CSE 167: Introduction to Computer Graphics Lecture #11: Performance Optimization

Jürgen P. Schulze, Ph.D.
University of California, San Diego
Spring Quarter 2015

Announcements

- Homework 5 due tomorrow at 1pm
- Homework 6 due next Friday

Lecture Overview

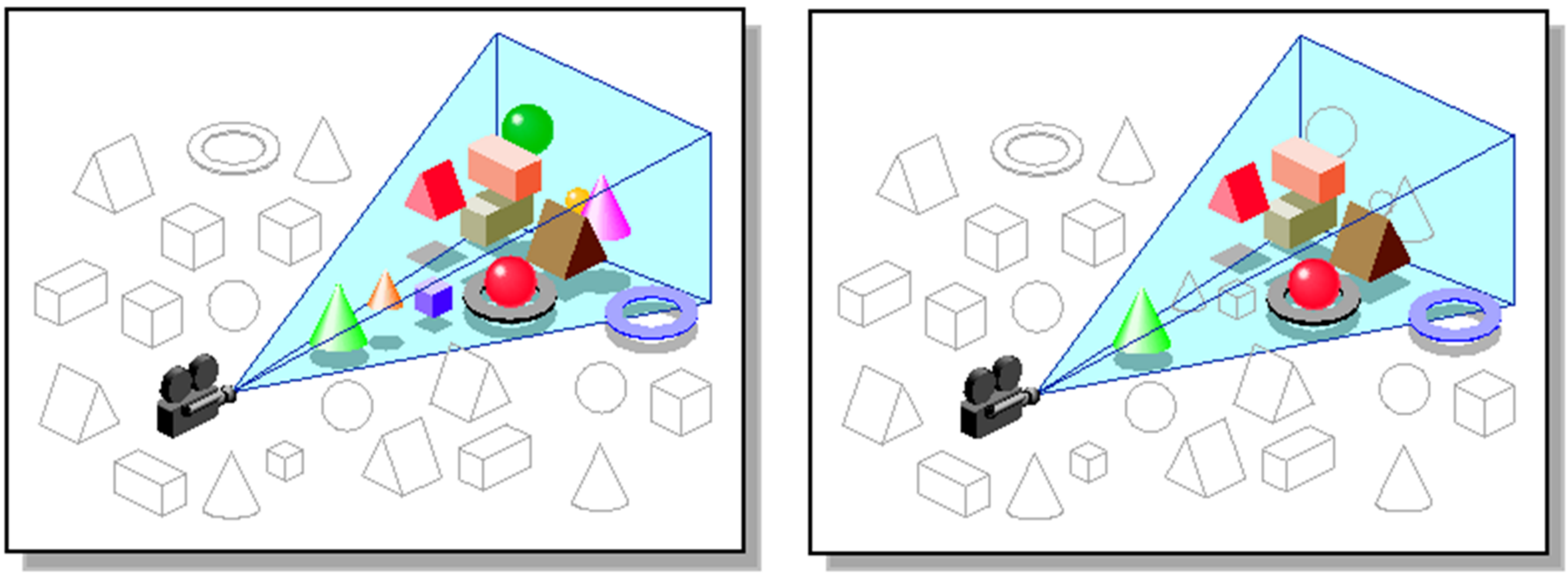
- Performance Optimization
  - Culling
  - Level of Detail Techniques

Culling

- Goal:
  Discard geometry that does not need to be drawn to speed up rendering
- Types of culling:
  - View frustum culling
  - Occlusion culling
  - Small object culling
  - Backface culling
  - Degenerate culling

Occlusion Culling

- Geometry hidden behind occluder cannot be seen
  - Many complex algorithms exist to identify occluded geometry



Images: SGI OpenGL Optimizer Programmer's Guide

UCSD

Video

- Umbra 3 Occlusion Culling explained
  - http://www.youtube.com/watch?v=5h4QgDBwQhc

UCSD

Small Object Culling

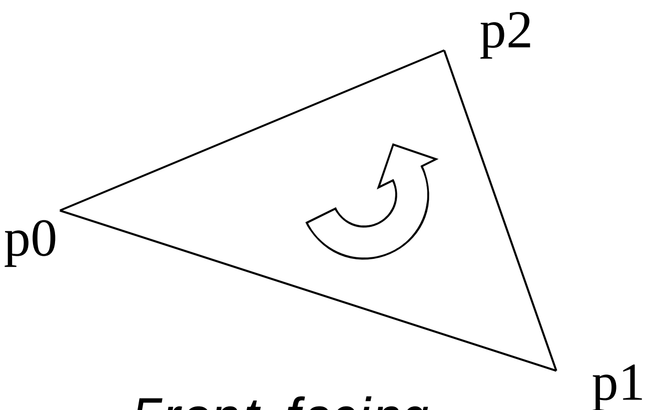
- Object projects to less than a specified size
  - Cull objects whose screen-space bounding box is less than a threshold number of pixels

Backface Culling

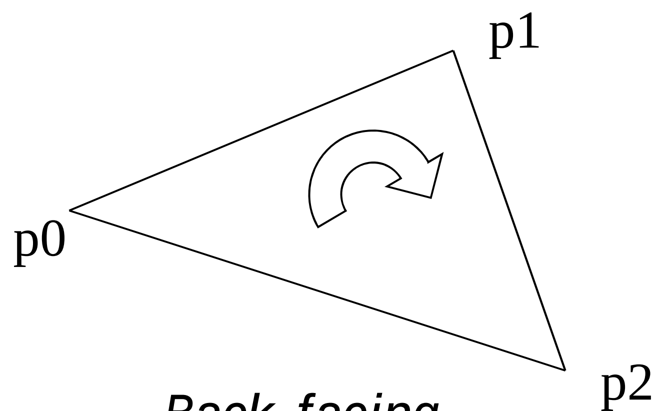
- Consider triangles as "one-sided", i.e., only visible from the "front"
- Closed objects
  - If the "back" of the triangle is facing the camera, it is not visible
  - Gain efficiency by not drawing it (culling)
  - Roughly 50% of triangles in a scene are back facing

Backface Culling

- Convention:
  Triangle is front facing if vertices are ordered counterclockwise



Front-facing



Back-facing

- OpenGL allows one- or two-sided triangles
  - One-sided triangles:
    glEnable(GL_CULL_FACE); glCullFace(GL_BACK)
  - Two-sided triangles (no backface culling):
    glDisable(GL_CULL_FACE)

Backface Culling

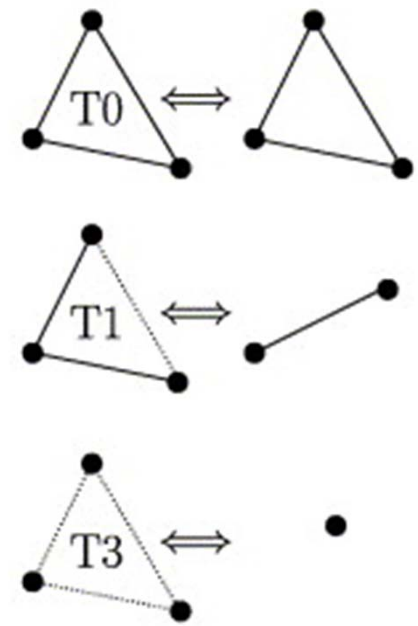
- Compute triangle normal after projection (homogeneous division)

$$\mathbf{n} = (\mathbf{p}_1 - \mathbf{p}_0) \times (\mathbf{p}_2 - \mathbf{p}_0)$$

- Third component of $\mathbf{n}$ negative: front-facing, otherwise back-facing
  - Remember: projection matrix is such that homogeneous division flips sign of third component

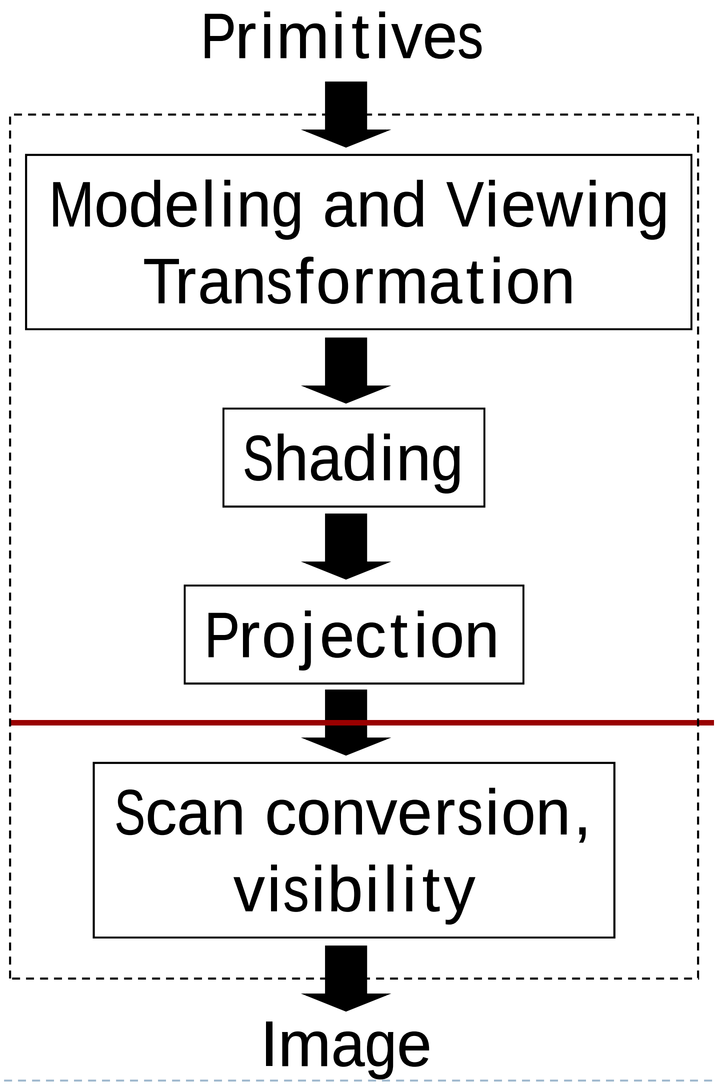
Degenerate Culling

- Degenerate triangle has no area
  - Vertices lie in a straight line
  - Vertices at the exact same place
  - Normal **n**=0



Source: Computer Methods in Applied Mechanics and Engineering, Volume 194, Issues 48–49

Rendering Pipeline

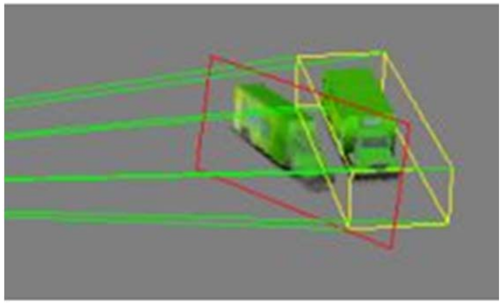


Culling, Clipping

- Discard geometry that will not be visible

Level-of-Detail Techniques

- Don't draw objects smaller than a threshold
  - Small feature culling
  - Popping artifacts
- Replace 3D objects by 2D impostors
  - Textured planes representing the objects



Impostor generation

Original vs. impostor

- Adapt triangle count to projected size

Size dependent mesh reduction
(Data: Stanford Armadillo)

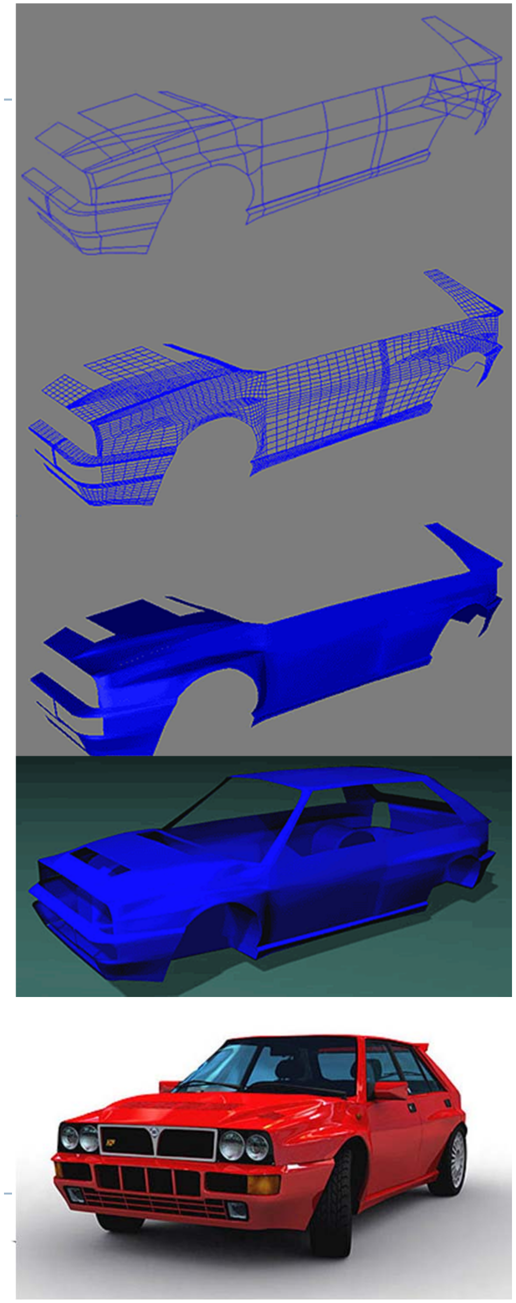
Lecture Overview

- Polynomial Curves
  - Introduction
  - Polynomial functions
- Bézier Curves
  - Introduction
  - Drawing Bézier curves
  - Piecewise Bézier curves

Modeling

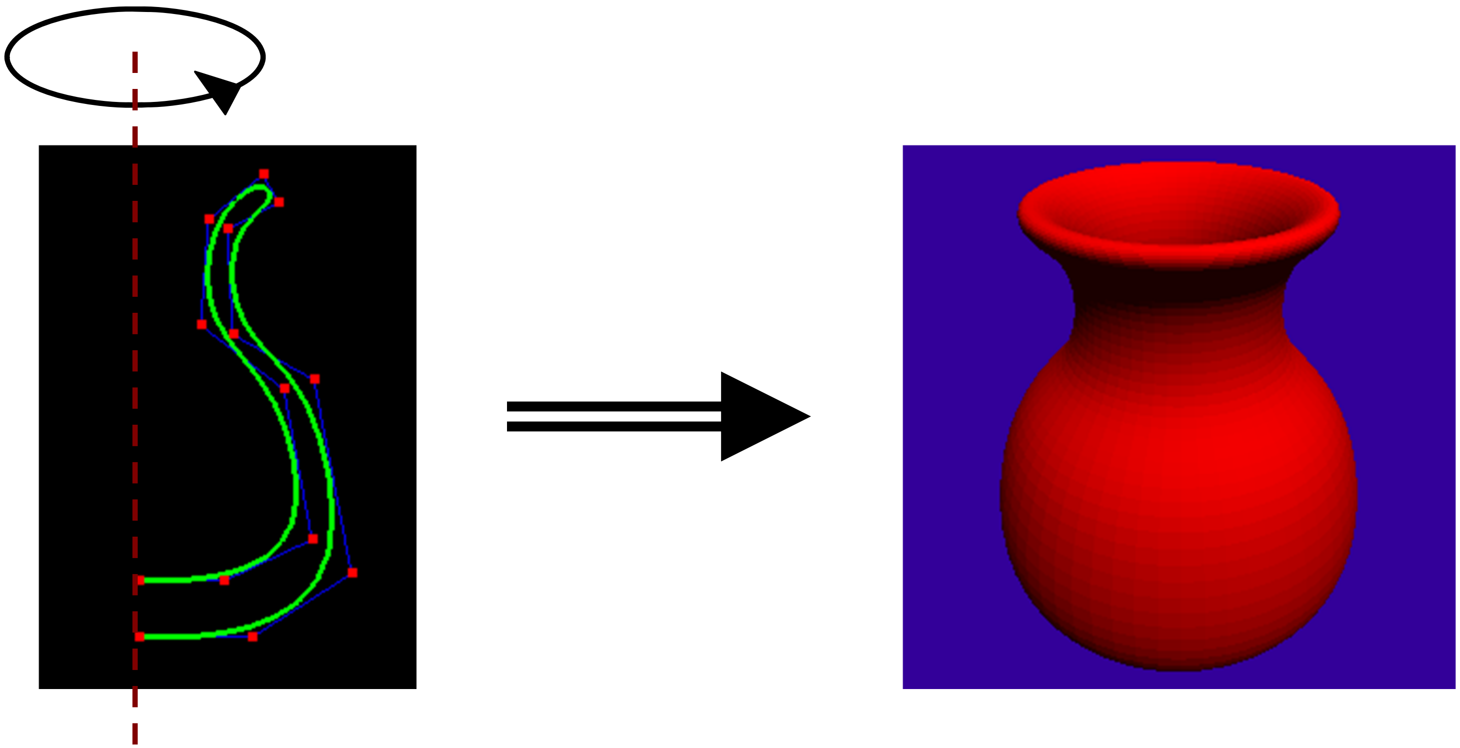
- Creating 3D objects
- How to construct complex surfaces?
- Goal
  - Specify objects with control points
  - Objects should be visually pleasing (smooth)
- Start with curves, then generalize to surfaces

- Next: What can curves be used for?



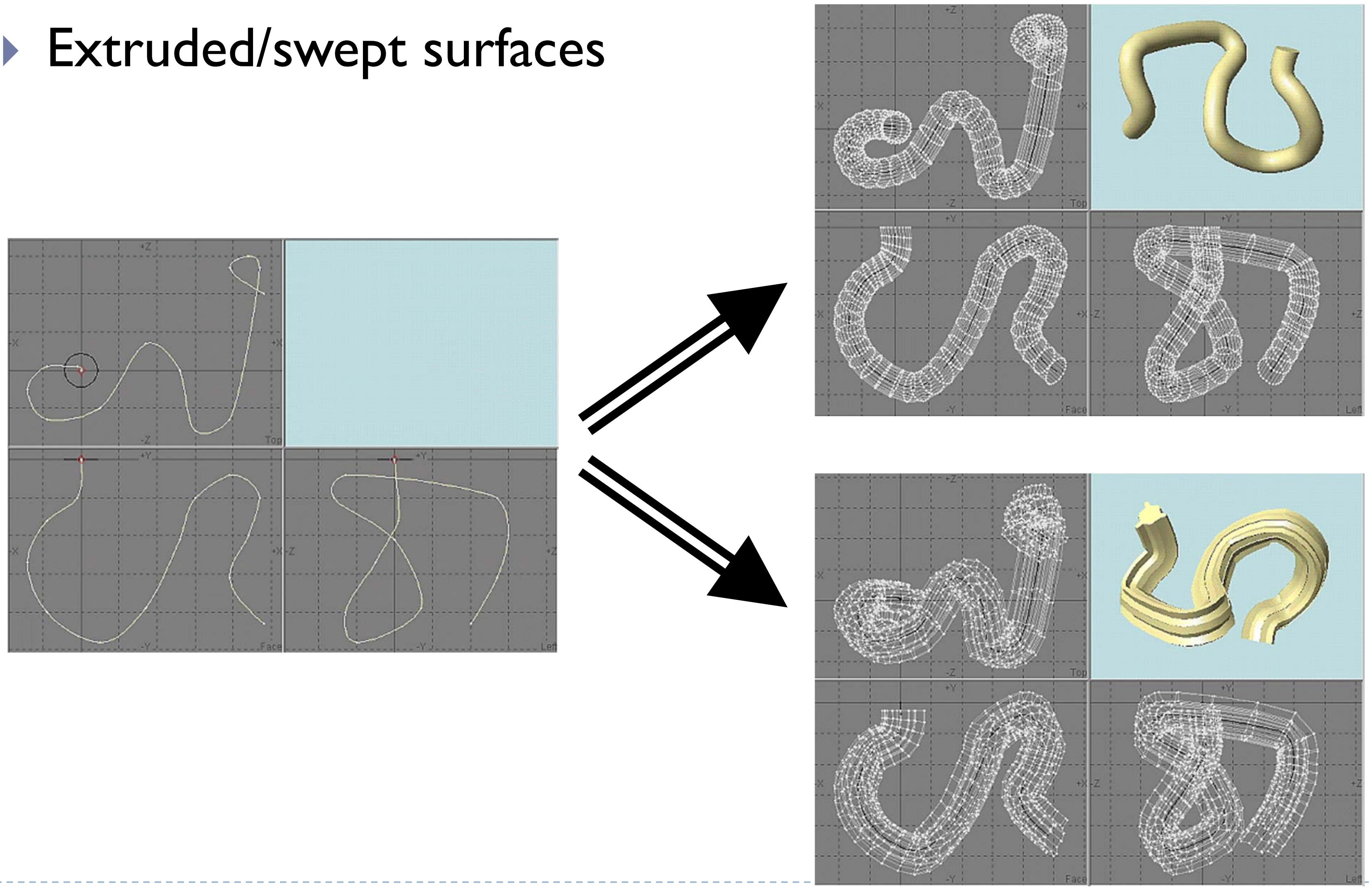
Curves

- Surface of revolution



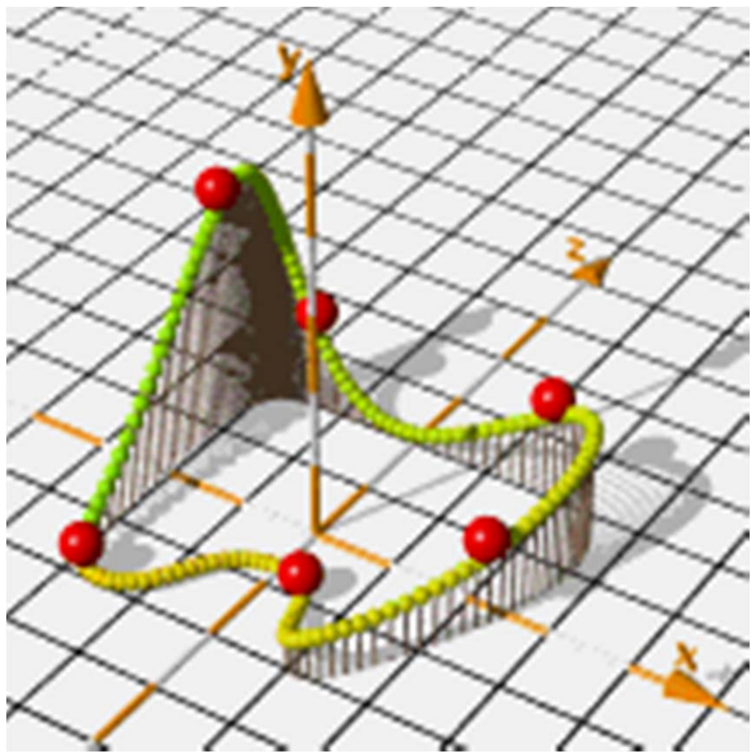
Curves

- Extruded/swept surfaces



Curves

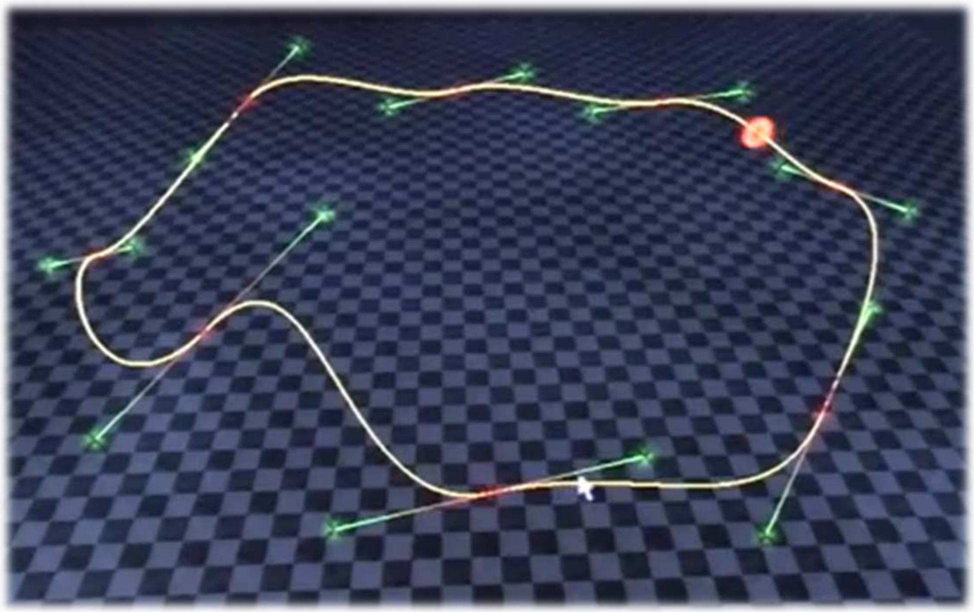
- Animation
  - Provide a "track" for objects
  - Use as camera path



UCSD

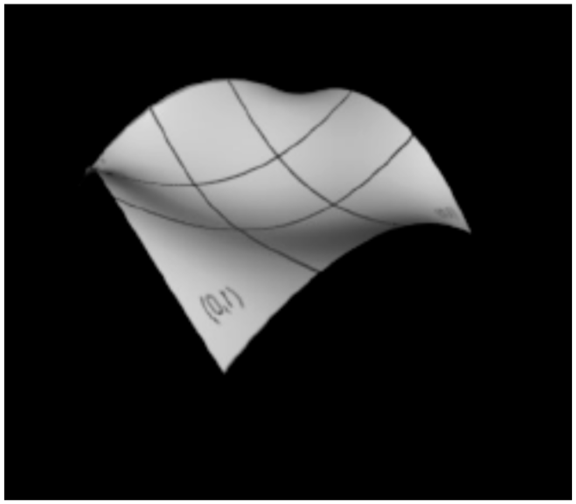
Video

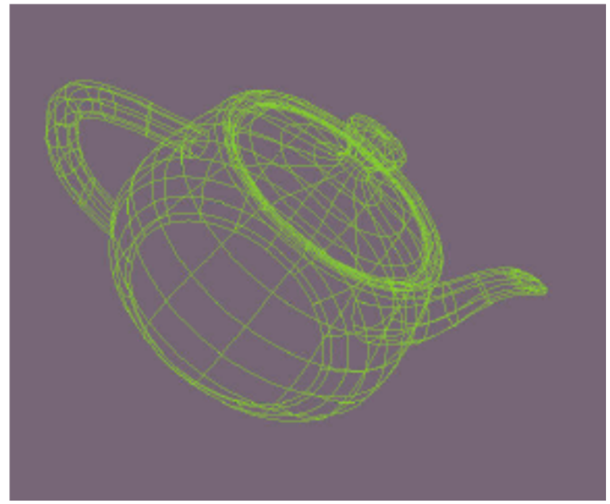
- Bezier Curves
  - http://www.youtube.com/watch?v=hlDYJNEiYvU



Curves

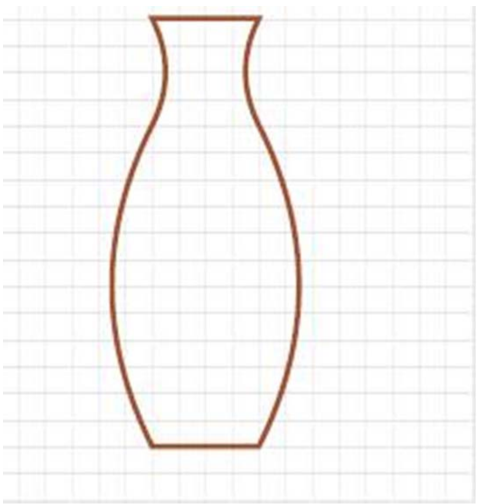
- Can be generalized to surface patches

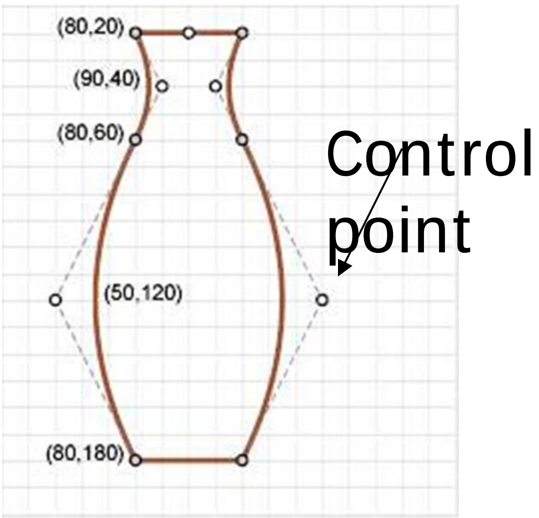




Curve Representation

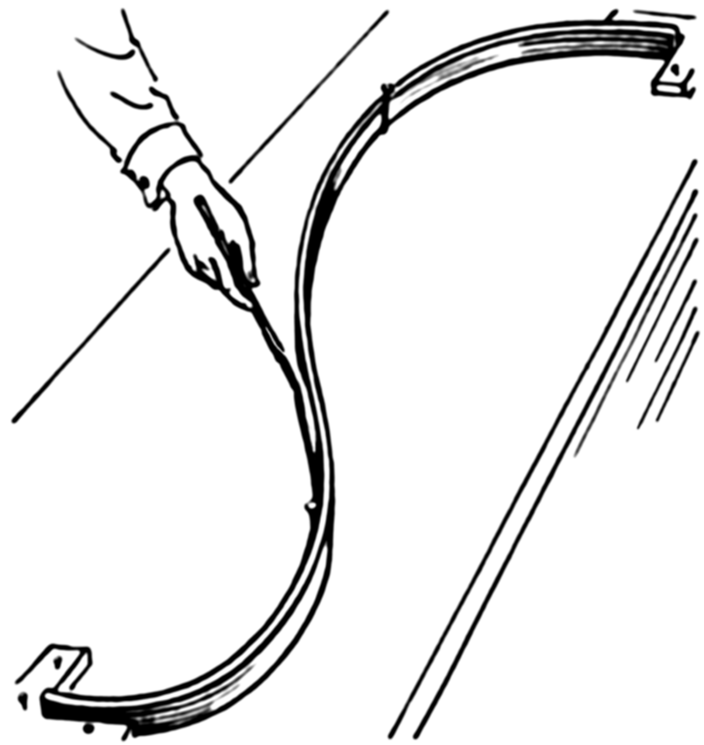
- Specify many points along a curve, connect with lines?
  - Difficult to get precise, smooth results across magnification levels
  - Large storage and CPU requirements
  - How many points are enough?
- Specify a curve using a small number of "control points"
  - Known as a *spline curve* or just *spline*





Spline: Definition

- Wikipedia:
  - Term comes from flexible spline devices used by shipbuilders and draftsmen to draw smooth shapes.
  - Spline consists of a long strip fixed in position at a number of points that relaxes to form a smooth curve passing through those points.

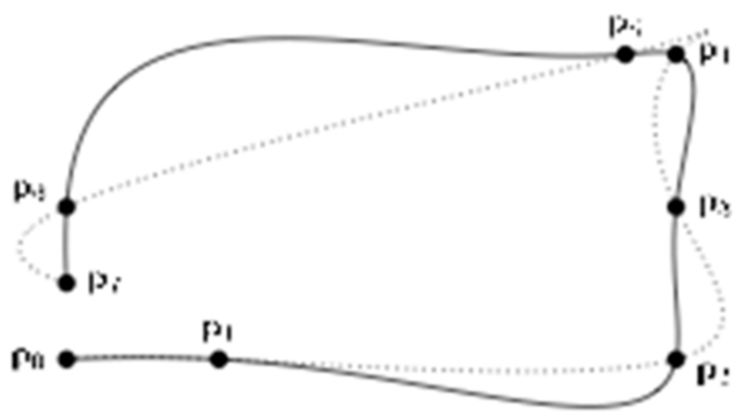


Lecture Overview

- Polynomial Curves
  - Introduction
  - Polynomial functions
- Bézier Curves
  - Introduction
  - Drawing Bézier curves
  - Piecewise Bézier curves

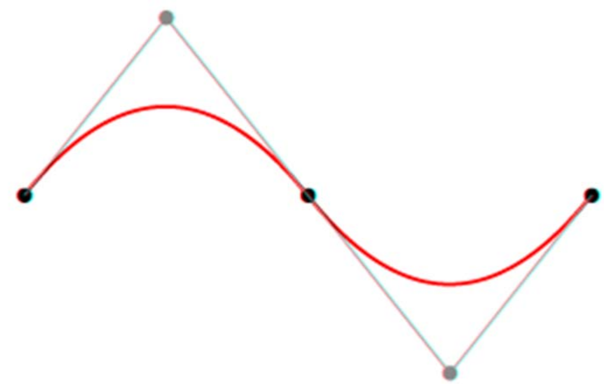
Interpolating Control Points

- "Interpolating" means that curve goes through all control points
- Seems most intuitive
- Surprisingly, not usually the best choice
  - Hard to predict behavior
  - Hard to get aesthetically pleasing curves



Approximating Control Points

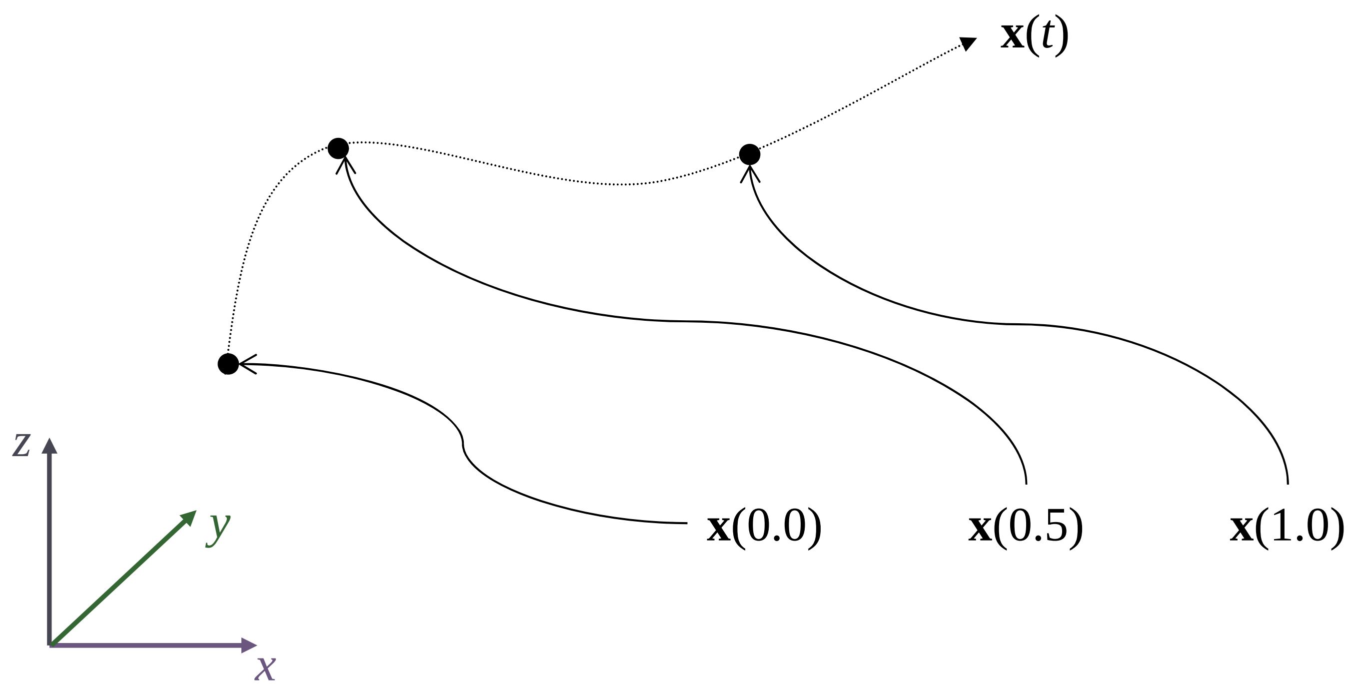
- Curve is "influenced" by control points



- Various types
- Most common: polynomial functions
  - Bézier spline (our focus)
  - B-spline (generalization of Bézier spline)
  - NURBS (Non Uniform Rational Basis Spline): used in CAD tools

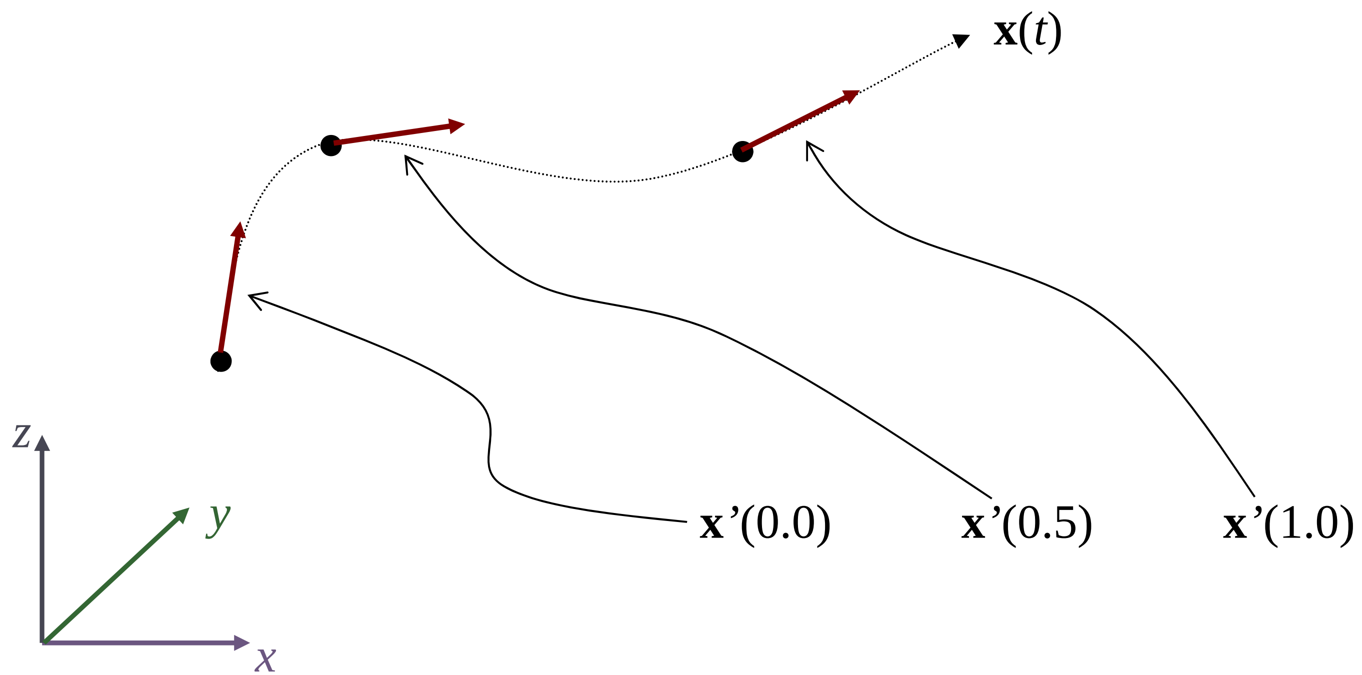
Mathematical Definition

- A vector valued function of one variable $\mathbf{x}(t)$
  - Given t, compute a 3D point $\mathbf{x}=(x,y,z)$
  - Could be interpreted as three functions: $x(t)$, $y(t)$, $z(t)$
  - Parameter t "moves a point along the curve"



Tangent Vector

- Derivative $\mathbf{x}'(t) = \frac{d\mathbf{x}}{dt} = (x'(t), y'(t), z'(t))$
- Vector x' points in direction of movement
- Length corresponds to speed

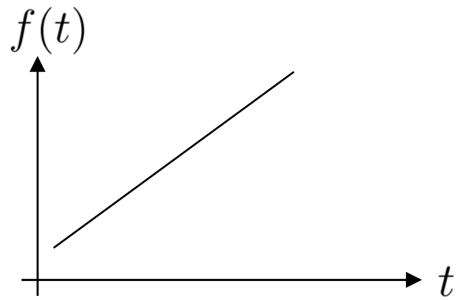


Lecture Overview

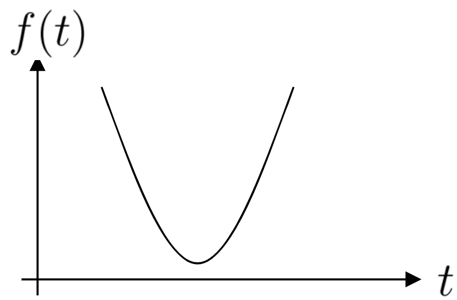
- Polynomial Curves
  - Introduction
  - Polynomial functions
- Bézier Curves
  - Introduction
  - Drawing Bézier curves
  - Piecewise Bézier curves

Polynomial Functions

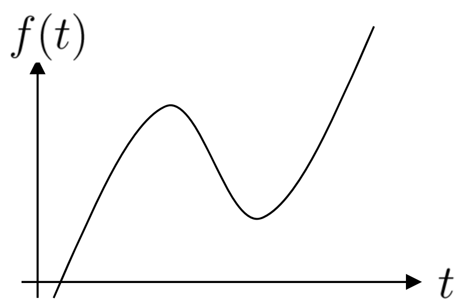
- Linear: $f(t) = at + b$
  (1st order)



- Quadratic: $f(t) = at^2 + bt + c$
  (2nd order)



- Cubic: $f(t) = at^3 + bt^2 + ct + d$
  (3rd order)



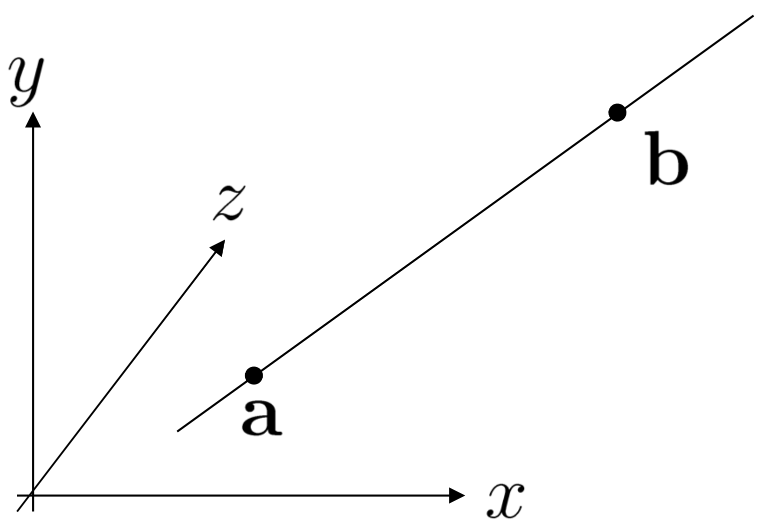
Polynomial Curves

- Linear $\mathbf{x}(t) = \mathbf{a}t + \mathbf{b}$

  $$\mathbf{x} = (x, y, z), \mathbf{a} = (a_x, a_y, a_z), \mathbf{b} = (b_x, b_y, b_z)$$

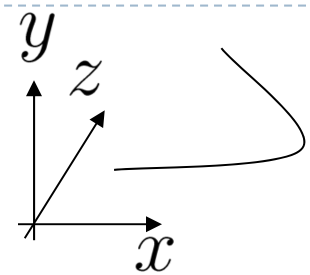
- Evaluated as:

$$x(t) = a_x t + b_x$$
$$y(t) = a_y t + b_y$$
$$z(t) = a_z t + b_z$$

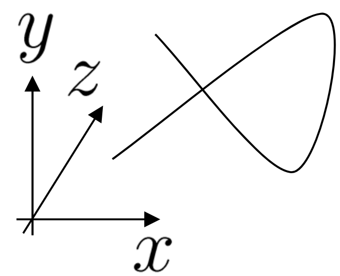


Polynomial Curves

- Quadratic: $\mathbf{x}(t) = \mathbf{a}t^2 + \mathbf{b}t + \mathbf{c}$
  (2nd order)



- Cubic: $\mathbf{x}(t) = \mathbf{a}t^3 + \mathbf{b}t^2 + \mathbf{c}t + \mathbf{d}$
  (3rd order)



- We usually define the curve for $0 \leq t \leq 1$

Control Points

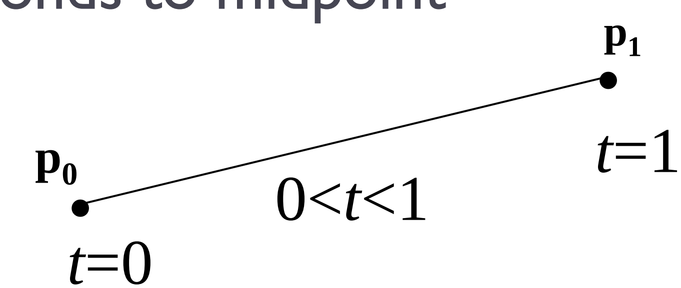
- Polynomial coefficients **a, b, c, d** can be interpreted as *control points*
  - Remember: **a, b, c, d** have x,y,z components each
- Unfortunately, they do not intuitively describe the shape of the curve
- Goal: intuitive control points

Control Points

- How many control points?
  - Two points define a line (1st order)
  - Three points define a quadratic curve (2nd order)
  - Four points define a cubic curve (3rd order)
  - $k+1$ points define a k-order curve
- Let's start with a line…

First Order Curve

- Based on linear interpolation (LERP)
  - Weighted average between two values
  - "Value" could be a number, vector, color, …
- Interpolate between points $\mathbf{p}_0$ and $\mathbf{p}_1$ with parameter t
  - Defines a "curve" that is straight (first-order spline)
  - $t=0$ corresponds to $\mathbf{p}_0$
  - $t=1$ corresponds to $\mathbf{p}_1$
  - $t=0.5$ corresponds to midpoint



$$\mathbf{x}(t) = Lerp\left(t,\ \mathbf{p}_0,\ \mathbf{p}_1\right) = \left(1-t\right)\mathbf{p}_0 + t\ \mathbf{p}_1$$

Linear Interpolation

- Three equivalent ways to write it
  - Expose different properties

1. Regroup for points **p**

$$\mathbf{x}(t) = \mathbf{p}_0(1-t) + \mathbf{p}_1 t$$

2. Regroup for t

$$\mathbf{x}(t) = (\mathbf{p}_1 - \mathbf{p}_0)t + \mathbf{p}_0$$

3. Matrix form

$$\mathbf{x}(t) = \begin{bmatrix} \mathbf{p}_0 & \mathbf{p}_1 \end{bmatrix} \begin{bmatrix} -1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} t \\ 1 \end{bmatrix}$$

Weighted Average

$$\mathbf{x}(t) = (1-t)\mathbf{p}_0 + \quad (t)\mathbf{p}_1$$
$$= B_0(t)\,\mathbf{p}_0 + B_1(t)\mathbf{p}_1, \text{ where } B_0(t) = 1-t \text{ and } B_1(t) = t$$

- Weights are a function of t
  - Sum is always 1, for any value of t
  - Also known as *blending functions*