Introduction to Computer Graphics

Marie-Paule Cani & Estelle Duveau

04/02 Introduction & projective rendering

11/02 **Prodedural modeling, Interactive modeling with parametric surfaces**

25/02 Introduction to OpenGL + lab: first steps & modeling

04/03 Implicit surfaces 1 + lecture/lab: transformations & hierarchies

11/03 Implicit surfaces 2 + Lights & materials in OpenGL

18/03 Textures, aliasing + Lab: Lights & materials in OpenGL

25/03 Textures in OpenGL: lecture + lab

01/04 Procedural & kinematic animation + lab: procedural anim

08/04 Physics: particle systems + lab: physics 1

22/04 Physics: collisions, control + lab: physics 2

29/04 Animating complex objects + Realistic rendering

06/05 **Talks: results of cases studies**

Modeling techniques for Computer Graphics

0. *Reconstruction*
   - *From real data* *Next year, in other courses?*

1. **Procedural modeling**
   - Automatic modeling of a self-similar objects or scenes

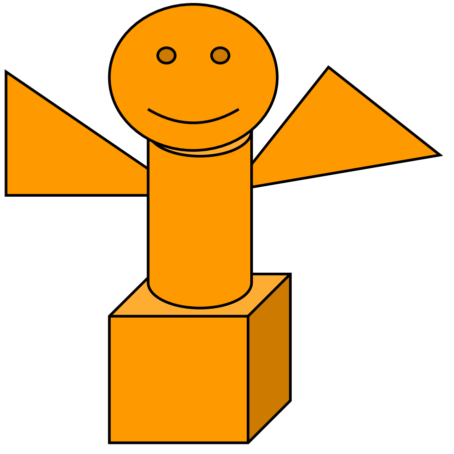
2. **Interactive modeling**
   - Provide tools for computer artists

Procedural modeling

Geometric primitives created by a program

- The oldest modeling method!
  - Basic example: OpenGL programming
  - Extension: use a description file

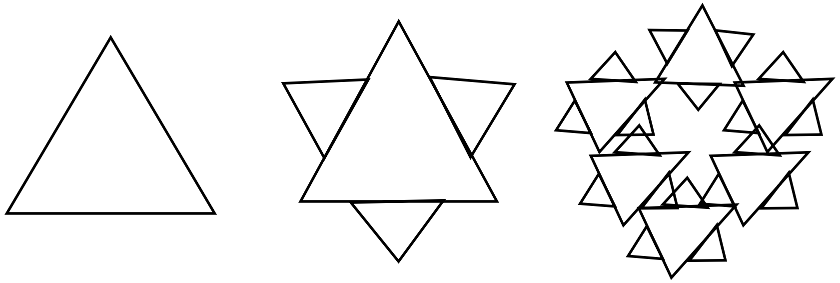
    Write/read parameters, not geometry!



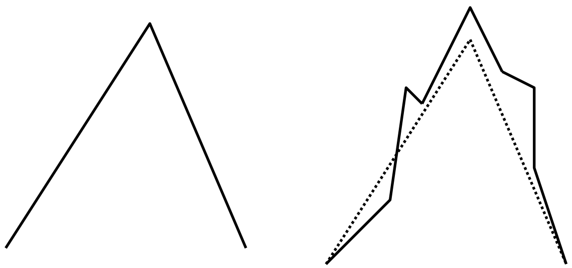
- Useful for large, repetitive scenes

Procedural modeling

- Example: Fractals
  - Recursively add details



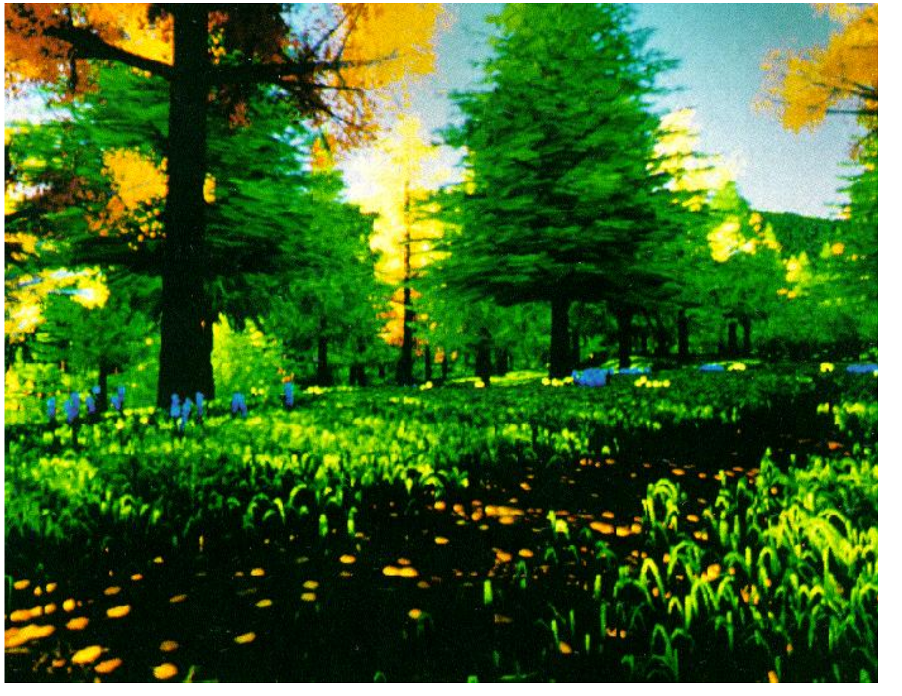
- Application to terrains
  - Add random displacements at each iteration



Procedural modeling

Reeve's natural scene

SIGGRAPH 1982

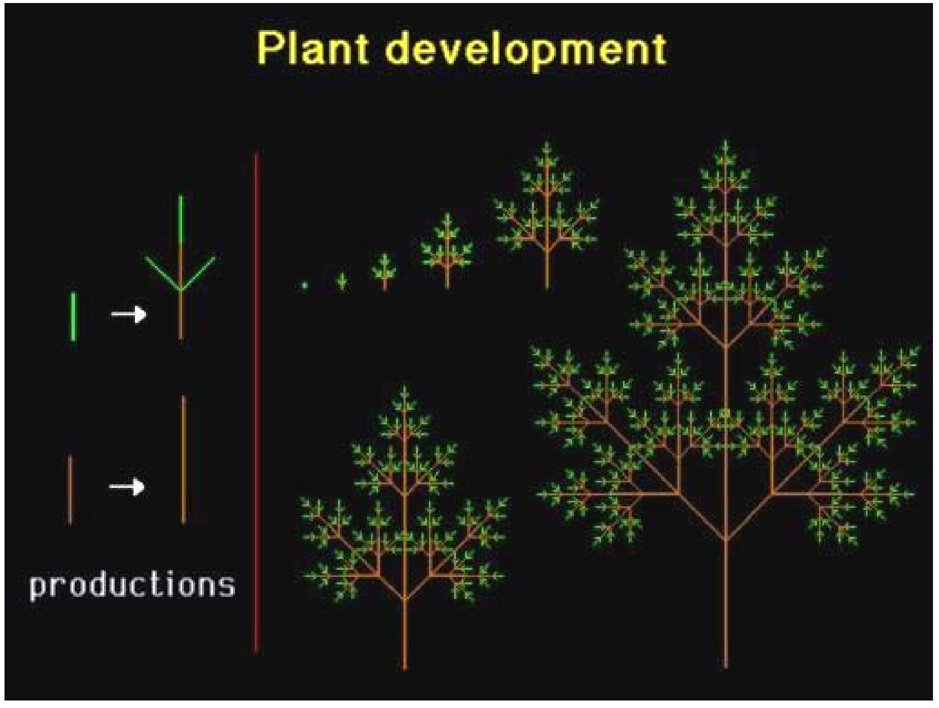


- Grass: particles under gravity
- Wind particles interact with grass
- Trees: recursively throw particles

The most complex scene ever built at that time!!

Procedural modeling

- Most common method for plants : L-systems
  - Simulate progressive growing using grammar rules



Inspired from biology!

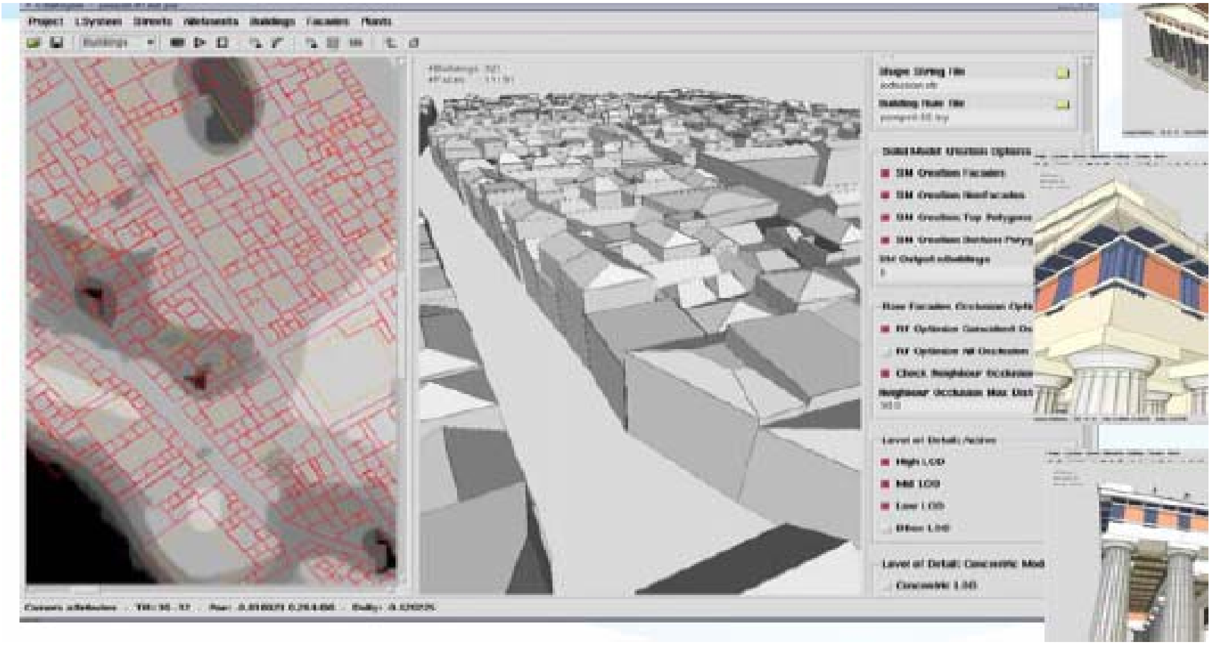
Procedural modeling

- Generalization: modeling with grammars
  1. Set of shapes
  2. Rules (take one shape and replace it with other shapes)
     – Apply rules with a given probability
     – Use random parameters in the created shapes
  3. Derivation (until « terminal shapes » only)
  4. Geometrical interpretation of the terminal shapes

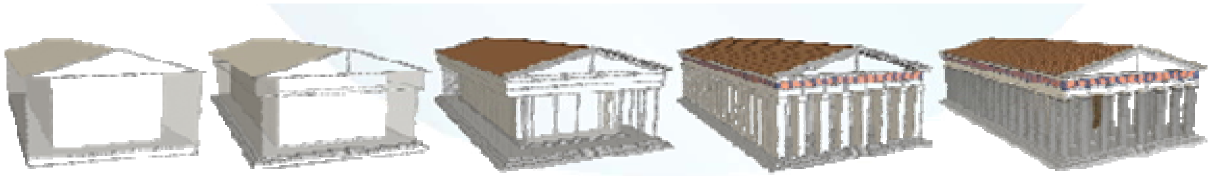
Many applications!

Procedural modeling

Example: cities

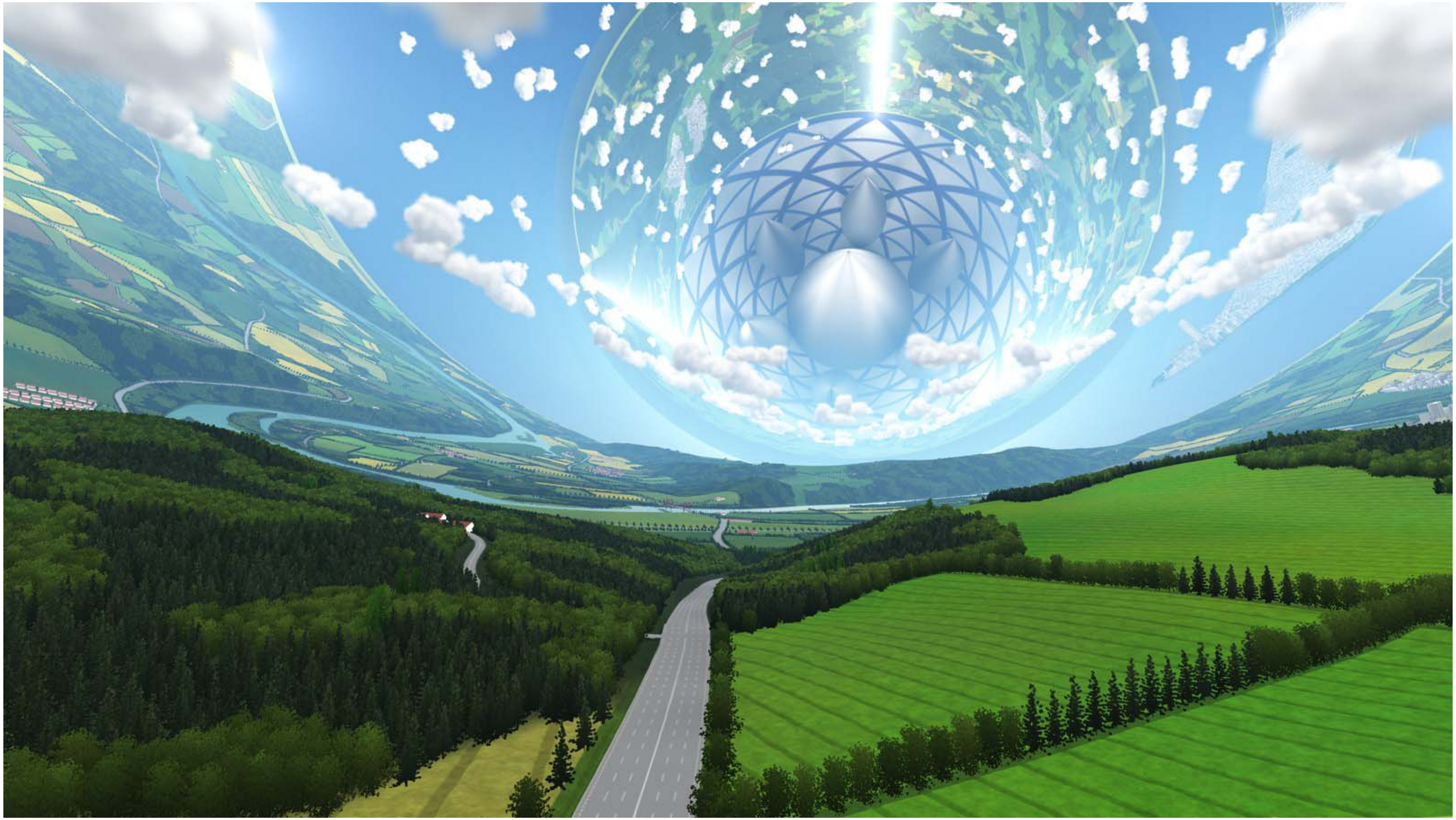


The method works at many different scales!



«Rama» 2006, Eric Bruneton

Procedural modeling



Modeling techniques

0. Reconstruction

- *From real data (not covered)*

1. Procedural modeling
   - Automatic modeling of a self-similar objects or scenes
2. **Interactive modeling**
   - Provide tools for computer artists

Interactive Modeling

Aim : Enable artists to create & refine the *shape they have in mind*

Humans **model shape indirectly**

- Input/output for dance, for music
- No output for shapes!
  - Use hands & tools
  - Create via a medium

Can we do this on a computer?

Store, undo/redo, cut, copy/paste, refine, deform, edit at any scale

Shape representation for shape design?

- Boundary representations (surfaces)
  - Polygons (discrete surfaces)
  - Splines, NURBS
  - Subdivision & multi-résolution surfaces
- Volumetric representations
  - Voxels (discrete volumes)
  - CSG (Constructive Solid Geometry)
  - Implicit surfaces

Most of them not introduced to ease interactive design !

Shape representation for shape design?

Criteria?

1. Real-time display after each interaction
2. No restriction on the created shape
   - Geometry: holes, branches, details…
   - Topology: any genius, allow topological changes
3. Avoid unnecessary degrees of freedom
   - Ex: closed objects: volumes vs. surfaces
4. Allow long modeling sessions
   - Complexity function of shape, not of user gestures!
5. Local & global, constant volume deformations

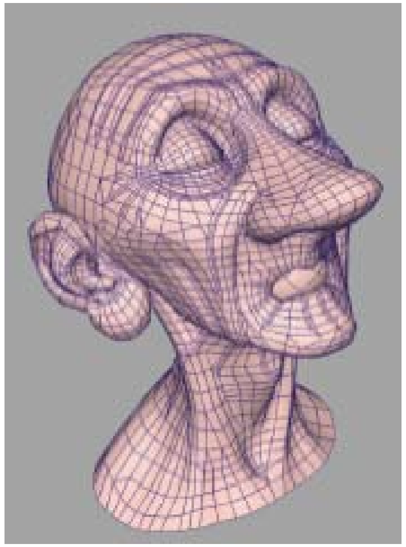
Choice of a representation?

Notion of ‘geometric model’

– Mathematical description of a virtual object (enumeration/equation of its surface/volume)

How should we represent this object…

– To get something smooth where needed ?
– To have some real-time display ?
– To save memory ?
– To ease subsequent deformations?



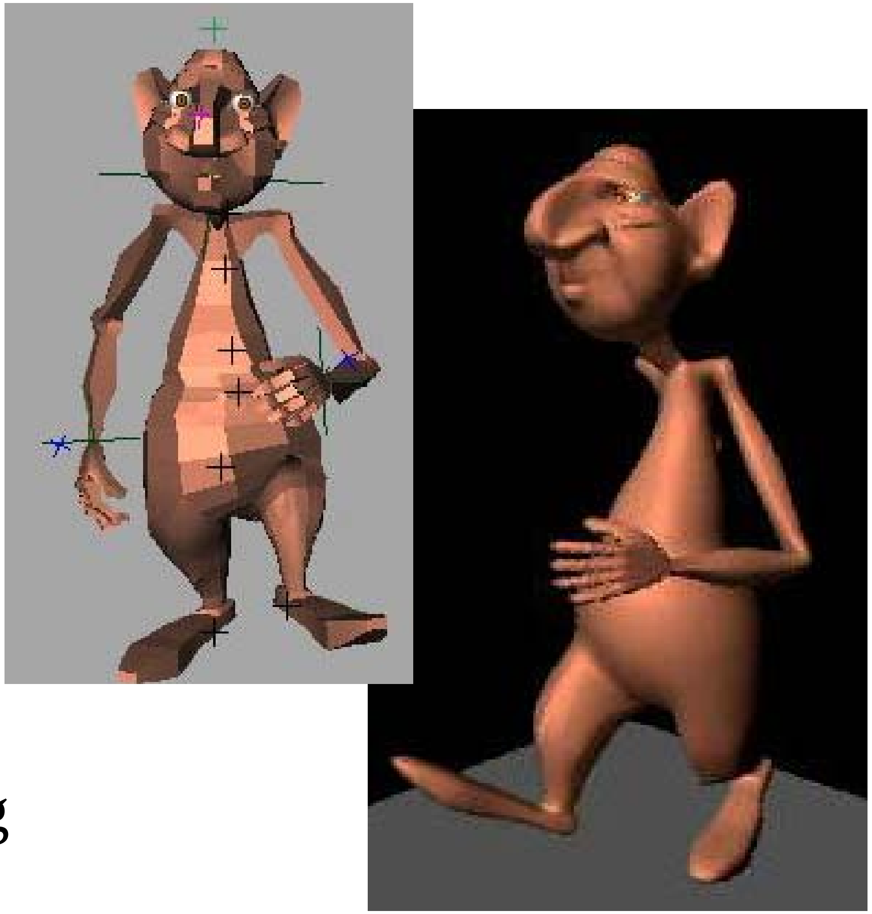
Why do we need Smooth Surfaces ?

Meshes

- Explicit enumeration of faces
- Many required to be smooth!
- Smooth deformation???

Smooth surfaces

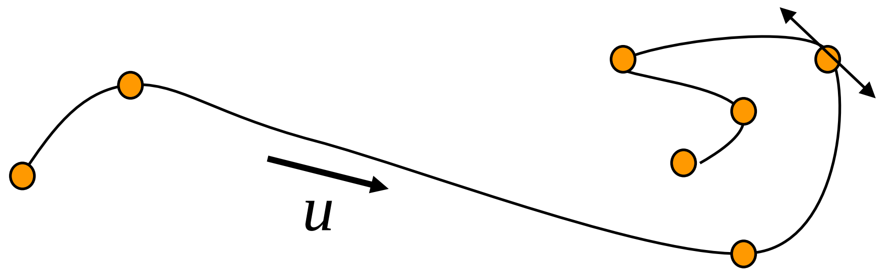
- Compact representation
- Will remain smooth
  - After zooming
  - After any deformation!
- Converted into faces for rendering



Parametric curves and surfaces

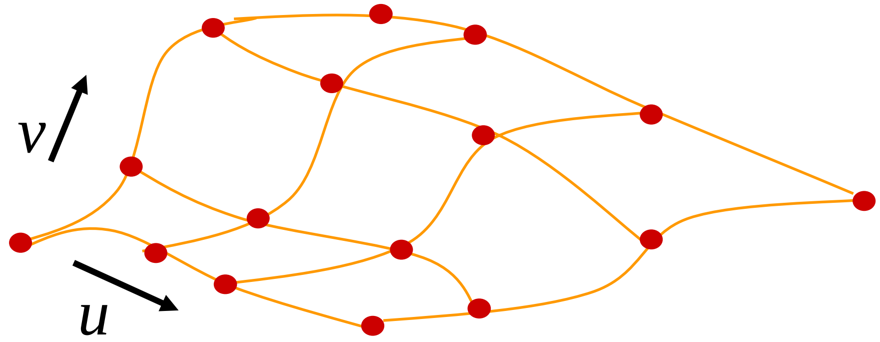
Defined by a parametric equation

- Curve: $C(u)$
- Surface: $S(u,v)$



Advantages

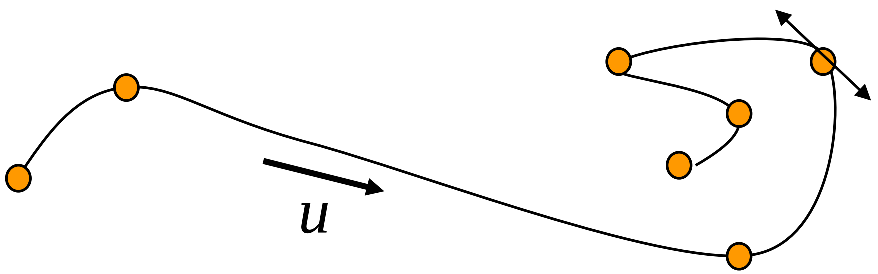
- Easy to compute points
- Easy to discretize
- Parametrization



Parametric curves: Splines

Motivations : interpolate/approximate points P_k

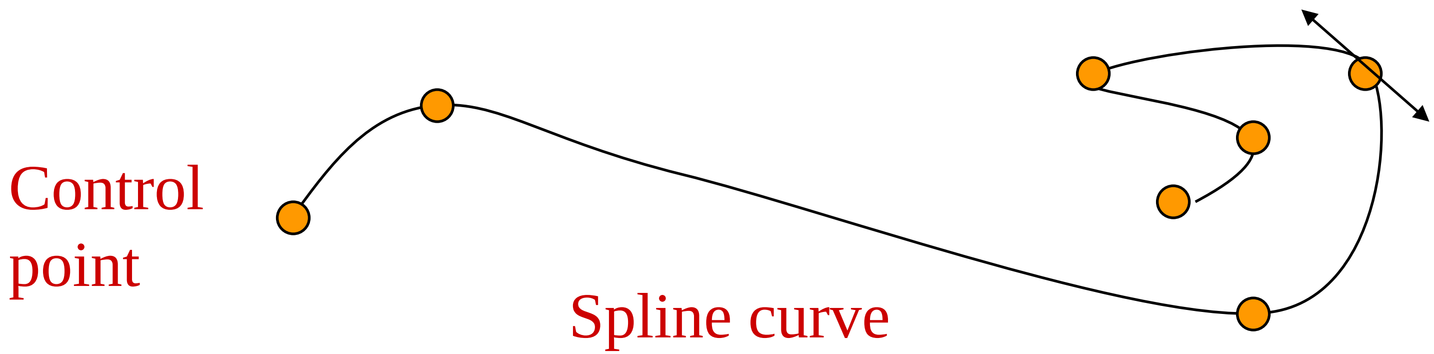
- Easier too give a finite number of “control points”
- The curve should be smooth in between



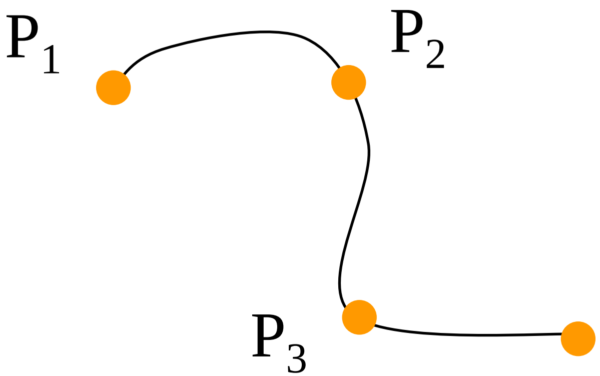
Why not polynomials? Which degree would we need?

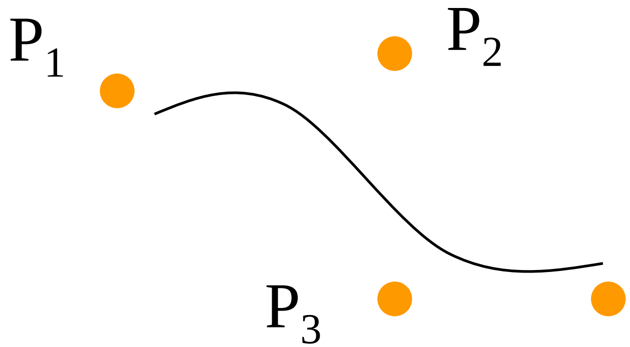
Spline curves

- Defined from control point
- Local control
  - Joints between polynomial curve segments
  - degree 3, C^1 or C^2 continuity



Interpolation vs. Approximation





Splines curves
Most important models

- Interpolation
  - Hermite curves C^1, cannot be local if C^2
  - Cardinal spline (Catmull Rom)

- Approximation
  - Bézier curves
  - Uniform, cubic B-spline (unique definition, subdivision)
  - Generalization to NURBS

Cardinal Spline, with tension=0.5

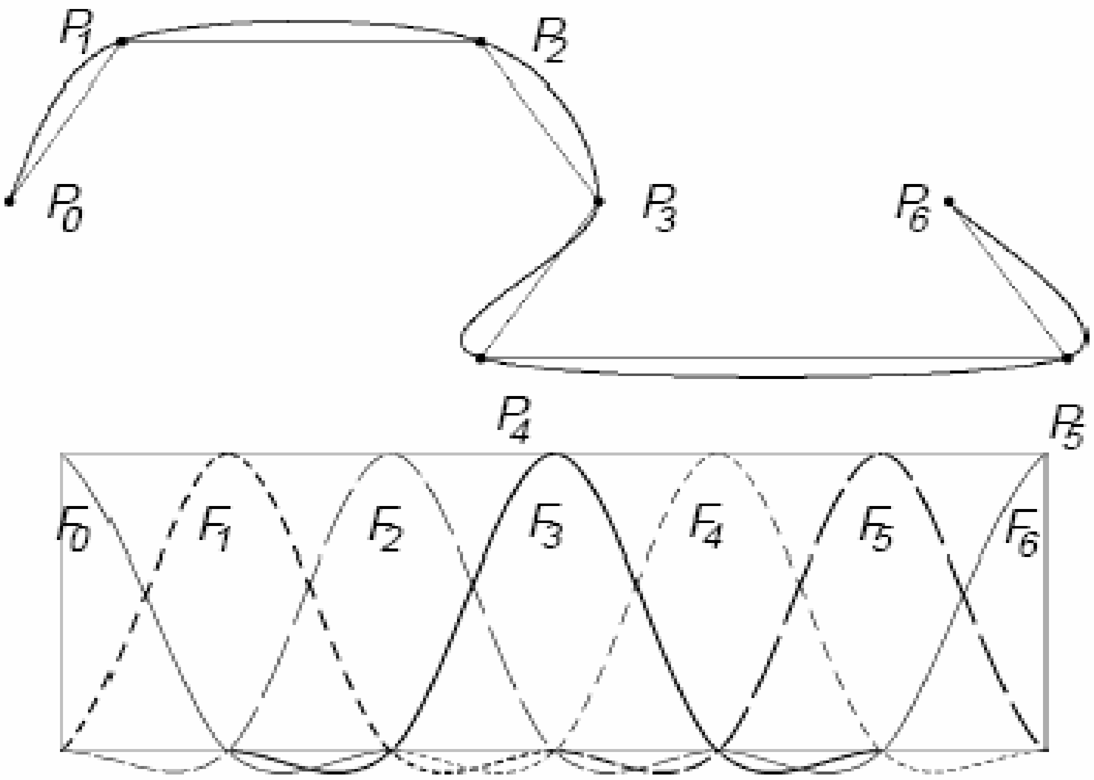


Figure 2: *Catmull-Rom spline curve*

Uniform, cubic Bspline

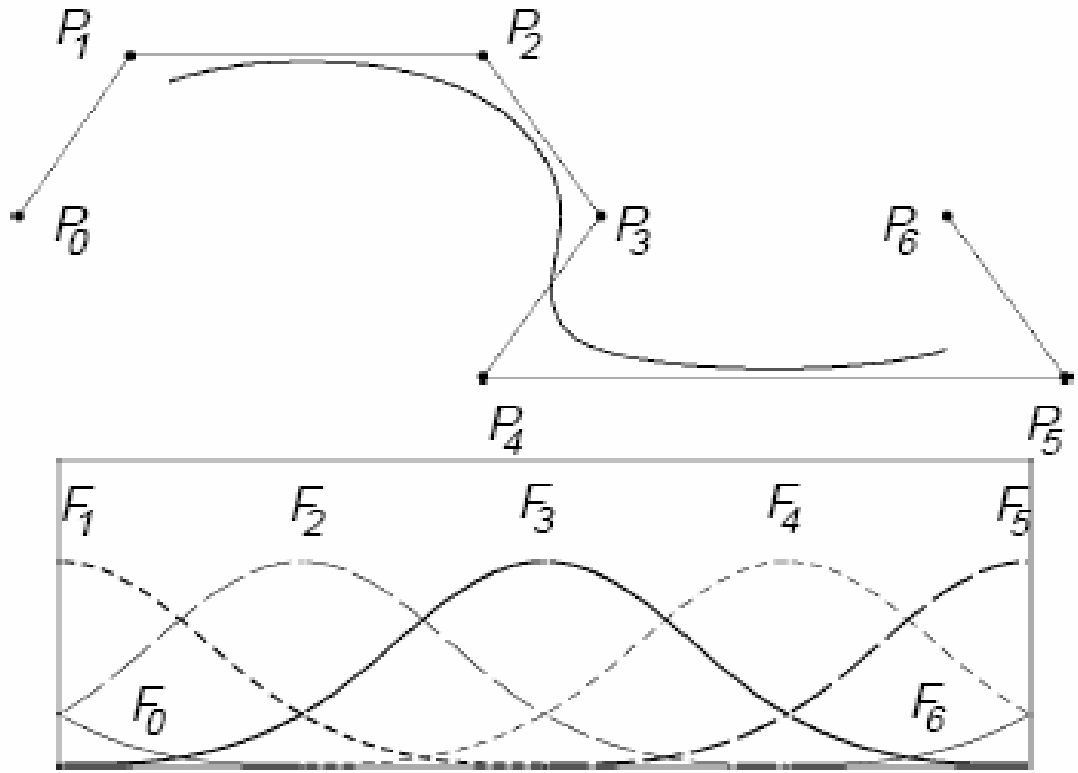


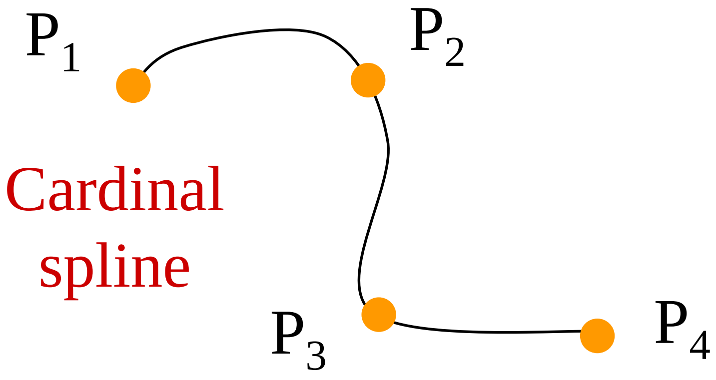
Figure 1: *Uniform B-spline curve*

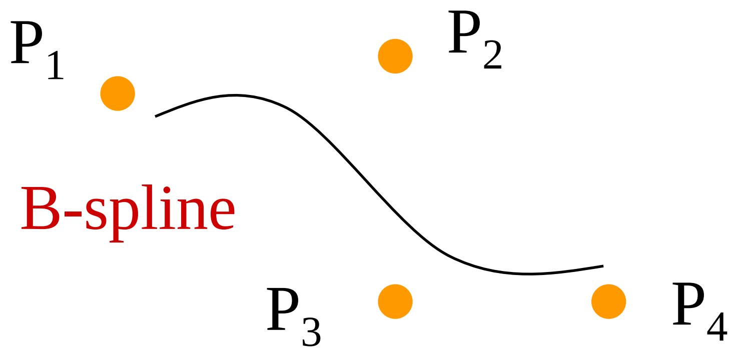
Cubic splines: matrix equation

$$Q_i(u) = (u^3\ u^2\ u\ 1)\ M_{spline}\ [P_{i-1}\ P_i\ P_{i+1}\ P_{i+2}]^t$$

$$M_{Catmull} = \frac{1}{2}\begin{bmatrix} -1 & 3 & -3 & 1 \\ 2 & -5 & 4 & -1 \\ -1 & 0 & 1 & 0 \\ 0 & 2 & 0 & 0 \end{bmatrix}$$

$$M_{Bspline} = \frac{1}{6}\begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix}$$





Splines surfaces

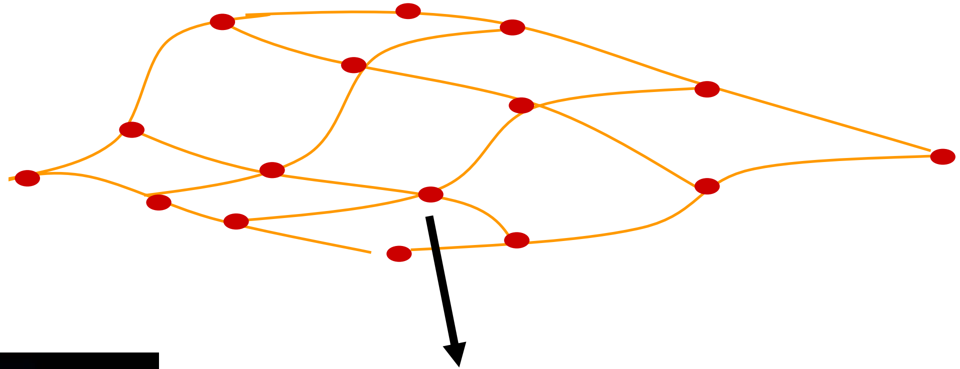
« Tensor product »: product of spline curves in *u* and *v*

$$Q_{i,j}(u, v) = (u^3\ u^2\ u\ 1)\ M\ [P_{i,j}]\ M^t\ (v^3\ v^2\ v\ 1)$$

– Smooth surface?

– Convert to meshes?

– Locallity?



Local deformation

Historic example

Interactive Modeling

Make it intuitive?

Inspire from real shape design!

Iterative shape design

1. Take or create simple shapes (“primitives”)
2. Deform them locally or globally
3. Assemble them

Iterate!

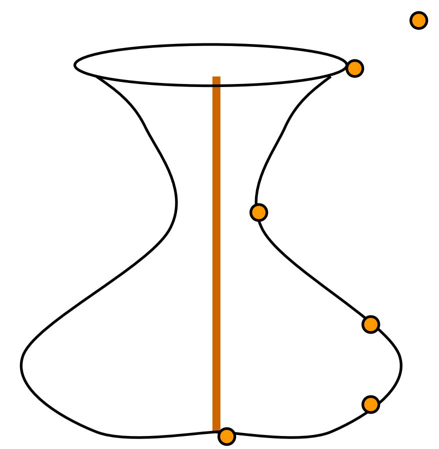
Parametric modeling
Step 1: Creating primitives

Difficult to specify 3D data with a mouse!

Idea: create shapes mostly from 2D input

1. Surfaces of revolution
   – Rotation of a planar profile around an axis

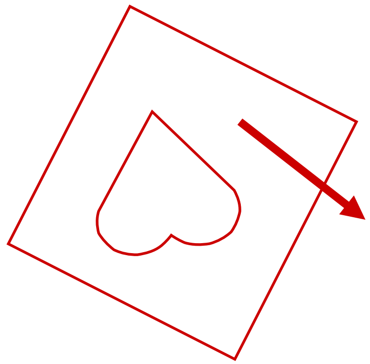
     Mesh; grid of control points…

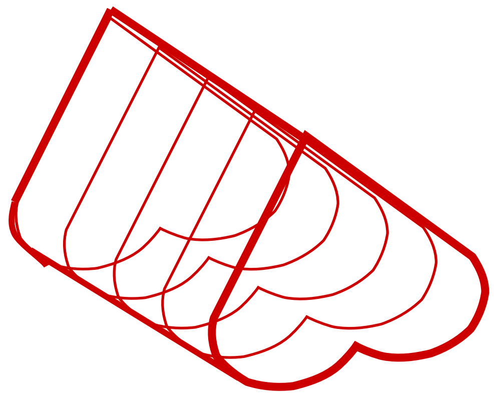


Parametric modeling
Step 1: Creating primitives

2. Lofting

- Data: a planar section, an axis
- Translated instances of the section





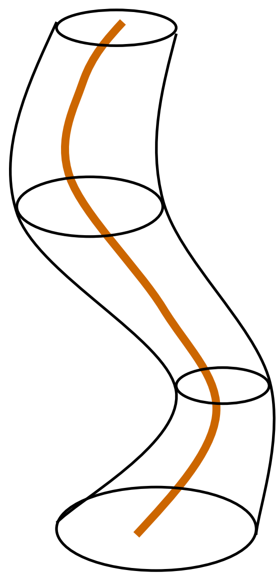
Generalization

“sweeping” gesture

Parametric modeling
Step 1: Creating primitives

3. Extrusion (also called "Free-form Sweeping")

- Data:
  - A planar cross section
  - A skeleton (3D curve)
  - A planar profile



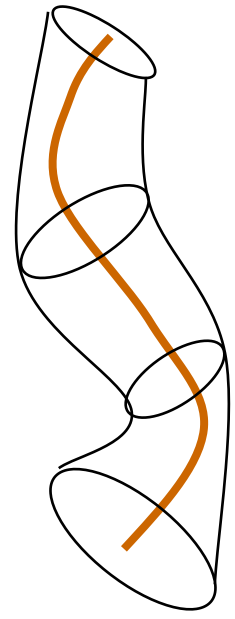
- The section is swept along the skeleton
- The profile is used as a scaling factor

Extrusion

Naïve idea

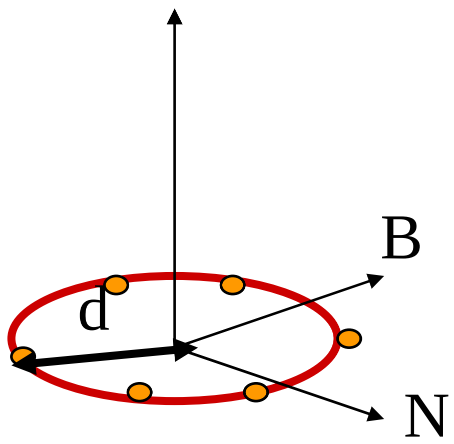
Place instances of the section regularly along the skeleton

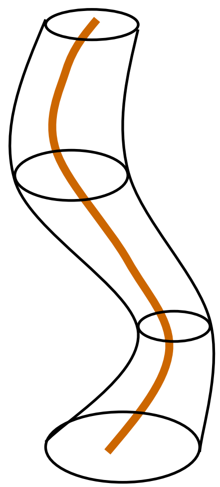
Does not work properly!



Extrusion

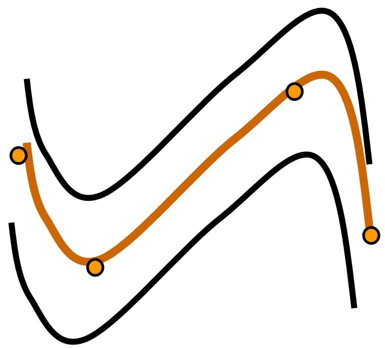
- Create offsets of the skeleton
  - Curves at fixed distance from skeleton, fixed angle / normal
- Adapt the offset distance using the profile



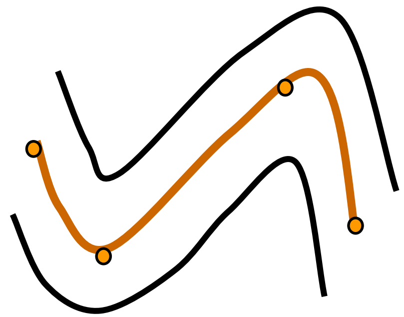


Extrusion

Issue : *Offsets are NOT translated curves*



Translated copies of the skeleton



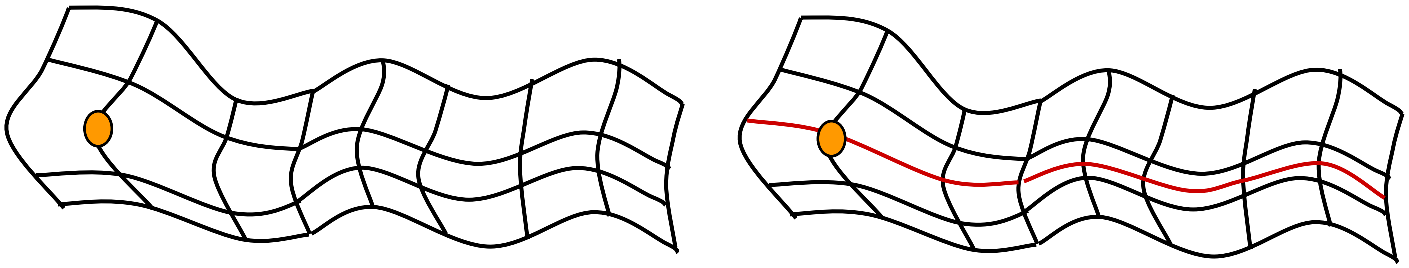
Offsets

- Note: offsets of splines curves are NOT spline curves

In practice, approximated using the same number of control points!

Step 2: Deform locally or Globally

OK for local deformation, but is locality controllable?

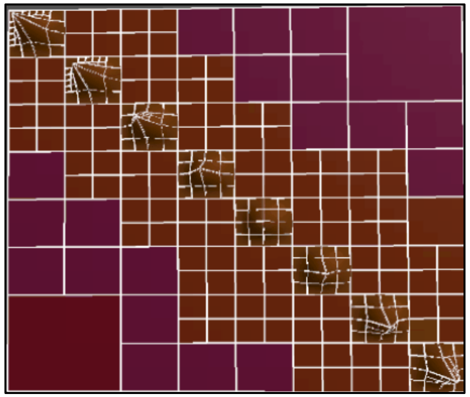


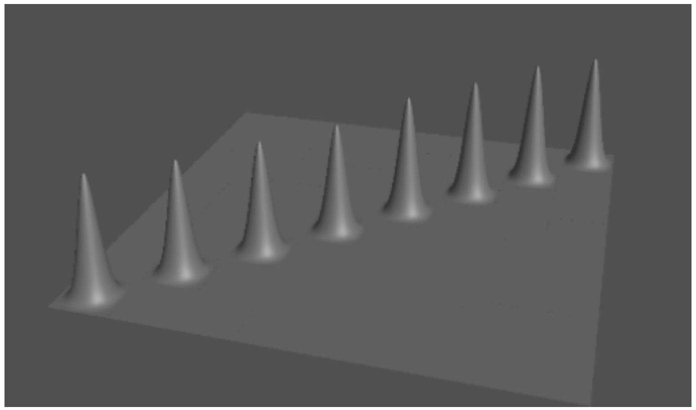
- Spline surfaces
  – Difficult to get details where needed!
  – Can we edit at a large scale once details have been added ?

Step 2: Deform locally or Globally
Issue: Control of locality

Hierarchical Spline Surfaces *[Forsey, Bartels SIGGRAPH 88]*

- Tree-structure of control-point grids
- Local coordinated for points : $P = G + O$,
  - $G = S_i(u_0,v_0)$ closest point on parent surface
  - O offset vector, expressed in the local frame of the parent

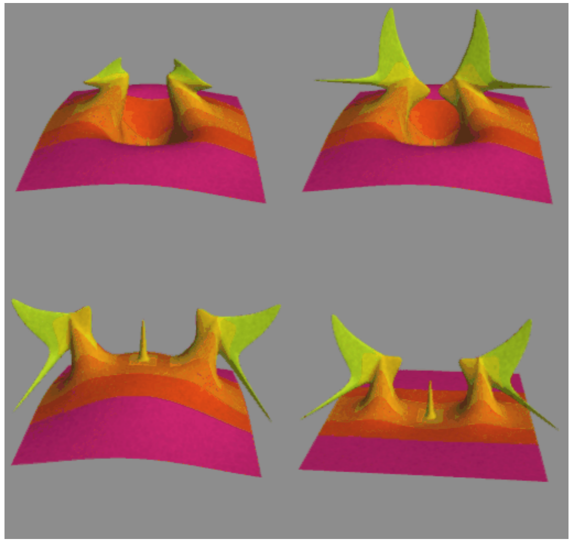




1. Model-based, local deformations
Issue: Control of locality

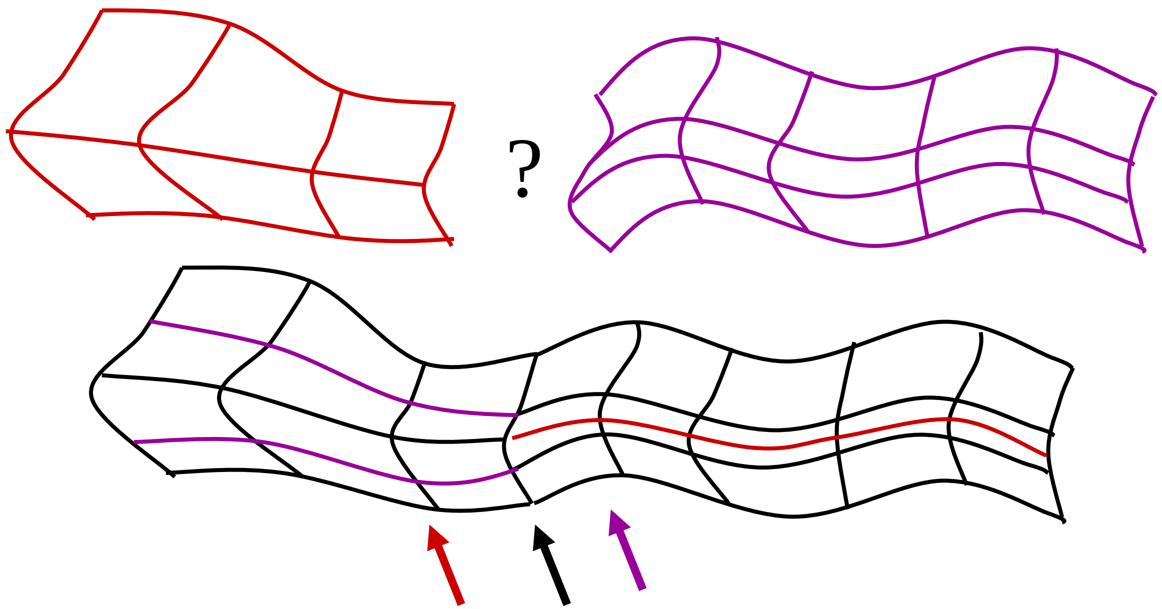
Hierarchical Spline Surfaces *[Forsey, Bartels SIGGRAPH 88]*

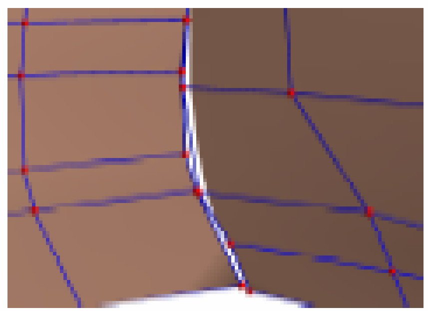
- Compact: 24 editable control points instead of 1225!
- Large scale deformations while keeping details!



Step 3: Assembly

- Fitting 2 surfaces : same number of control points





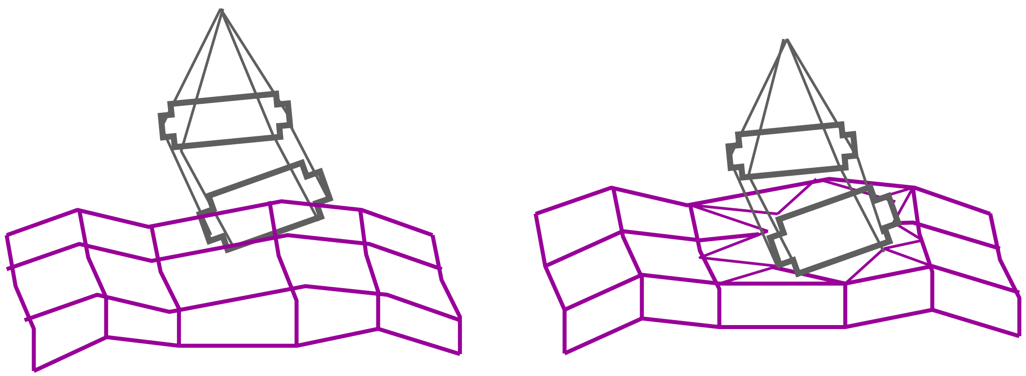
a. B-spline surfaces

Step 3: Assembly

Closed surfaces can be modeled

- Generalized cylinder: duplicate rows of control points
- Closed extremity: degenerate the spline surface!

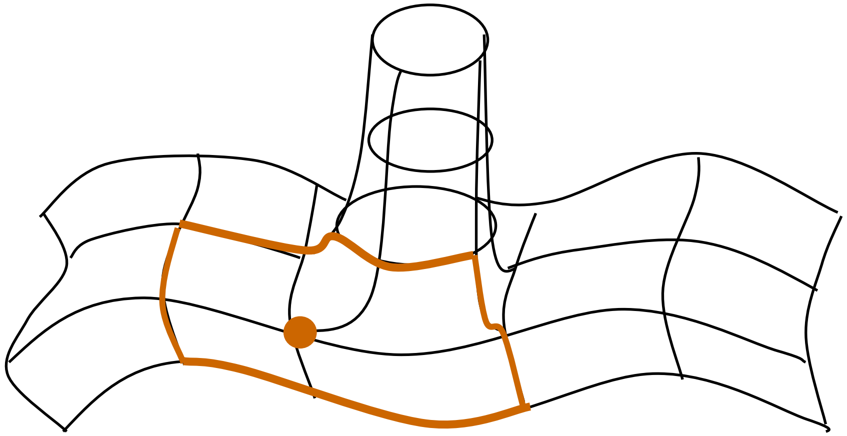
Can we fit them arbitrarily?



Step 3: Assembly

Branches ?

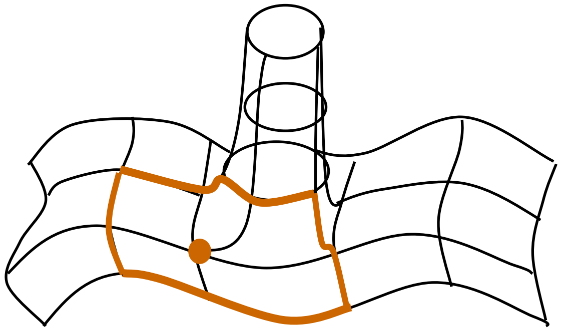
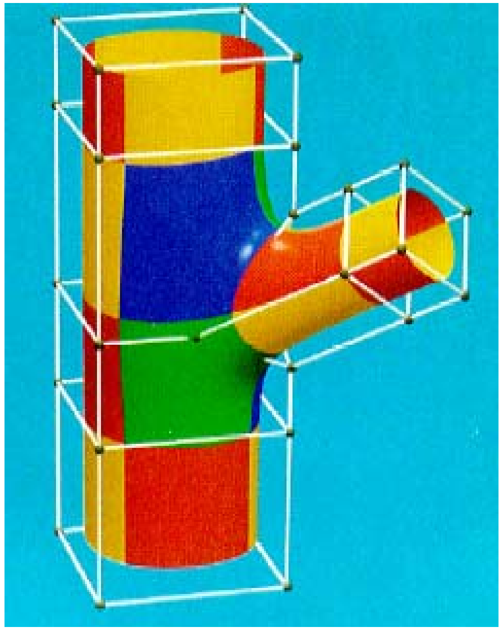
- 5 sided patch ?
- joint between 5 patches ?

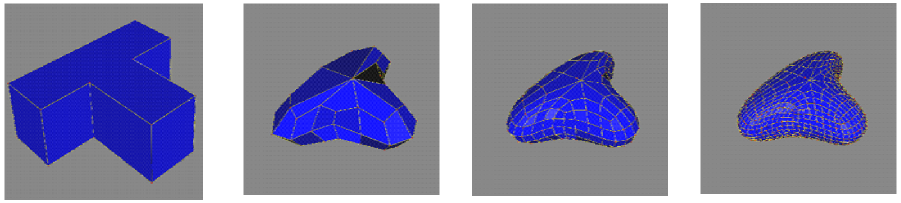


Advanced bibliography
Generalized B-spline Surfaces of Arbitrary Topology

[Charles Loop & Tony DeRose, SIGGRAPH 1990]

- n-sided generalization of Bézier surfaces: "Spatches"

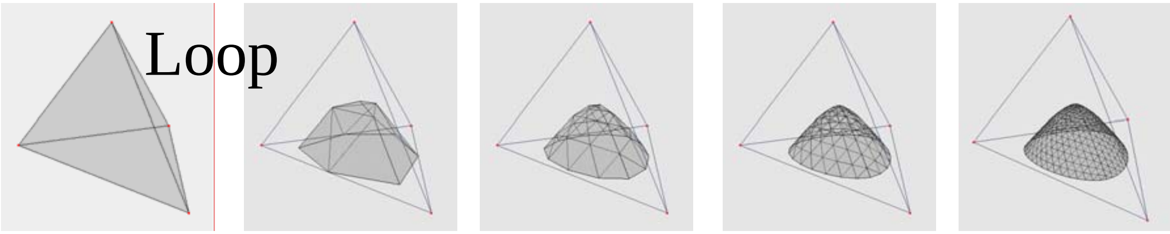



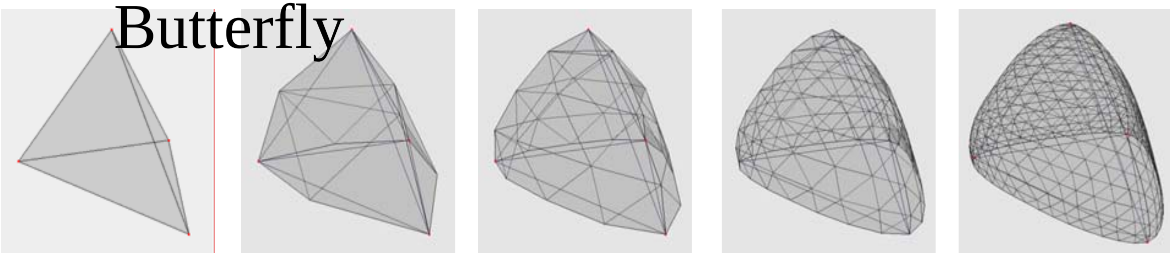
Subdivision Surfaces

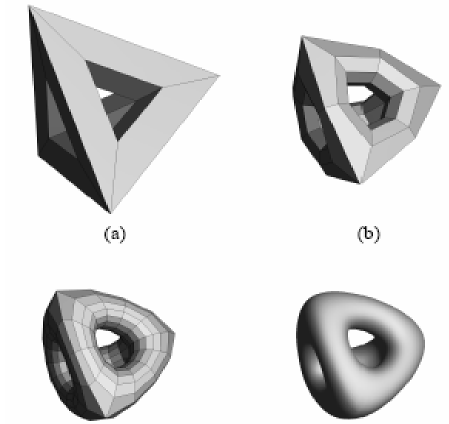
- Topology defined by the control polygon
- Progressive refinement (interpolation or approximation)

Loop



Butterfly

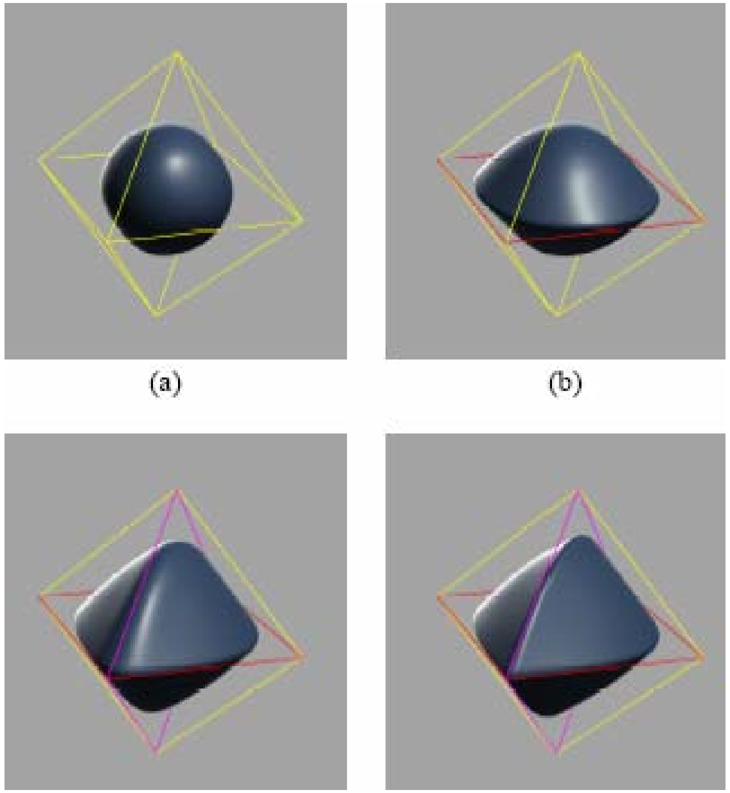


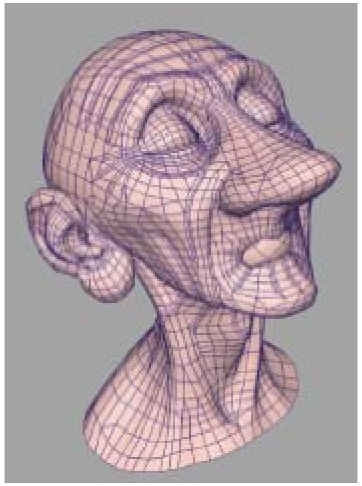


Catmull-Clark

Advanced bibliography
Subdivision Surfaces in Character Animation

[Tony DeRose, Michael Kass, Tien Truong, Siggraph 98]





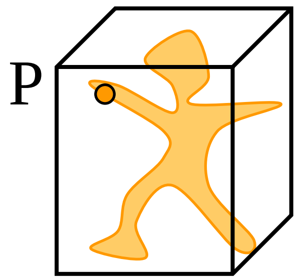
Keeping some sharp creases
where needed

Complement: another use of splines
Defining "Space deformations"

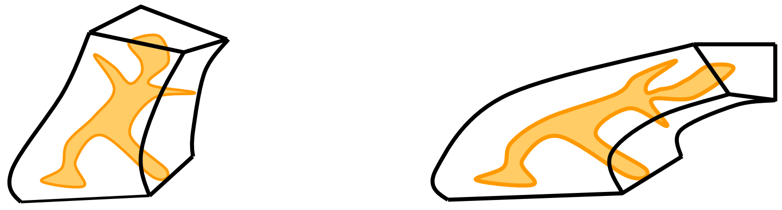
"Free form deformations" (FFDs)

1. Place the object in a Spline volume (3D grid of control points)

$$Q_{i,j,k}\,(u, v,w) = \sum B_i(u)\, B_j(v)\, B_k(w)\, P_{ijk}$$

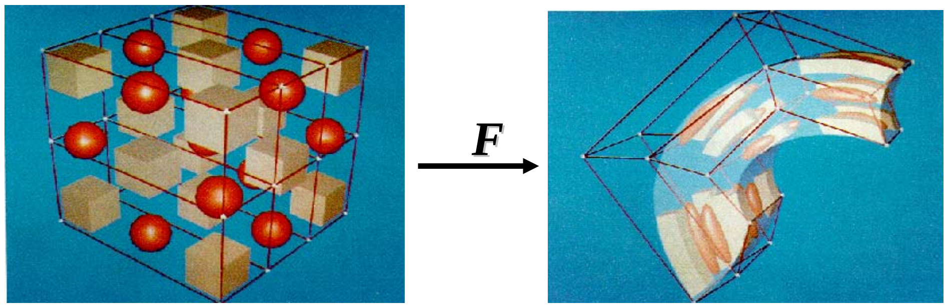


2. "Freeze" each vertex P to *(u,v,w)*
3. Move the volume's control points
4. Re-compute the object's vertices : $P = Q_{i,j,k}\,(u_0, v_0, w_0)$



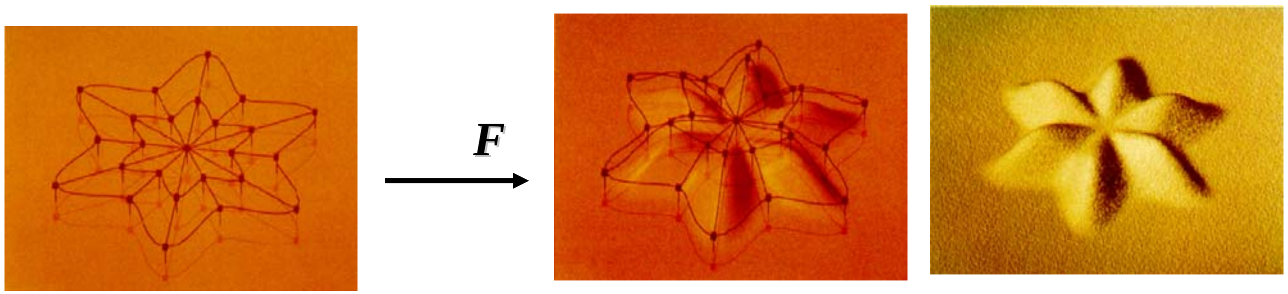
Complement: another use of splines
Defining "Space deformations"

"Free form deformations" FFDs



[Sederberg, Parry 1986]

- Lattice = Bézier volume
- Pb of locality



[Coquillart 1990]

Extended FFD