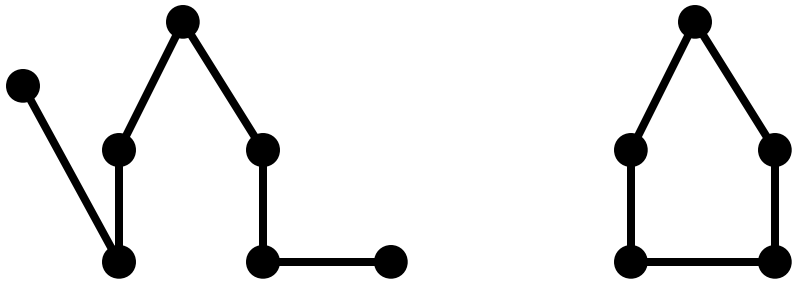
Describing Shape

2D Object Definition (1/3)

Lines and Polylines

- *Polylines*: lines drawn between ordered points

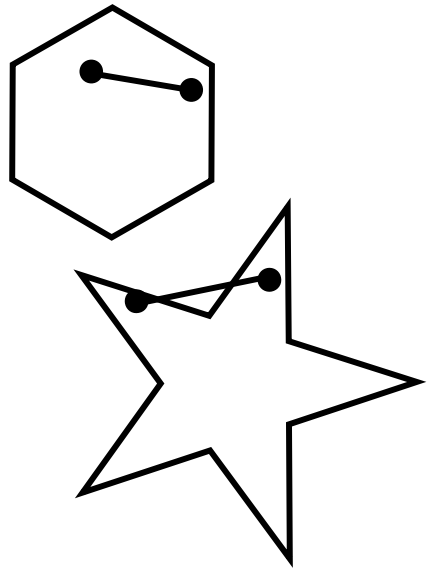


- Same first and last point make *closed polyline* or *polygon*
- If it does not intersect itself, called *simple polygon*

Convex vs. Concave Polygons

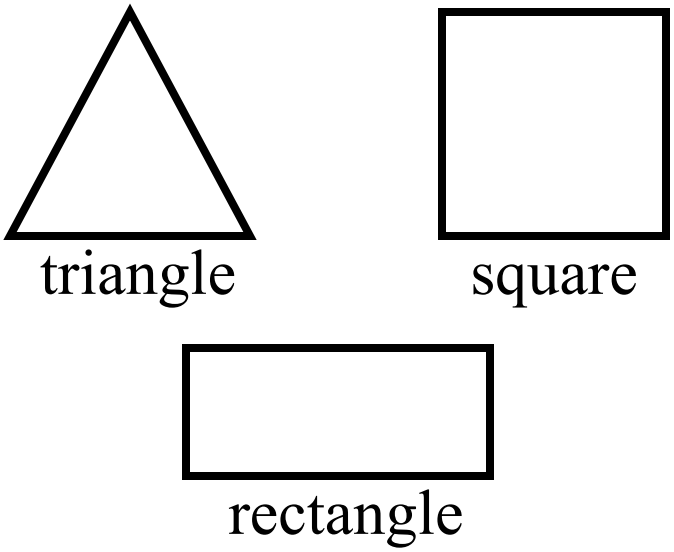
Convex: For every pair of points in the polygon, the line between them is fully contained in the polygon.

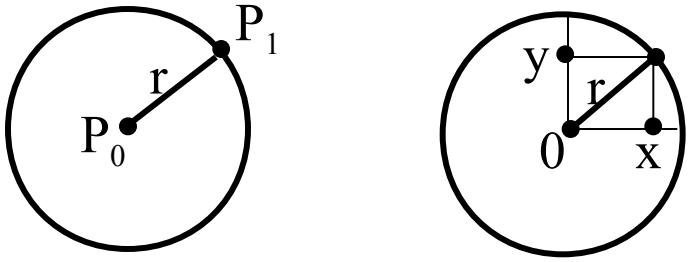
Concave: Not convex: some two points in the polygon are joined by a line not fully contained in the polygon.



2D Object Definition (2/3)

Special polygons

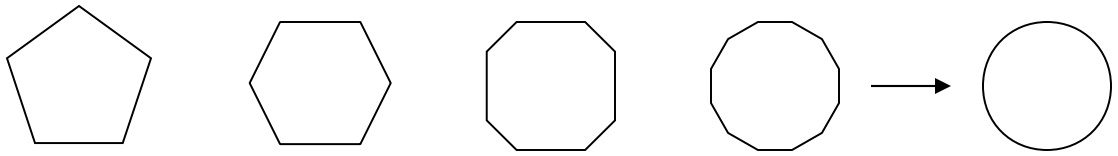




2D Object Definition (3/3)

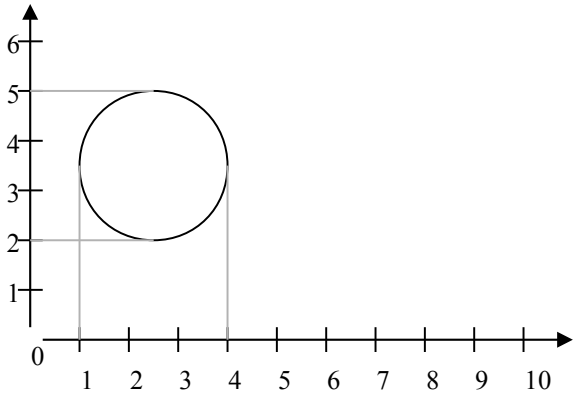
Circle as polygon

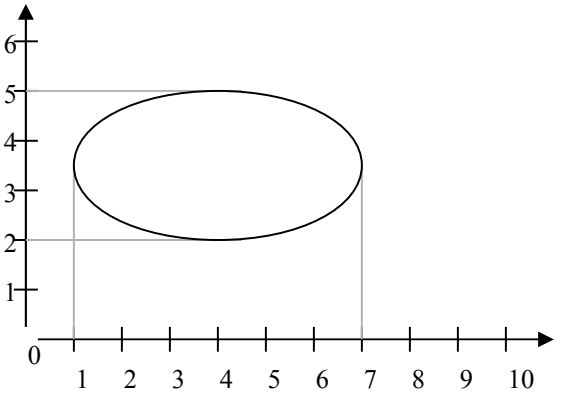
- A circle can be approximated by a polygon with many sides (>15)



(Aligned) Ellipses

A circle scaled along the x or y axis





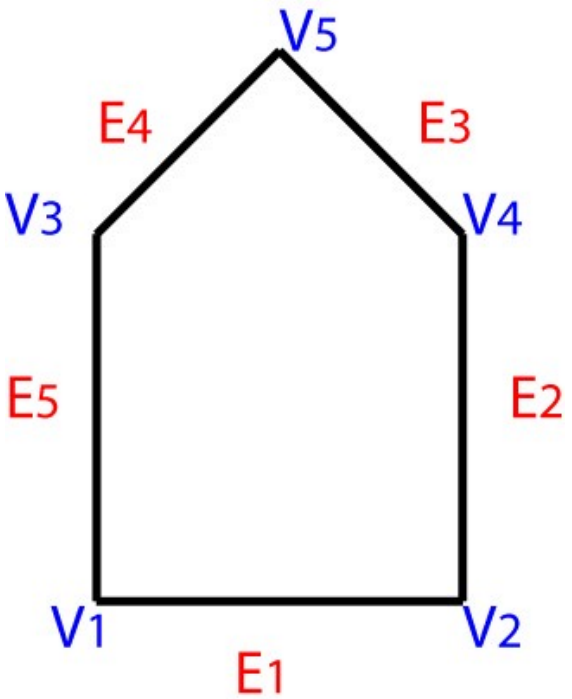
Example: height, on *y*-axis, remains 3, while length, on *x*-axis, changes from 3 to 6

Representing Shape in 2D

- General purpose, simple overhead, reasonable efficiency: Vertex and Edge tables
- Each vertex listed once
- Each edge is ordered pair of indices into vertex table
- Sufficient information to draw shape and perform simple operations.
- Order does not matter, convention is edges listed in counterclockwise order.

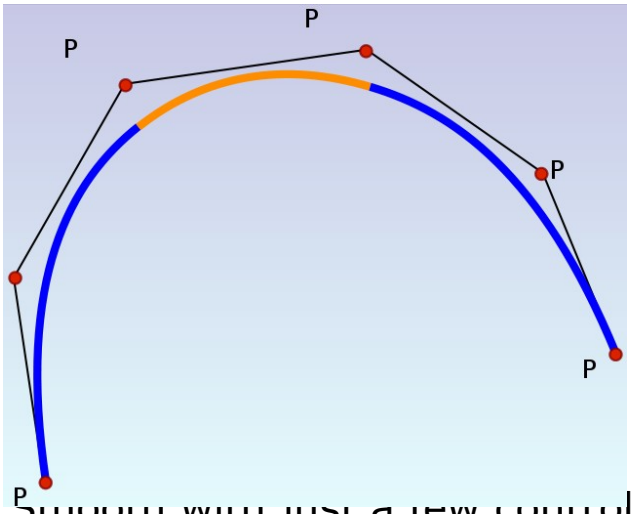
Vertexes	
1	(0,0)
2	(1,0)
3	(0,1)
4	(1,1)
5	(0.5,1.5)

Edges	
1	(1,2)
2	(2,4)
3	(4,5)
4	(5,3)
5	(3,1)



Splines (An Alternate Representation)

- How they work: Parametric curves governed by control points
- Mathematically: Several representations to choose from. More complicated than vertex lists. See chapter 23 of the book for more information.



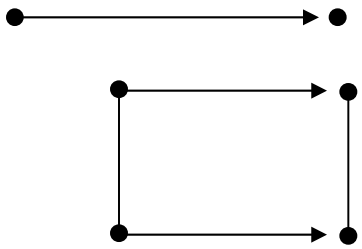
- Advantage: Smooth with just a few control points
- Disadvantage: Can be hard to control
- Uses:
  - representation of smooth shapes. Either as outlines in 2D or with Patches or Subdivision Surfaces in 3D
  - animation Paths for tweening
  - approximation of truncated Gaussian Filters

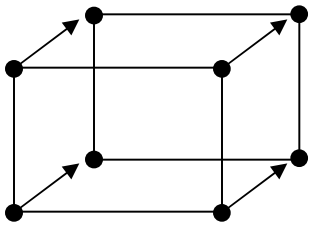
2D to 3D Object Definition

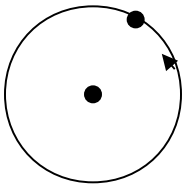
Vertices in motion *("Generative object description")*

- Line is drawn by tracing path of a point as it moves (one dimensional entity)

- Square drawn by tracing vertices of a line as it moves perpendicularly to itself (two dimensional entity)



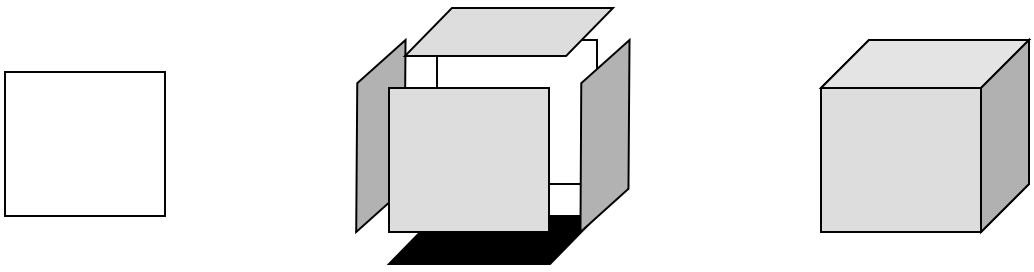




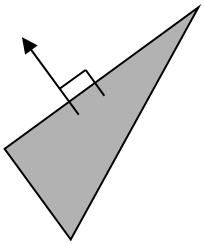
- Cube drawn by tracing paths of vertices of a square as it moves perpendicularly to itself (three-dimensional entity)

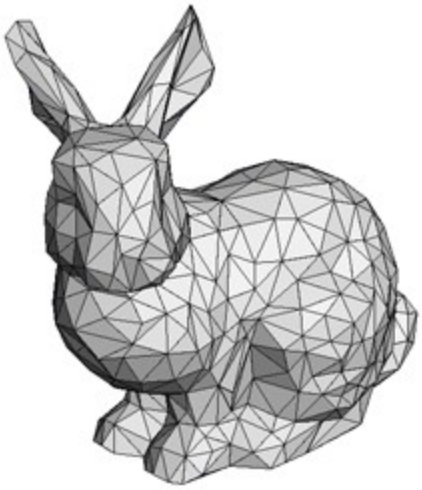
- Circle drawn by swinging a point at a fixed length around a center point

Building 3D Primitives

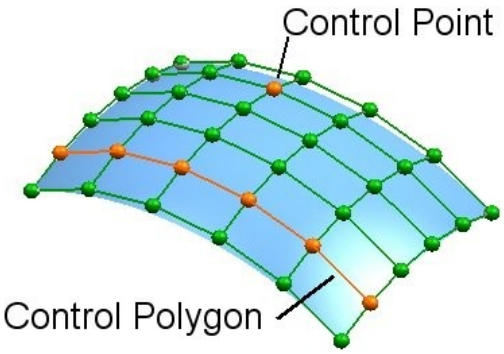


- Triangles and tri-meshes





- Parametric polynomials, like the aforementioned splines used to define surface patches.



Triangle Meshes

- Most common representation of shape in three dimensions
- All vertices of triangle are guaranteed to lie in one plane (unlike quadrilaterals or other polygons)
- Uniformity makes it easy to perform mesh operations: subdivision, simplification, etc.
- Many different ways to represent triangular meshes:

http://en.wikipedia.org/wiki/Polygon_mesh

Representing Shape in 3D

- Analogous to 2D: Vertex and Triangle Tables
- Each vertex gets listed once
- Each triangle is ordered triple of indices into the vertex table
- Edges between vertexes inferred from triangles
- Only need the triangular mesh representations to draw the shapes; need not do operations on meshes during CS123
- Counterclockwise ordering of vertices for normals

Face-Vertex Meshes

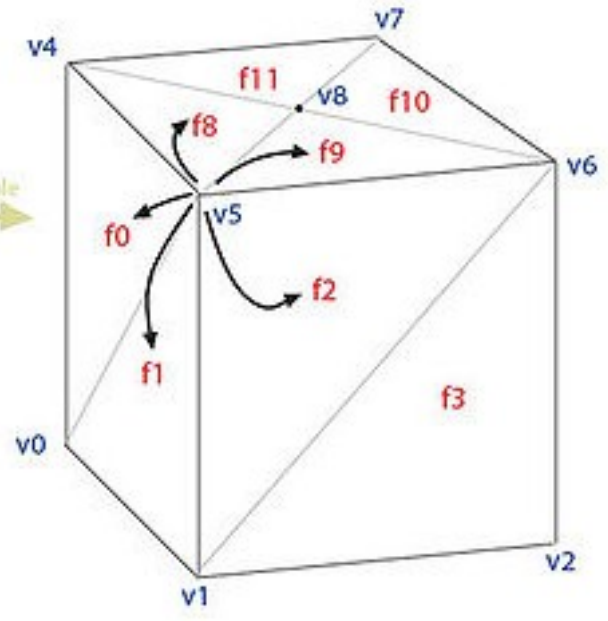
Face List

f0	v0 v4 v5
f1	v0 v5 v1
f2	v1 v5 v6
f3	v1 v6 v2
f4	v2 v6 v7
f5	v2 v7 v3
f6	v3 v7 v4
f7	v3 v4 v0
f8	v8 v5 v4
f9	v8 v6 v5
f10	v8 v7 v6
f11	v8 v4 v7
f12	v9 v5 v4
f13	v9 v6 v5
f14	v9 v7 v6
f15	v9 v4 v7

Vertex List

v0	0,0,0	f0 f1 f12 f15 f7
v1	1,0,0	f2 f3 f13 f12 f1
v2	1,1,0	f4 f5 f14 f13 f3
v3	0,1,0	f6 f7 f15 f14 f5
v4	0,0,1	f6 f7 f0 f8 f11
v5	1,0,1	f0 f1 f2 f9 f8
v6	1,1,1	f2 f3 f4 f10 f9
v7	0,1,1	f4 f5 f6 f11 f10
v8	.5,.5,0	f8 f9 f10 f11
v9	.5,.5,1	f12 13 14 15

example



http://upload.wikimedia.org/wikipedia/en/thumb/2/2d/Mesh_fv.jpg/500px-Mesh_fv.jpg