer graphics

- Many of the mathematical and algorithmic tools are useful in other engineering and scientific context
- Linear algebra
- Splines
- Differential equations
- Monte-Carlo integration
- …

Textbooks

- No textbook is required
- Recommendations
  - 3D Computer Graphics (Watt)
  - 3D Computer Graphics: A Mathematical Introduction with OpenGL (Buss)
    - **There is a free online version** available from Books24x7
  - Real-Time Rendering, 3rd ed. (Akenine-Möller, Haines, Hoffman)
  - Fundamentals of Computer Graphics, 3rd ed. (Shirley, Marschner)

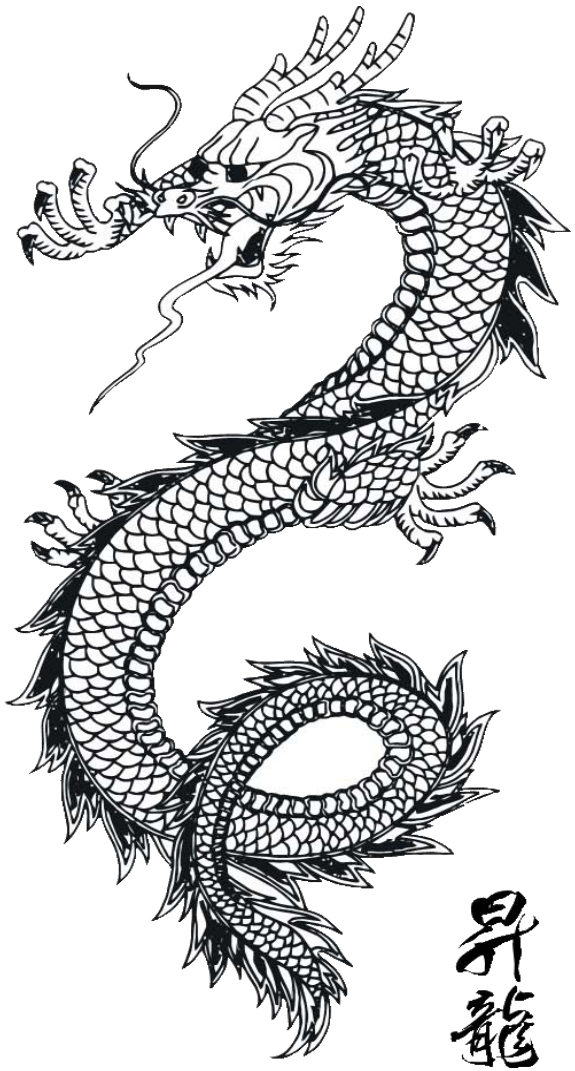
6.837 Computer Graphics

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

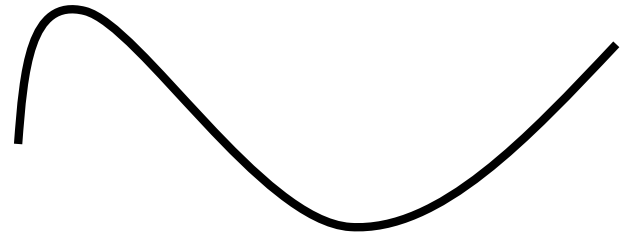
Modeling 1D Curves in 2D

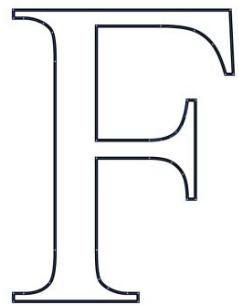
- Polylines
  - Sequence of vertices connected by straight line segments
  - Useful, but not for smooth curves
  - This is the representation that usually gets drawn in the end (a curve is converted into a polyline)

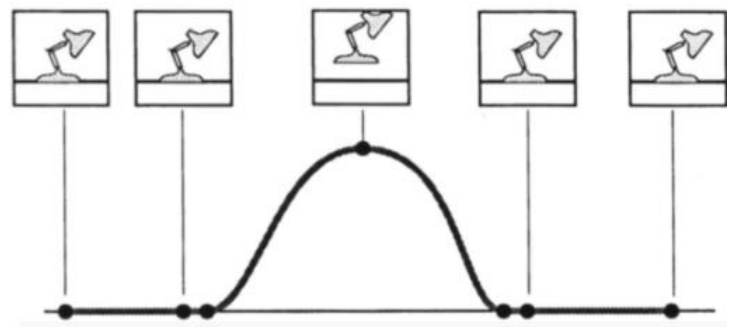
- **Smooth curves**
  - How do we specify them?
  - A little harder (but not too much)

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

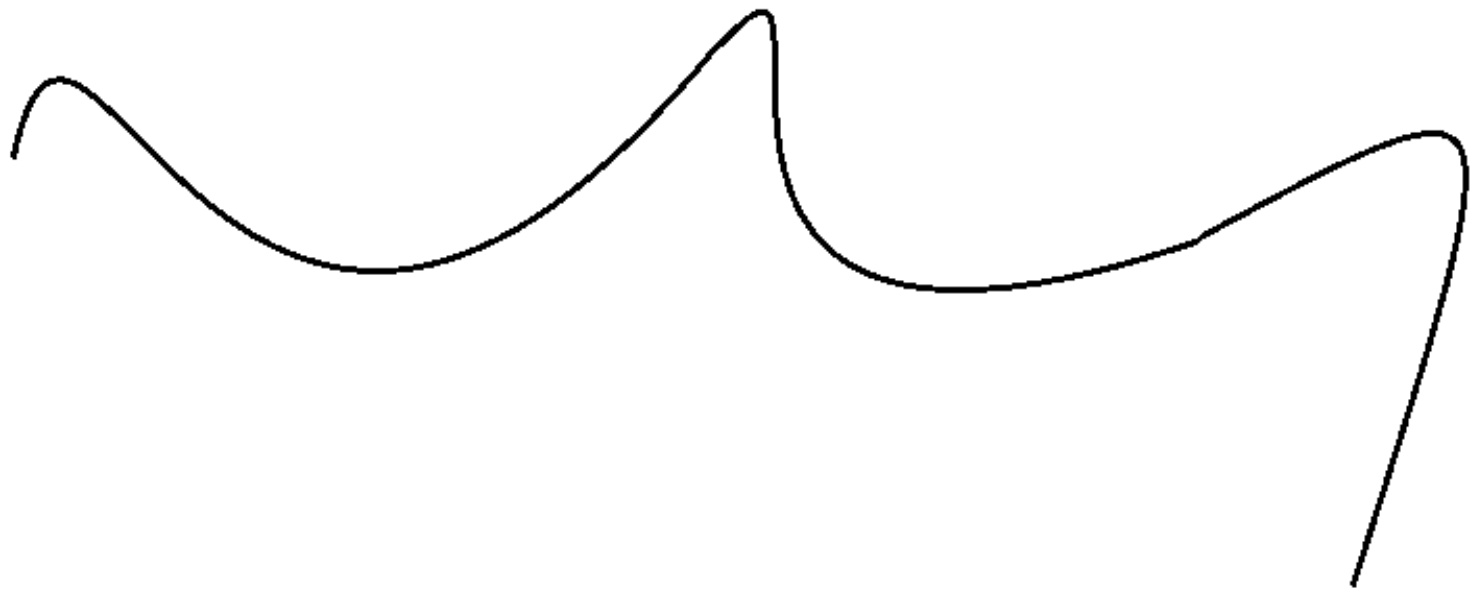




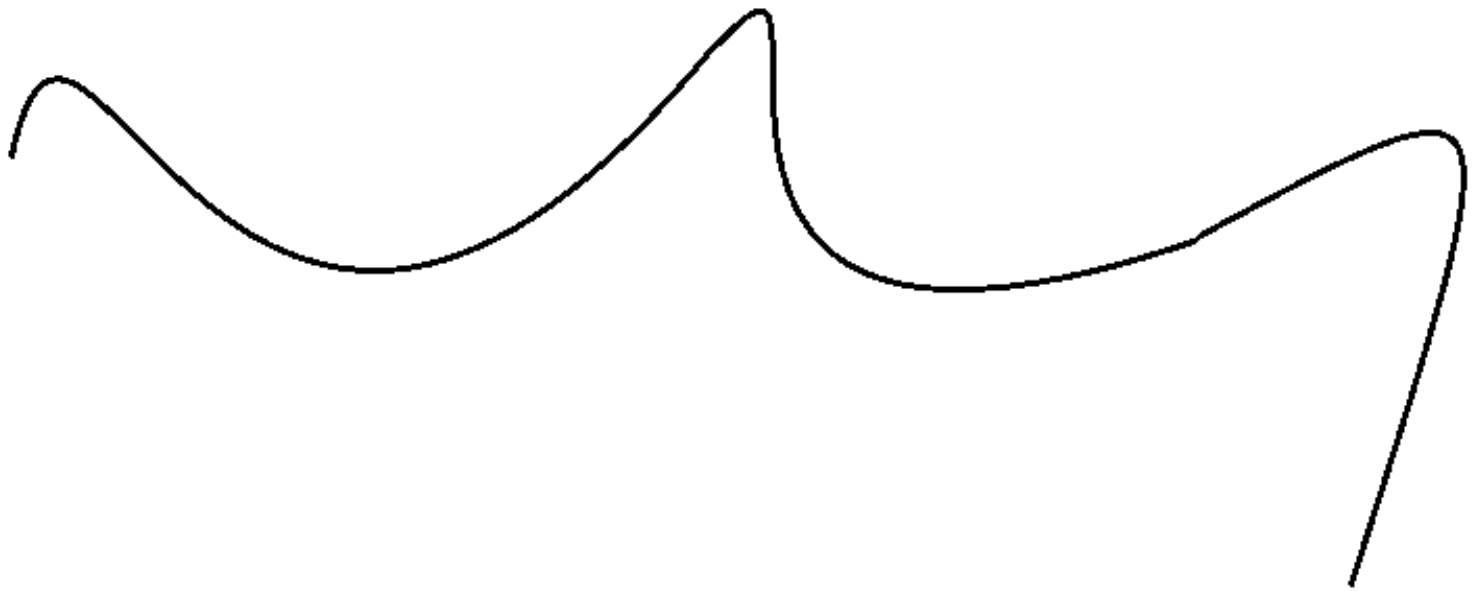


ACM © 1987 "Principles of traditional animation applied to 3D computer animation"

How Many Dimensions?

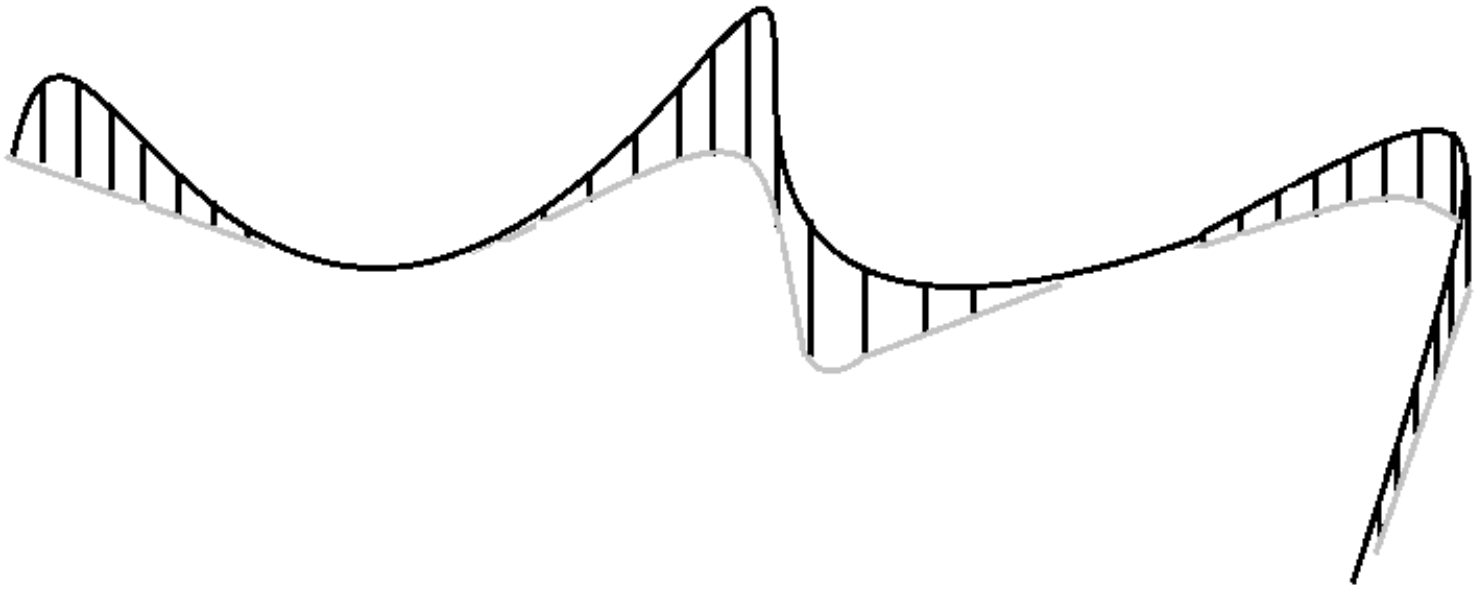


How Many Dimensions?



**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

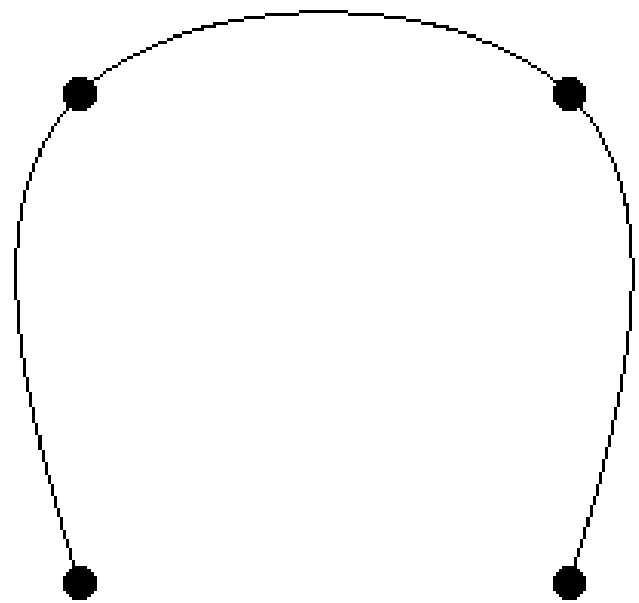
- A continuous 1D set of points in 2D (or 3D)
- A mapping from an interval S onto the plane
  - That is, *P(t)* is the point of the curve at parameter *t*

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

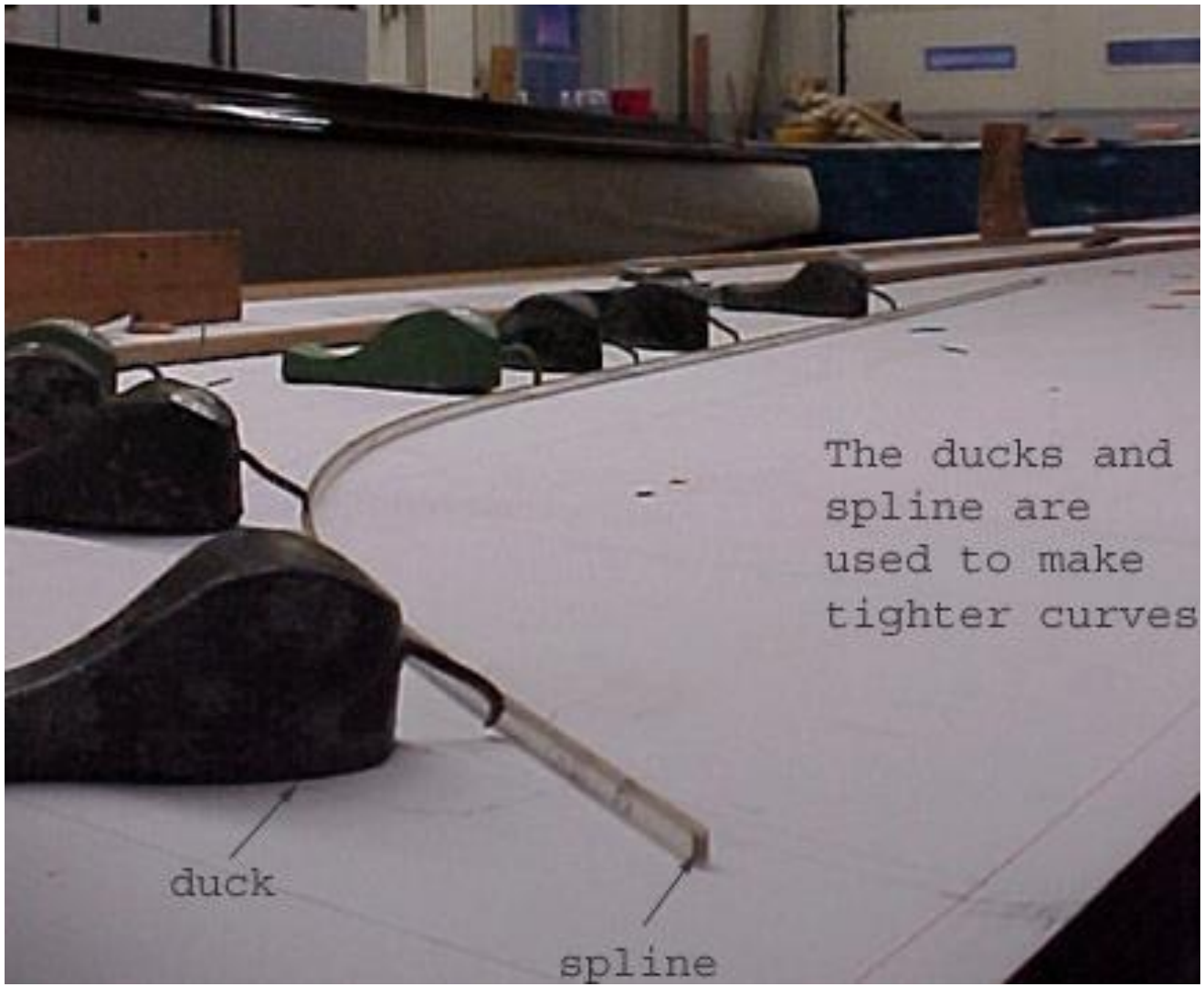
- Big differences
  - It is easy to generate points on the curve from the 2nd
  - The second definition can describe trajectories, the speed at which we move on the curve

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



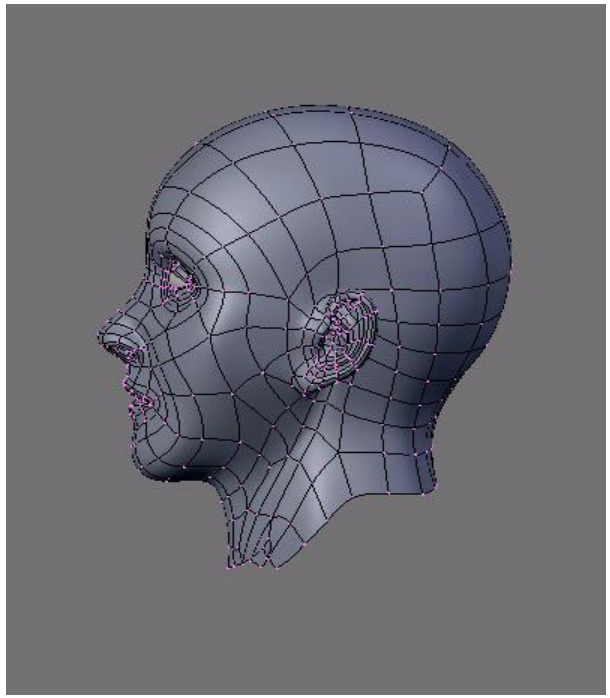
Courtesy of The Antique Boat Museum.

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.



Questions?