CPSC 424
Curves:
Implicit vs. Explicit vs. Parametric

Syllabus

Curves in 2D and 3D

- **Implicit vs. Explicit vs. Parametric curves**
- Bézier curves, de Casteljau algorithm
- Continuity
- B-Splines
- Subdivision Curves

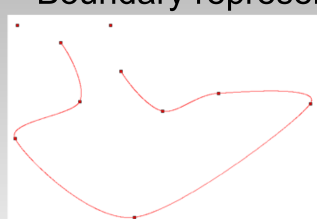
Properties of Curves and Surfaces

Surfaces/Meshes/Advanced Topics

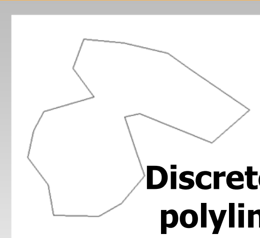
How to represent shape (2D or 3D)?

Geometry – Curves/2D shapes

- Boundary representations



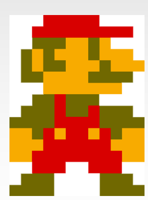
Freeform – splines



Discrete - polyline

- Alternative: 2D shapes

Primitive based

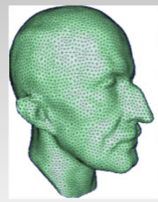


Pixel based

How to represent 3D shape?

- Most common: Boundary representations

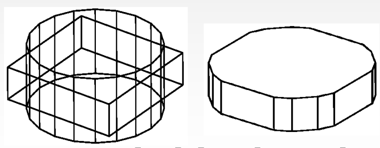
Freeform – smooth surface



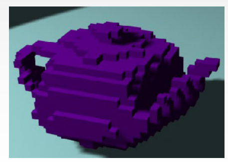
Mesh – polygonal surface

- Alternative: Volumetric representations

Voxel based



Primitive based



Modeling Geometry

Approaches:

- Fixed set of primitives
  - *Curves: lines, circles, rectangles…*
  - *Surfaces: spheres, cylinders…*
  - *Hard to assemble arbitrary (smooth) geometry*
- Freeform curves/surfaces
  - *Single representation for arbitrarily complex geometry*
  - *Curves and surfaces as functions with built-in smoothness properties*
  - *Bézier curves, splines*
- Discrete: meshes

Curves: Explicit vs. Implicit vs. Parametric

Curves & Surfaces as Explicit Functions

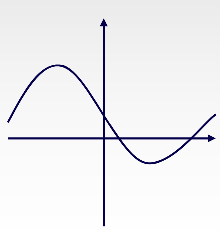
Curves:

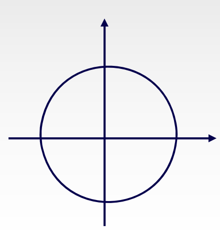
$$y = F(x)$$

Surfaces:

$$z = F(x, y)$$

Examples:



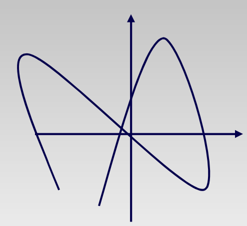


Not a function in Cartesian coord.,

$$y = \pm\sqrt{1 - x^2}$$

Curves & Surfaces as Explicit Functions

Not representable as a function:



Limitations of explicit functions:

- Cannot model every curve in 2D
- No true 3D curves possible
  - *All curves confined to a plane*

Curves & Surfaces as Implicit Functions

Curves

$$F(x, y) = 0$$

Surfaces

$$F(x, y, z) = 0$$

Interpretation for curves:

- Iso-lines (contours) in a terrain

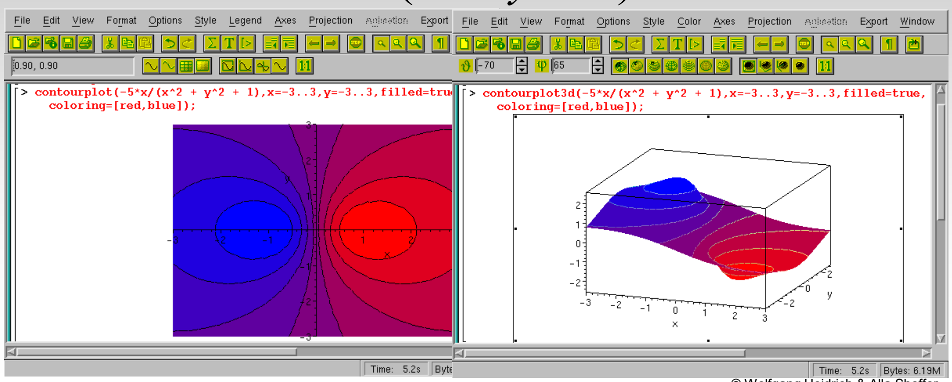
Property:

- If F is continuous, implicit curves and surfaces are always closed or extend to infinity

Curves & Surfaces as Implicit Functions

Examples:

- $x^2 + y^2 - 1 = 0$
- $-5x/(x^2 + y^2 + 1) = c$



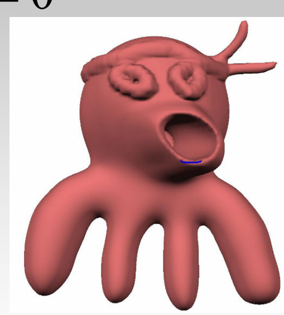
Curves & Surfaces as Implicit Functions

Conversion:

- Explicit to implicit: trivial $y - F(x) = 0$
- Implicit to explicit: hard
  - *Solving for y involves root finding!*

Limitations of implicits:

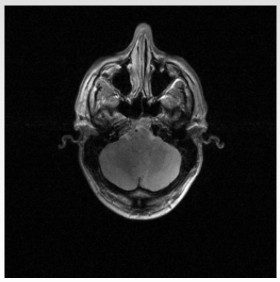
- Curves only in 2D
  - *Every implicit function in 3D describes a surface!*
- Often unintuitive
- Challenging to render (typically use ray tracing)
- Useful for many settings including modeling, machine learning



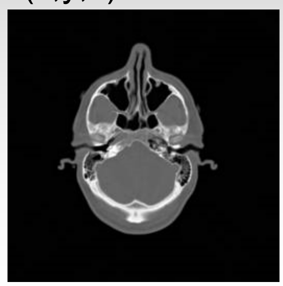
Implicit Functions in Medical Imaging

Data:

- CT & MRI scanners produce volume of *density* values F(x,y,z)
- Individual features (bone surface, brain surface) are iso-surfaces of the volume: F(x,y,z)=c



MRI



CT

Curves & Surfaces as Parametric Functions

Concept:

- Curve as function of artificial "time" parameter t

2D curve:

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} F_x(t) \\ F_y(t) \end{pmatrix} =: F(t); F : \mathcal{R} \mapsto \mathcal{R}^2$$

3D curve:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} F_x(t) \\ F_y(t) \\ F_z(t) \end{pmatrix} =: F(t); F : \mathcal{R} \mapsto \mathcal{R}^3$$

Curves & Surfaces as Parametric Functions

Curve example:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} \cos t \\ \sin t \\ t \end{pmatrix}$$

Surfaces (in 3D):

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} F_x(s,t) \\ F_y(s,t) \\ F_z(s,t) \end{pmatrix} = F(s,t); F : \mathcal{R}^2 \mapsto \mathcal{R}^3$$

Curves & Surfaces as Parametric Functions

This works in arbitrary dimensions!

- Curves:

$$\mathbf{x} = F(t); F : \mathcal{R} \mapsto \mathcal{R}^d$$

- Surfaces:

$$\mathbf{x} = F(s,t); F : \mathcal{R}^2 \mapsto \mathcal{R}^d$$

- Hypersurfaces:

$$\mathbf{x} = F(\mathbf{t}); F : \mathcal{R}^n \mapsto \mathcal{R}^d ; n < d$$

Notation:

- Bold variables ($\mathbf{t}, \mathbf{x}$) denote vectors, while italics denote scalars (t, d).

Parametric Curves

Advantage:

- Arbitrary curves in arbitrary dimensions

Still a problem:

- Unintuitive
  - *Try to find a formula for a specific curve you have in mind!*
- Hard to program with
  - *Deal with arbitrary mathematical functions*

Solution:

- Restrict yourself to specific class of functions

Spline Curves

Description = basis functions + coefficients

$$F(t) = \sum_{i=0}^{n} P_i B_i(t) = (x(t), y(t))$$

$$x(t) = \sum_{i=0}^{n} P_i^x B_i(t)$$

$$y(t) = \sum_{i=0}^{n} P_i^y B_i(t)$$

- Same basis functions for all coordinates

Example: Polynomial Curves

Polynomial Curves:

- Restrict to polynomial functions of degree ≤ m:

$$\mathbf{x} = \sum_{i=0}^{m} \mathbf{b}_i t^i$$

- Note: $\mathbf{b}_i$ are vectors!
- Example curve in 2D:

$$\begin{pmatrix} x \\ y \end{pmatrix} = \sum_{i=0}^{m} \begin{pmatrix} b_{x,i} \\ b_{y,i} \end{pmatrix} t^i$$

Polynomial Curves

Advantages:

- Computationally easy to handle
  - $\mathbf{P}_0 \ldots \mathbf{P}_n$ *uniquely describe curve (finite storage, easy to represent)*

Disadvantages:

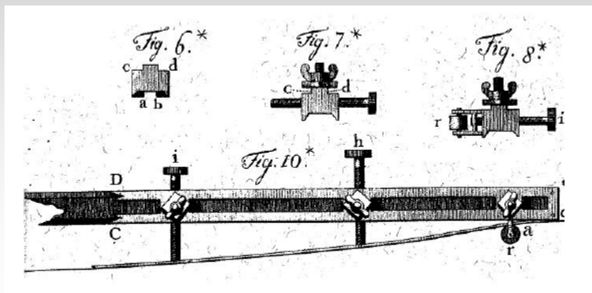
- Not all shapes representable

What basis functions B_i should we use?

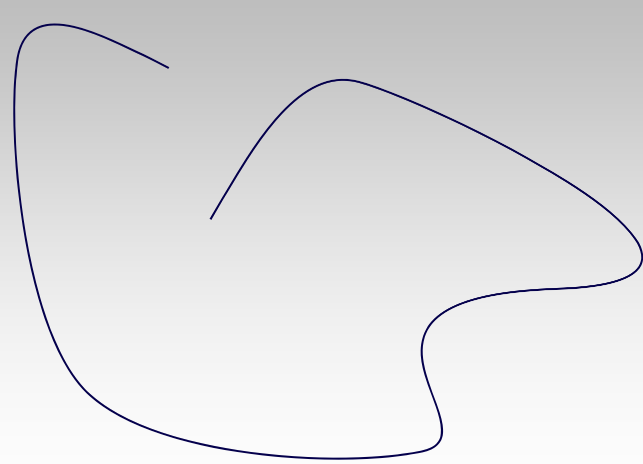
Splines: parametric curves over geometric base

Geometric meaning of coefficients (base)

- Approximate/interpolate set of positions, derivatives, etc..



Example: Curves in PowerPoint



Parametric Spline Curves

Commonly used classes:

- Polynomials
  - *Bézier curves, Hermite interpolation etc.*
- Piecewise polynomials
  - *B-splines*
- Rational and piecewise-rational curves
  - *Rational Bézier curves, rational B-splines (NURBS)*