Modeling

- Need to usefully represent objects in the world
- Need to provide for easy interaction
  - manual modeling
    - user would like to “fiddle” until it is right (e.g. CAD)
    - user has an idea what an object is like
  - fitting to measurements
    - laser range finder data
- Support rendering/geometric computations

Modeling tools

- Polygon meshes
- Fitting curves to points (from data)
- Fitting curves to points (user interaction)
- Generating shapes with sweeps
- Constructive solid geometry

Polygon Meshes

- Common, straightforward, often built in (e.g. torus mesh)
- Ready to render (many of the representations discussed soon are often be reduced to polygon meshes for rendering)
- Problems
  - Awkward to provide user editing
  - The number of polygons can be very large
    - Some kind of adaptive process makes sense
    - More polygons at high curvature points
    - More polygons where the object is larger
    - extra care then needs to be taken to avoid temporal aliasing

Explicit curve representation

- Usual representation learned first
- Generally less useful in graphics, but know the term
- Explicit curve is a function of one variable. Examples
  - line, y=m*x + b
  - circle (need to glue two together) y = ±sqrt(r*r - x*x)
- Explicit surface is a function of two variables. Examples
  - plane z = m*x+ n*y + b

Implicit representation

- Also less useful for this section, but again, know the term
- An implicit curve is given by the vanishing of some functions
  - circle on the plane, x*x+y*y-r*r=0
  - twisted cubic in space, x*y-z=0, x*z-y*y=0, x*x-y=0
- An implicit surface is given by the vanishing of some functions
  - sphere in space x*x+y*y+z*z-r*r=0
  - plane a x+ b y + c z+d=0

Parametric representation

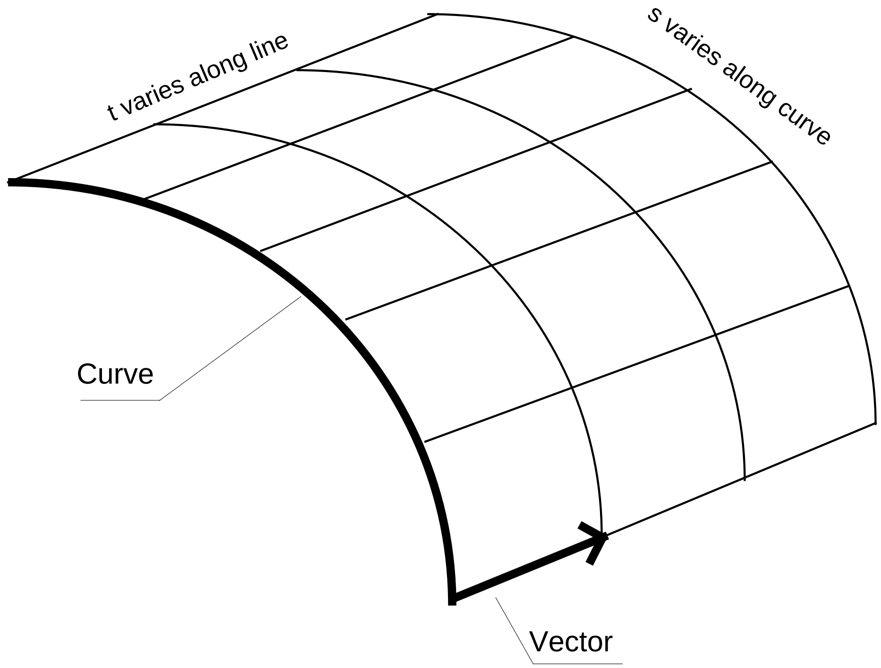
- A parametric **curve** is given as a function of one parameter. Examples:
  - circle as (cos t, sin t)
  - twisted cubic as (t, t*t, t*t*t)
- A parametric **surface** is given as a function of two parameters. Examples:
  - sphere as (cos s cos t, sin s cos t, sin t)
- Advantage - easy to compute normal, easy to render, easy to put patches together, ranges can be easy (e.g. half circle)
- Disadvantage - ray tracing can be hard

Generating Surfaces

- We can construct surfaces from curves in a variety of user intuitive ways
- Extruded surfaces
- Generalized cones
- Surfaces of revolution
- Sweeping (generalized cylinders)

Extruded surfaces

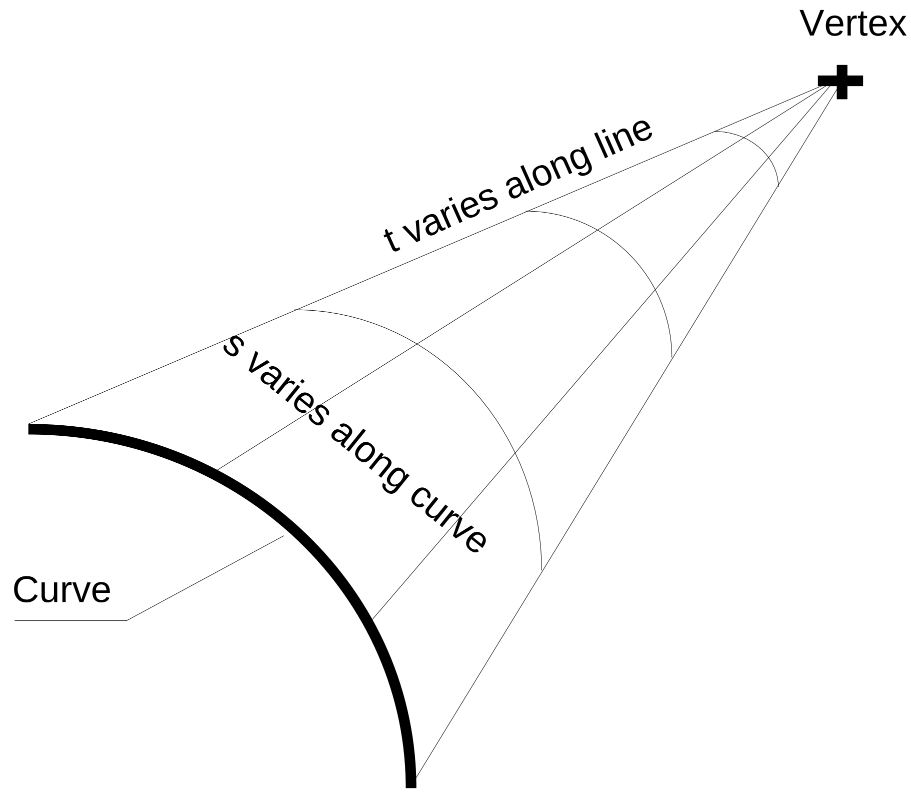
- Geometrical model - Pasta machine
- Take curve and “extrude” surface along vector
- Many human artifacts have this form - rolled steel, etc.



$$(x(s,t), y(s,t), z(s,t)) = (x_c(s), y_c(s), z_c(s)) + t(v_0, v_1, v_2)$$

Cones

- From every point on a curve, construct a line segment through a single fixed point in space - the vertex
- Curve can be space or plane curve, but shouldn't pass through the vertex

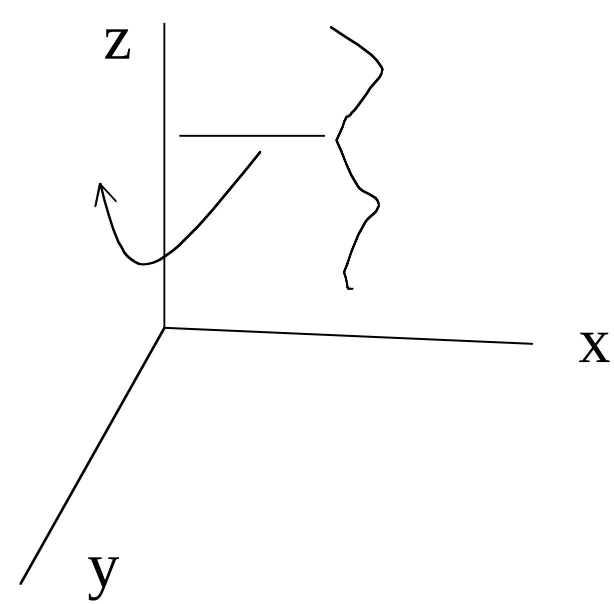


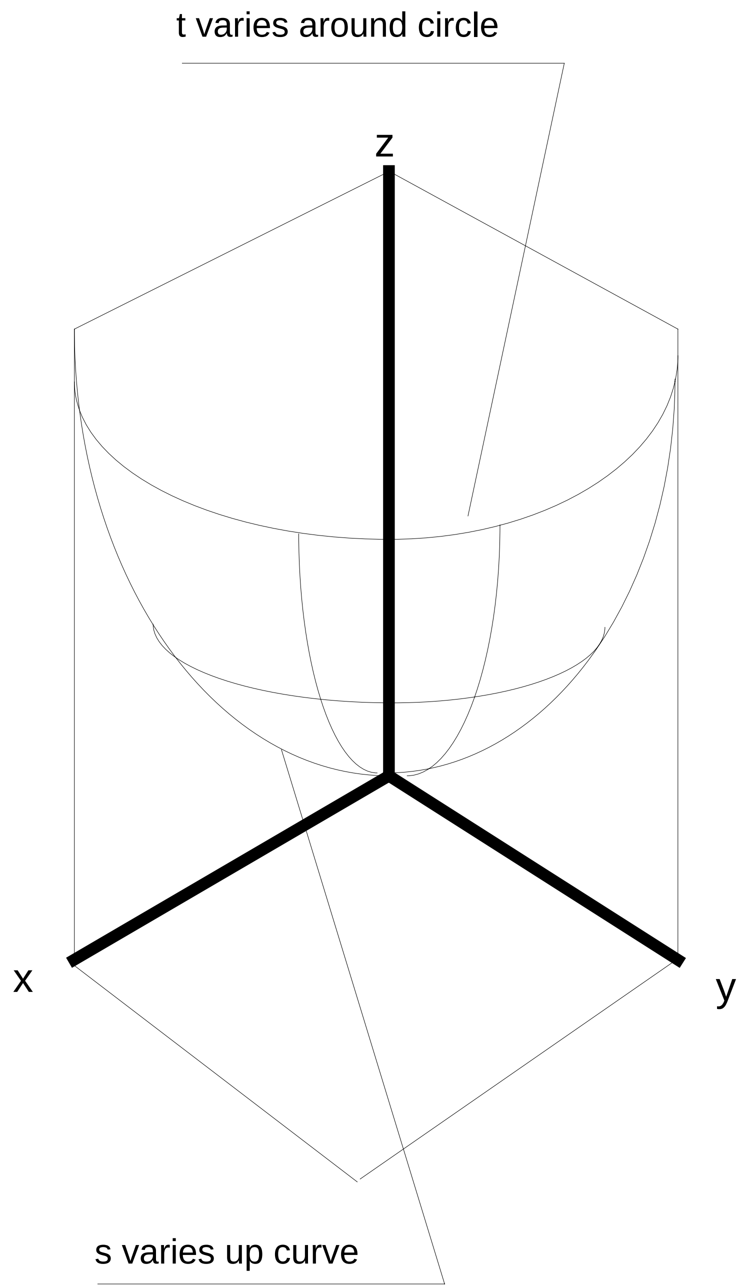
Surfaces of revolution - 1

- Plane curve + axis
- "spin" plane curve around axis to get surface
- Choice of plane is arbitrary, choice of axis affects surface
- In the example to the right, curve is on x-z plane, axis is z axis. (think of $x_c(s)$ as a radius)

$$(x(s,t), y(s,t), z(s,t)) =$$

$$(x_c(s)\cos(t), x_c(s)\sin(t), z_c(s))$$





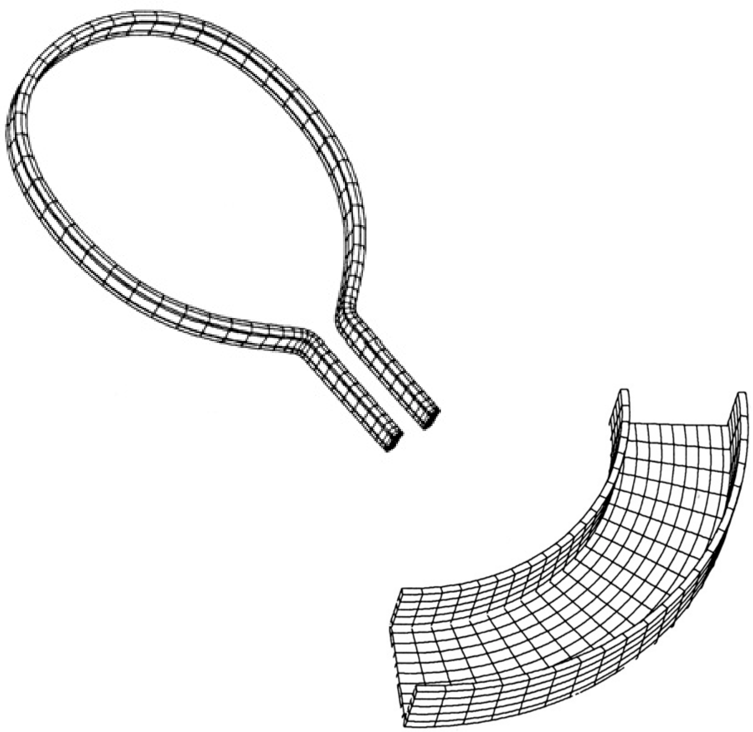
Surfaces of revolution -2

Many artifacts are SOR’s, as they’re easy to make on a lathe.

Controlling is quite easy - concentrate on the cross section.

Axis crossing cross-section leads to ugly geometry.

Sweeps/Generalized Cylinders



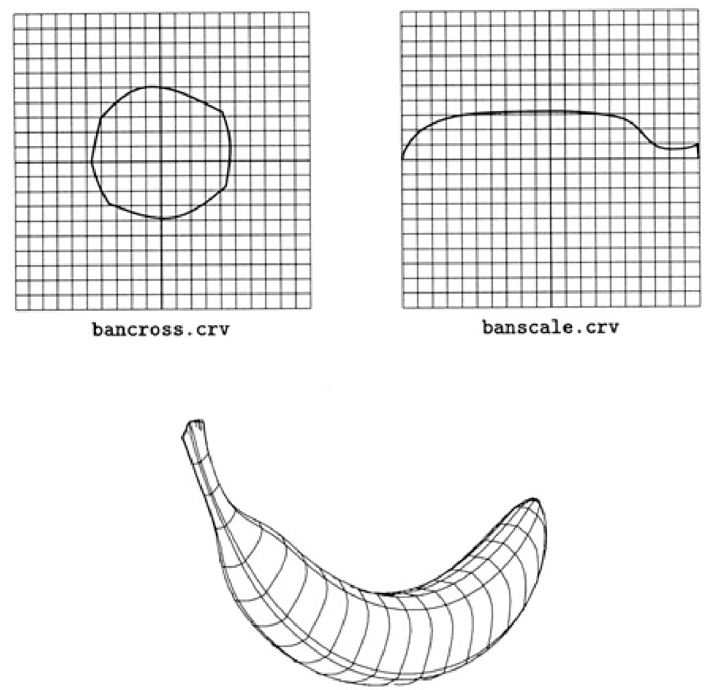
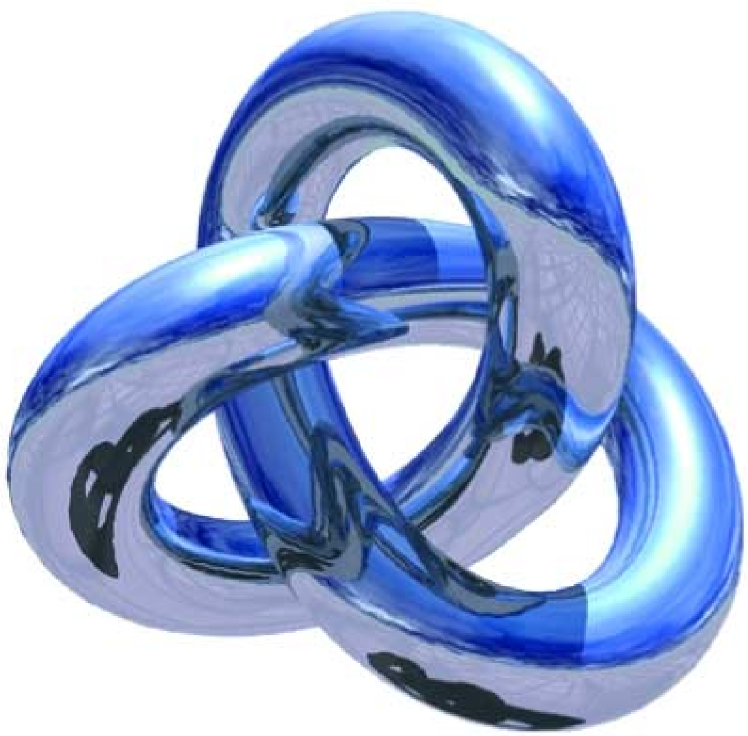


Figure 3.8: Banana example. A banana is represented by an affine transformation surface. The cross section is scaled, translated along z from -1 to 1, and rotated around the y axis. □

[Synder 92, via CMU course page]

Sweeps/Generalized Cylinders



MetaCreations, via CMU course page

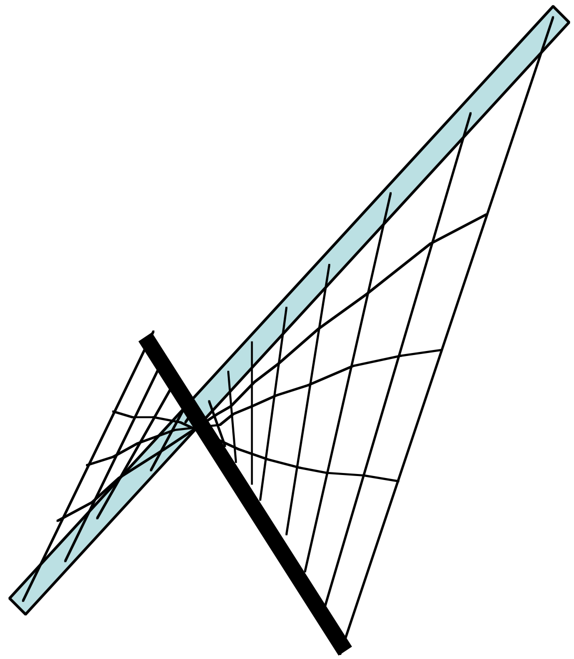
Ruled surfaces -1

- Popular, because it's easy to build a curved surface out of straight segments - e.g. pavilions, etc.
- Take two space curves, and join corresponding points - same s - with line segment.
- Even if space curves are lines, the surface is usually curved.

$$(x(s,t), y(s,t), z(s,t)) =$$
$$(1-t)(x_1(s), y_1(s), z_1(s)) +$$
$$t(x_2(s), y_2(s), z_2(s))$$

Ruled Surfaces - 2

Easy to explain,
hard to draw!

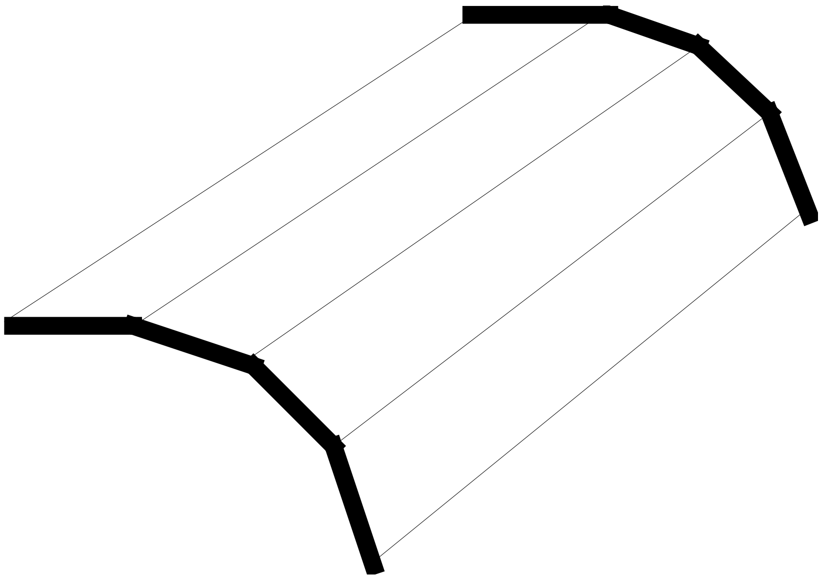


Normals

- Normal is cross product of tangent in t direction and s direction.
- Cylinder: normal is cross-product of curve tangent and direction vector
- Surface of revolution: take curve normal and spin round axis

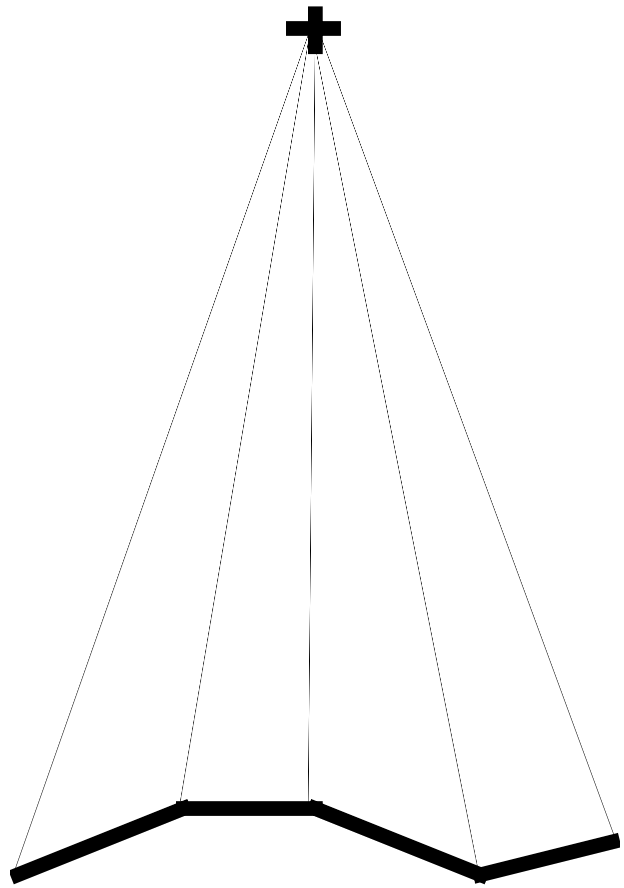
Rendering

- Cylinders: small steps along curve, straight segments along t generate polygons; exact normal is known.



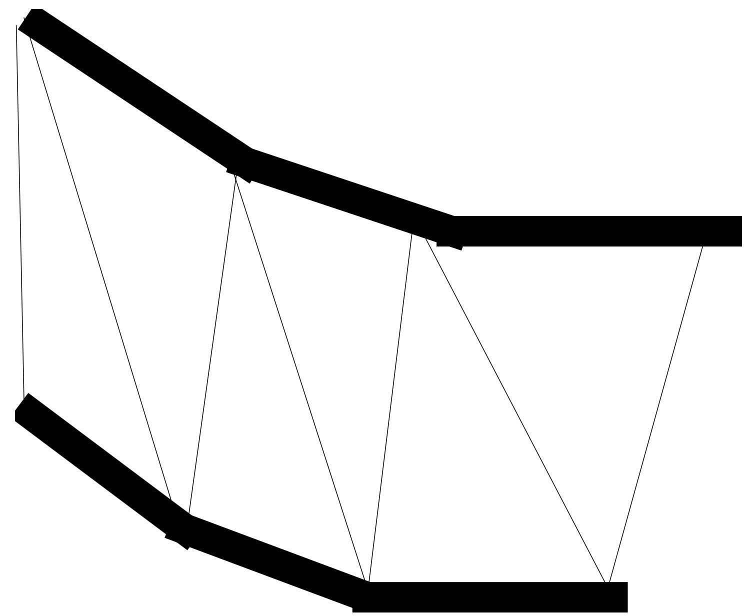
Rendering

- Cone: small steps in s generate straight edges, join with vertex to get triangles, normals known exactly except at vertex.



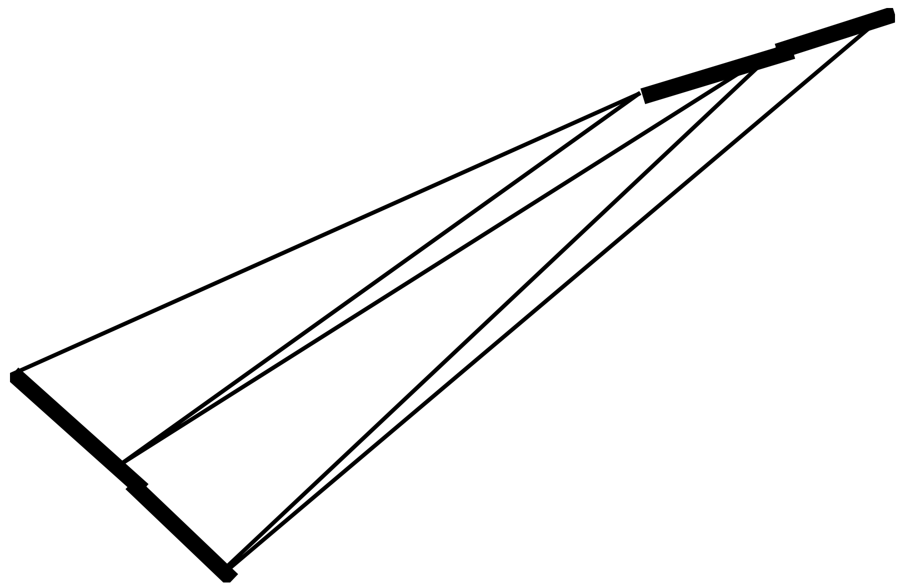
Rendering

- Surface of revolution: small steps in s generate strips, small steps in t along the strip generate edges; join up to form triangles. Normals known exactly.



Rendering

- Ruled surface: steps in s generate polygons, join opposite sides to make triangles - otherwise “non planar polygons” result. Normals known exactly.



Specifying Curves from Points

- Want to modulate curves via “control” points.
- Strategy depends on application. Possibilities:
  - Force a polynomial of degree N-1 through N points (Lagrange interpolate)
  - Specify a combination of “anchor” points and derivatives (Hermite interpolate)
  - Other “blends” (Bezier, B-splines)--more useful than Lagrange/Hermite

Specifying Curves from Points-II

- Issues:
  - Continuity of curve and derivatives (geometric, parametric)
  - Local versus global control
  - Polynomials verses other forms
  - Higher polynomial degree versus stitching lower order polynomials together
  - Polynomial degree (usually 3--fewer is not flexible enough, and higher gives hard to control wiggles).

Parametric vs Geometric Continuity

- **Parametric continuity**:
  - The curve and derivatives up to k are continuous *as a function of parameter value*
  - C^k
  - Useful for (for example) animation
- **Geometric continuity**
  - curve, derivatives up to k'th are the same for equivalent parameter values
  - D^k
  - i.e. there exists a reparametrisation that would achieve parametric continuity
  - Useful, because we often don't require parametric continuity,