Outcomes

- Understand the basic concepts in modeling curves
- Understand the basic concepts in modeling surfaces
- Understand the ideas behind the use of sweeping curves to approximate objects

Code Examples

cylinder.cpp

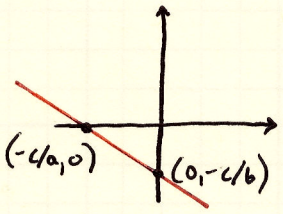
table.cpp

Curves

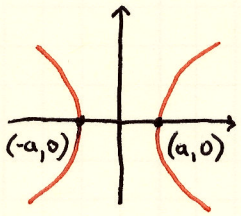
- Curves in OpenGL, unlike straight segments, cannot be drawn exactly, and must be approximated.
- Curves combined with straight line segments form one dimensional objects.
- Specifying Plane Curves
  - Plane curves are those which lie on a plane in $\mathbb{R}^2$, and can be specified either implicitly or parametrically
  - Implicit
    - Plane curves are specified implicitly by the following equation of a curve C

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

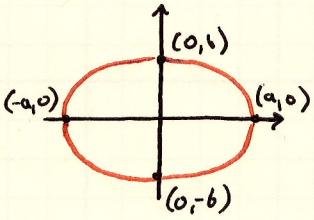
      - If the points of C are those whose coordinates satisfy the equation, then it is said to be the implicit equation of C.
      - C is then the graph of this equation.
    - Examples



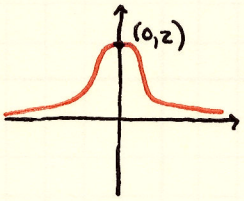
Straight Line: $ax + by + c = 0$



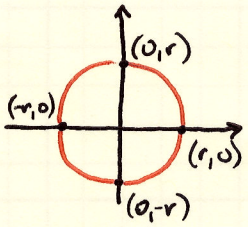
Hyperbola: $\frac{x^2}{a^2} - \frac{y^2}{b^2} = 1$



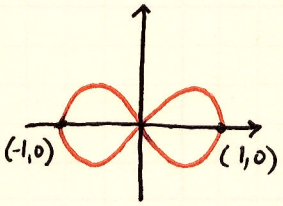
Ellipse: $\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1$



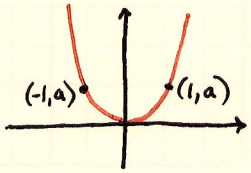
Witch of Agnesi: $y(x^2+4) = 8$



Circle: $x^2 + y^2 = r^2$



Lemniscate of Bernoulli: $(x^2+y^2)^2 = x^2 - y^2$



Parabola: $y = ax^2$

- An implicit equation is often written in the form $F(x,y) = G(x,y)$ with the RHS not necessarily equal to 0.

- Parametric

  - A plane curve c is specified parametrically by the two equations:

    $$x = f(t), \quad y = g(t), \quad \text{where } t \in T$$

  - T is a given set called the parameter space, or parameter domain.
    - an interval in $\mathbb{R}$
    - Typically: $[-\pi, \pi]$, $[0, \infty]$, $(-\infty, \infty)$

  - Examples
    - Straight line: $x = t, \ y = -\frac{a}{b}t - \frac{c}{b}, \ t \in (-\infty, \infty), \ b \neq 0$
    - Ellipse: $x = a\cos t, \ y = b \sin t, \ t \in [-\pi, \pi]$
    - Circle: $x = r\cos t, \ y = r \sin t, \ t \in [-\pi, \pi]$
    - Parabola: $x = t, \ y = at^2, \ t \in (-\infty, \infty)$
    - Hyperbola: $x = a \sec t, \ y = b \tan t, \ t \in [-\pi, -\pi/2) \cup (-\pi/2, \pi/2) \cup (\pi/2, \pi]$
    - ~~Ellipse (repeat)~~:

  Ex: Find an implicit equation for: $x = \cos t, \ y = \sin t, \ t \in [-\pi, \pi]$

  - Implicit definition provides a Boolean check for points on a curve, while the parametric definition allows for generating points on a curve.

- Specifying Space Curves
  - Curves in $\mathbb{R}^3$ are called space curves.
  - Implicit Definition:

    $$F(x,y,z) = 0 \qquad G(x,y,z) = 0$$

    - Two equations as $\mathbb{R}^3$ is unconstrained and thus needs an additional equation to reduce the resulting object's dimension by one.
      - Example:

        $x^2 + y^2 + z^2 - 1 = 0$, defines a sphere

        $\left.\begin{array}{l} x^2+y^2+z^2-1=0 \\ x+y+z-1=0 \end{array}\right]$ Together define a circle in $\mathbb{R}^3$

      - A Basic plane curve in $\mathbb{R}^3$ is define by

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

- Parametric

$$x = f(t),\ y = g(t),\ z = h(t),\ t \in T$$

- For a helix:

$$x = r\cos t,\ y = r\sin t,\ z = t,\ t \in \mathbb{R}$$

- For a straight line:

$$x = a_1 t + b_1,\ y = a_2 t + b_2,\ z = a_3 t + b_3,\ t \in \mathbb{R}$$

where $a_i, b_i,\ 1 \le i \le 3$

Ex: Sketch on paper the curve, called a conical helix, specified by:

$x = t\cos t$
$y = t\sin t$
$z = t$

$t \in (-\infty, \infty)$

- Drawing Curves

- Here we describe the ~~ground~~ procedure for drawing a general space curve c given parametrically by:

$$x = f(t),\ y = g(t),\ z = h(t),\ t \in T$$

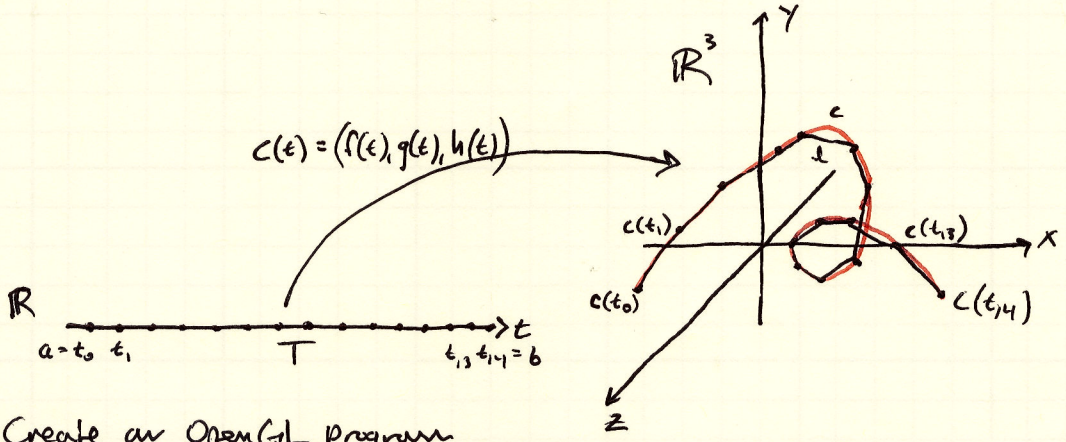
- Assume the parameter domain T is $[a,b]$, a closed interval, the point $(f(t), g(t), h(t))$ on c is denoted $c(t)$

- Imagine:

  - T as being a real line
  - c lies on a "separate" 3-space

- $c(t)$ can then be thought of as a mapping from the former to the latter.

  - A sample: $a = t_0 < t_1 < \cdots < t_n = b$ of $n+1$ points from $[a,b]$ is called a sample grid. It maps to a sample $c(t_0), c(t_1), \ldots, c(t_n)$ of $n+1$ points of c, called the mapped sample.



EX: Create an OpenGL program which draws the twisted cubic specified by

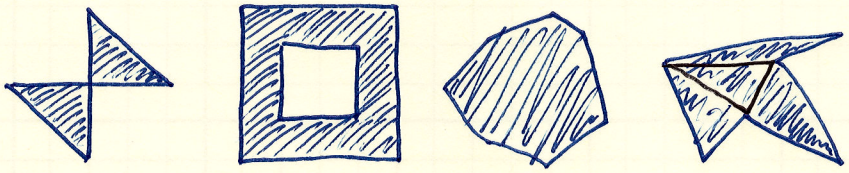
$$x = t,\ y = t^2,\ z = t^3$$ drawn near the origin

Surfaces

- 2D objects are composed of surfaces. Similar to 1D objects, those aspects of 2D objects which cannot be exactly represented by triangles, must be approximated.

- **Polygons**

  - **Simple Planar Polygon** is a polygon which lies on a plane, whose boundary consists of a single component which is a non-self-intersecting loop of straight line segments.



Self-intersecting boundary | Multiple Boundary components | Convex | Non-convex w/ Triangulation indicated

  - Regardless of convexity it is recommended that all polygons be first triangulated and then drawn with GL_TRIANGLE* primatives.

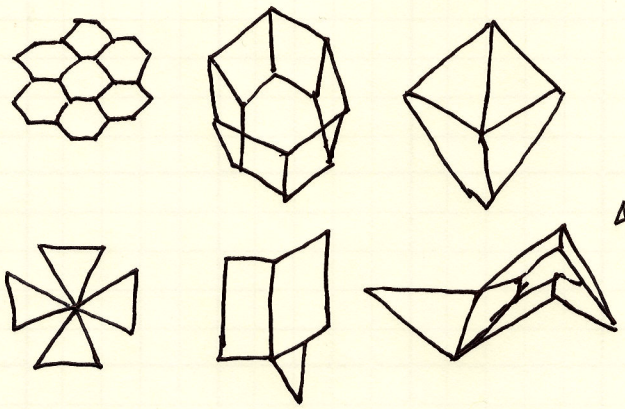
- **Meshes**

  - A **polygon mesh** (or simply **mesh** or **polyhedral surface**) is a union of polygons satisfying the following two conditions:

    1.) Any two polygons in the union are disjoint or intersect in a vertex of both or intersect in an edge of both

    2.) The neighborhood around each is "sheet" like

- **Examples**



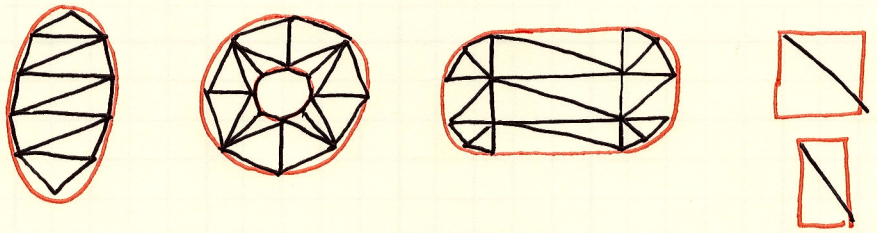
Ex: Is this figure a mesh?

- Condition two is for a mesh to be a so-called topological manifold, which guarantees a certain respectability to shapes that a mesh can take

- The convex polygons comprising a mesh are its faces
- The boundary of a mesh consist of those edges with a polygon on only one side.
- A mesh with no boundary is called a closed mesh and bounds a solid object.
- It is best if the faces are all triangles, which is called a triangular mesh.
- The only 2D objects OpenGL can draw exactly are meshes and unions of meshes. Others must be approximated.

- Planar Surfaces

- A generalization of a polygon. This is simply a surface which lies on a plane and which has no further restrictions.



- The drawing of a planar surface can be reduced to the drawing of an approximating mesh, as follows:

1.) Use the curve drawing approach to construct polyline approximations of curved edges along the surface's boundary. Combine this with existing straight lines to form the entire boundary

2.) Triangulate the approximating polygon(s)

Ex: Create an OpenGL program which draws a rounded rectangle.

- General Surfaces

- Surfaces which may be neither planar, nor a union of polygons may also be drawn by approximation by triangular meshes as well.
- First Specification:
  - Implicit: $F(x,y,z) = 0$
    - Such that the points of surface S are those whose coordinates (x,y,z) satisfy the equation
  - Parametric: $x = f(u,v), y = g(u,v), z = h(u,v)$, where $(u,v) \in W$
    - Parameter space W is a subset of $\mathbb{R}^2$
    - Two parameters, $u + v$, since it has 2 dimensions

- Drawing General Surfaces

- Similar to approximating a curve, but rather than using straight line segments approximating sub-arcs, triangles are used to approximate small patches of the surface and a triangular mesh for the entire surface.
- A surface S is specified as

$$x = f(u,v),\ y = g(u,v),\ z = h(u,v),\ (u,v) \in W$$

where W is the plane rectangle $[a,b] \times [c,d]$, then the above equations shape W into S in xyz-space

- Sample W at the $(p+1)(q+1)$ points

$$(u_i, v_j),\ 0 \le i \le p,\ 0 \le j \le q$$

of rectangular sample grid where

$$a = u_0 < u_1 < \dots < u_p = b$$

$$c = v_0 < v_1 < \dots < v_q = d$$

the mapped sample

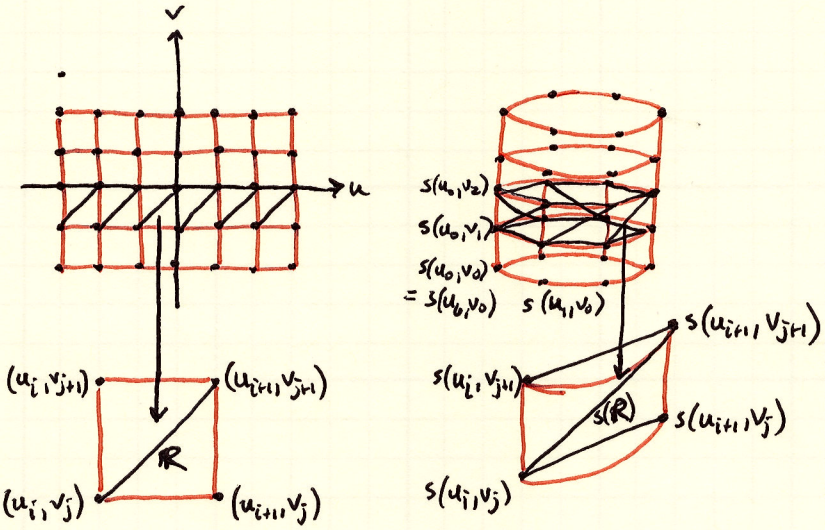
$$S(u_i, v_j),\ 0 \le i \le p,\ 0 \le j \le q$$

of $(p+1)(q+1)$ points of S are used as vertices of a triangular mesh approximation of S. This mesh consists of the $2pq$ triangular faces

$$\Delta S(u_i, v_{j+1}) S(u_i, v_j) S(u_{i+1}, v_{j+1}) \text{ and } \Delta S(u_{i+1}, v_{j+1}) S(u_i, v_j) S(u_{i+1}, v_j),\ 0 \le i \le p,\ 0 \le j \le q-1$$

- Example: The cylinder defined by

$$x = \cos u,\ y = \sin u,\ z = v,\ (u,v) \in [-\pi,\pi] \times [-1,1]$$



- Cylinder: Cylinder.cpp shows a triangular mesh approximation of a circular cylinder given by

$$x = f(u,v) = \cos u,\ y = g(u,v) = \sin u,\ z = h(u,v) = v$$

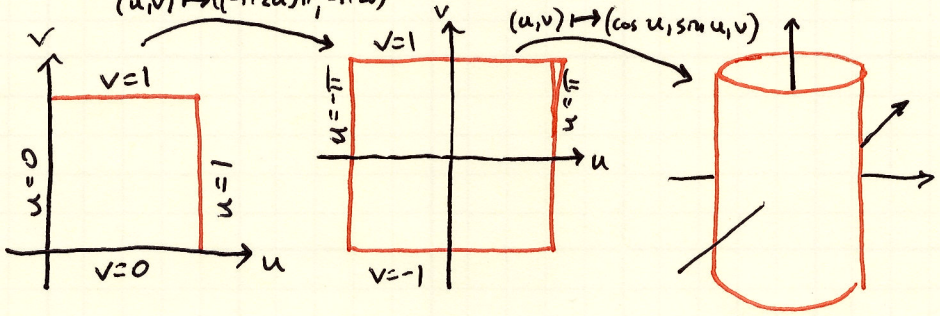
- The cylinder is constructed using the general method above but constructs the parameter space from $[-\pi,\pi] \times [-1,1]$ to $[0,1] \times [0,1]$ using the following functions:

```
float f(int i, int j) {
  return (cos((-1 + 2 * (float) i/p) * PI));
}
float g(int i, int j) {
  return (sin(-1 + 2 * (float) i/p) * PI));
}
float h(int i, int j) {
  return (-1 + 2 * (float) j/p);
}
```

f applies mapping: $u \mapsto (-1+2u)\pi$ to scale $[0,1]$ to $[-\pi,\pi]$, then applies cos

g applies mapping: $u \mapsto (-1+2u)\pi$ then applies sin

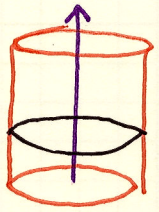
h applies mapping: $v \mapsto -1+2v$ to scale $[0,1]$ to $[-1,1]$



- Cylinder.cpp serves as a template for constructing any surface from a set of parametric equations.

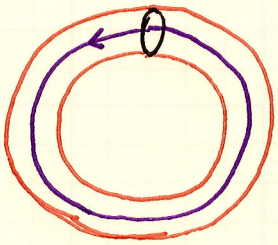
° Swept Surfaces

- A powerful design method to create surfaces is by sweeping a curve
- Examples
  - ° Cylinder: sweep a circle along a line perpendicular to the center of the circle:



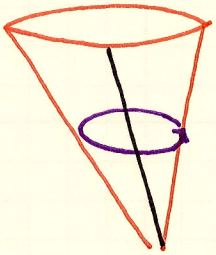
- The curve that sweeps the surface is called its profile curve, or profile
- The path followed by the profile is trajectory
- The surface is called a swept surface

  - ° Torus: a circular profile traveling along a circular trajectory



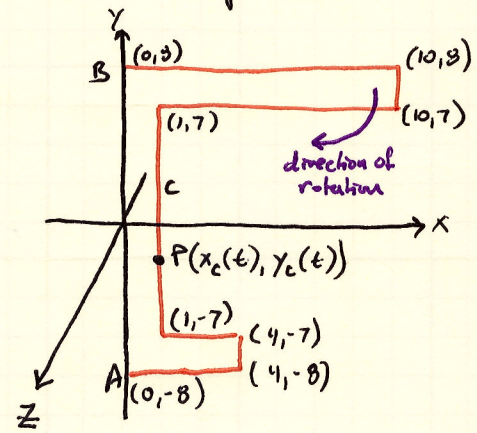
- When the trajectory is a circle, the swept surface is called a surface of revolution

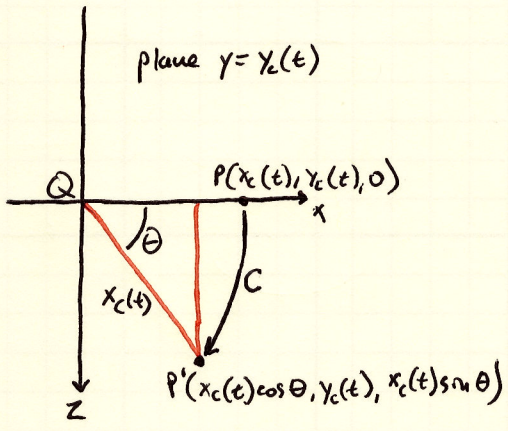
  - ° Cone: A surface of revolution swept by a straight ~~line~~ segment profile in a circular trajectory about the cone's axis.



- When the trajectory is a straight segment, the resulting surface is said to be an extrusion, or extruded surface, obtained by extruding the profile.
- In such a case the profile is called the base curve

- Table: Created by sweeping curve C about the y-axis. C is a polyline consisting of 7 segments starting at point A and ending at point B





- C is first parameterized using the length t along c measured from A to P as the parameter value for a point P on C
- The x-coordinate, $x_c(t)$, with parameter t is as follows:

$$x_c(t) = \begin{cases} t, & 0 \le t \le 4 \\ 4, & 4 \le t \le 5 \\ 9-t, & 5 \le t \le 8 \\ 1, & 8 \le t \le 22 \\ t-21, & 22 \le t \le 31 \\ 10, & 31 \le t \le 32 \\ 42-t, & 32 \le t \le 42 \end{cases}$$

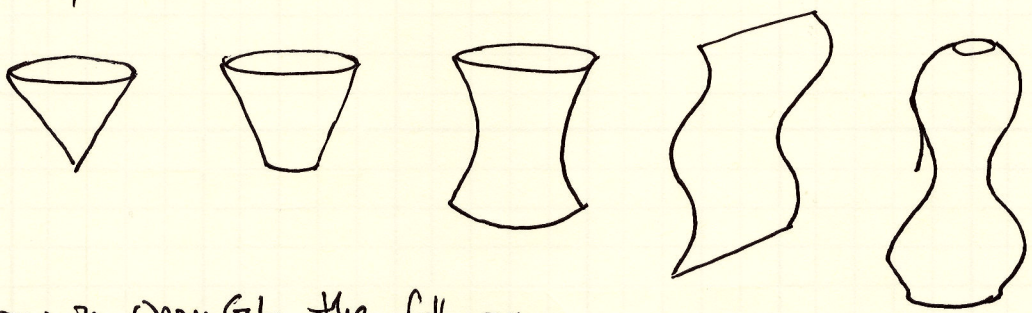
- The y-coordinate, $y_c(t)$, with parameter t is as follows:

$$y_c(t) \begin{cases} -8, & 0 \le t \le 4 \\ t-12, & 4 \le t \le 5 \\ -7, & 5 \le t \le 8 \\ t-15, & 8 \le t \le 22 \\ 7, & 22 \le t \le 31 \\ t-24, & 31 \le t \le 32 \\ 8, & 32 \le t \le 42 \end{cases}$$

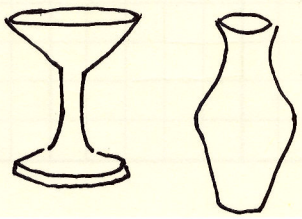
- On the xy-plane, a point P with coordinates $(x_c(t), y_c(t), 0)$ rotates by an angle of θ on a circle on the plane $y = y_c(t)$, centered at $Q = (0, y_c(t), 0)$ to a new point P'.
  - Here C is the part of the circular trajectory of P and since $|QP'| = |QP| = x_c(t)$, $P' = (x_c(t)\cos\theta, y_c(t), x_c(t)\sin\theta)$
- The parametric equations for the table are:

$$X = x_c(t)\cos\theta,\ Y = y_c(t),\ Z = x_c(t)\sin\theta,\ 0 \le t \le 42,\ -\pi \le \theta \le \pi$$

Ex: Modify cylinder.cpp to draw wireframe surfaces resembling:



Ex: Draw in OpenGL the following



using swept surfaces.

For Next Time

- Read Ch. 10 Sections 2.8 - 5
- Review Prior Lectures
- Continue HW 2
- Come to Class
-

Additional Notes