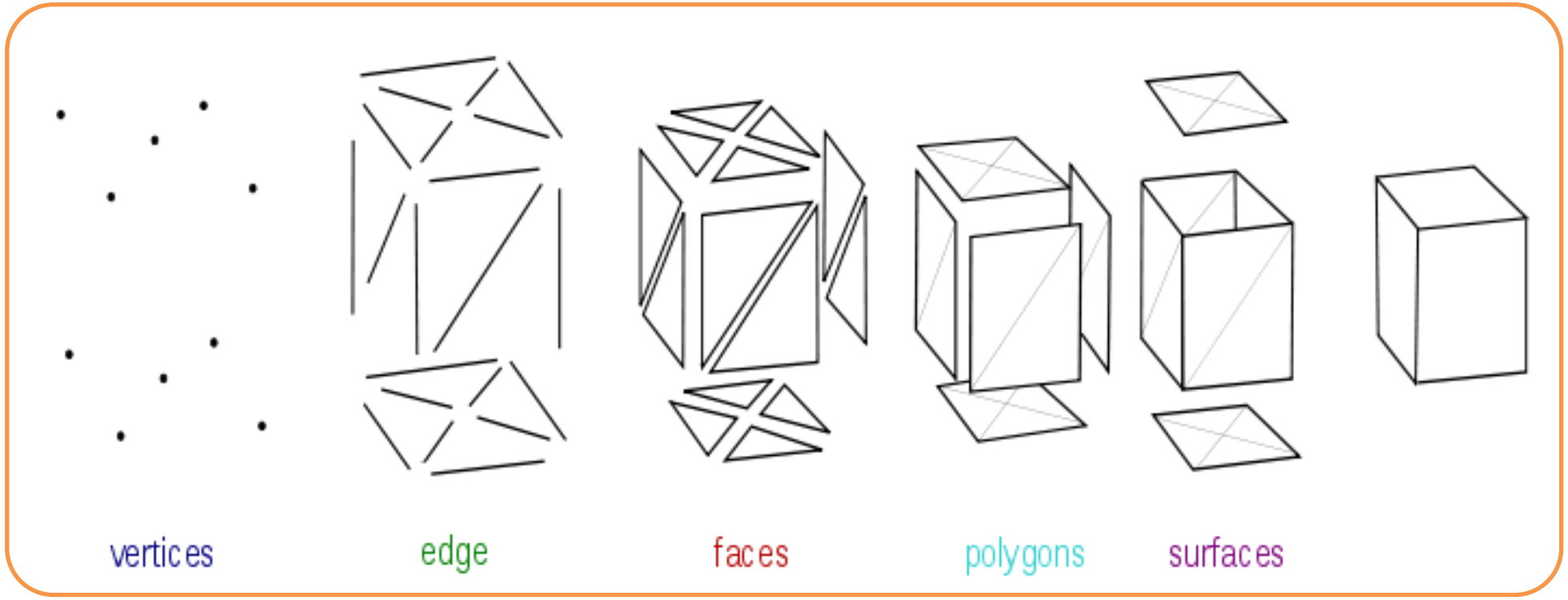
POLYGON MESH

Advance Computer Graphics

Definition

- **A polygon mesh** is a surface that is constructed out of a set of polygons that are joined together by common edges
- **A polygon mesh** is a collection of vertices, edges and faces that defines the shape of a polyhedral object in 3D computer graphics and solid modeling.

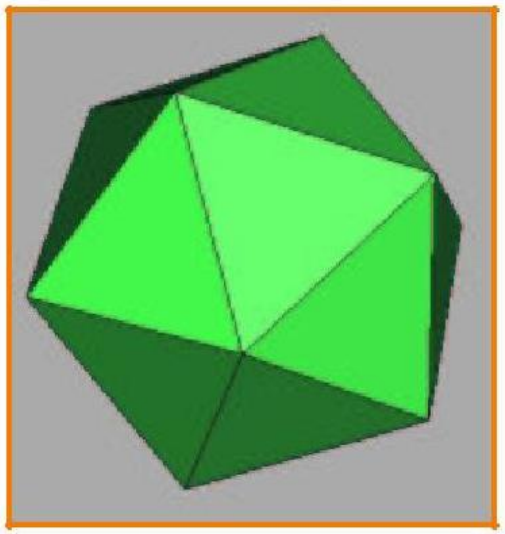
Elements Of Mesh Modeling

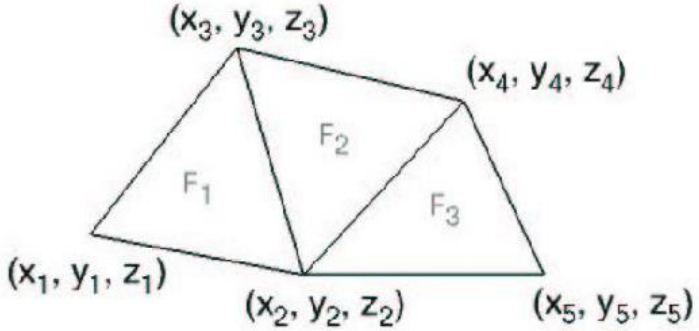


Vertex and Face Tables

Each face lists vertex references

- Shared vertices
- Still no topology information





VERTEX TABLE			
V_1	X_1	Y_1	Z_1
V_2	X_2	Y_2	Z_2
V_3	X_3	Y_3	Z_3
V_4	X_4	Y_4	Z_4
V_5	X_5	Y_5	Z_5

FACE TABLE			
F_1	V_1	V_2	V_3
F_2	V_2	V_4	V_3
F_3	V_2	V_5	V_4

Construction Of Polygonal Meshes

- Although it is possible to construct a mesh by manually specifying vertices and faces, it is much more common to build meshes using a variety of tools. A wide variety of 3D graphics software packages are available for use in constructing polygon meshes.
- One of the more popular methods of constructing meshes is box modeling, which uses two simple tools:
  - **Subdivide**
  - **extrude**

One of the more popular methods of constructing meshes is box modeling, which uses two simple tools

- The **subdivide** tool splits faces and edges into smaller pieces by adding new vertices. For example, a square would be subdivided by adding one vertex in the center and one on each edge, creating four smaller squares.

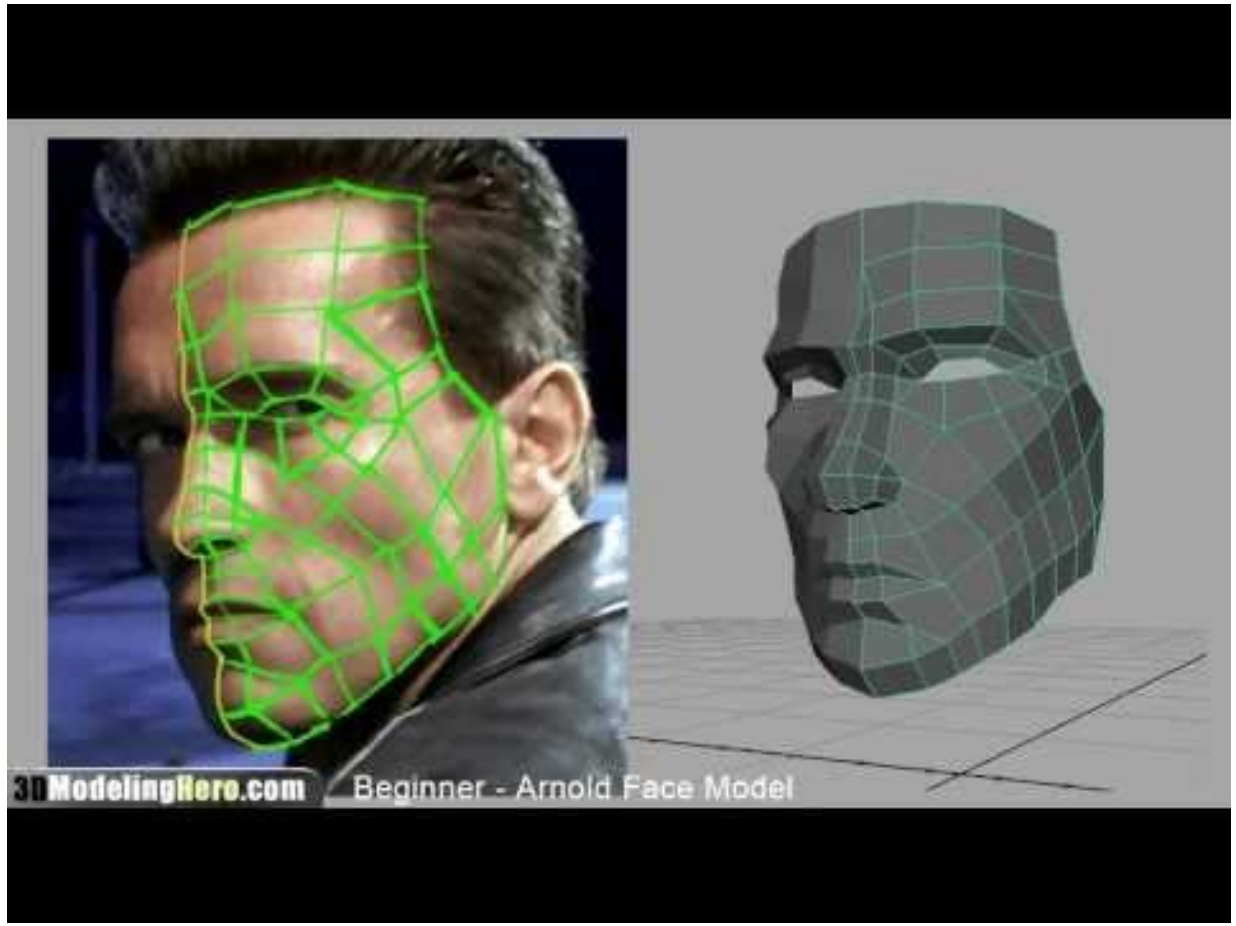
- The **extrude** tool is applied to a face or a group of faces. It creates a new face of the same size and shape which is connected to each of the existing edges by a face. Thus, performing the **extrude** operation on a square face would create a cube connected to the surface at the location of the face.

Construction Of Polygonal Meshes

- A second common modeling method is sometimes referred to as **inflation modeling** or **extrusion modeling**.
- In this method, the user creates a 2D shape which traces the outline of an object from a photograph or a drawing.
- The user then uses a second image of the subject from a different angle and extrudes the 2D shape into 3D, again following the shape's outline

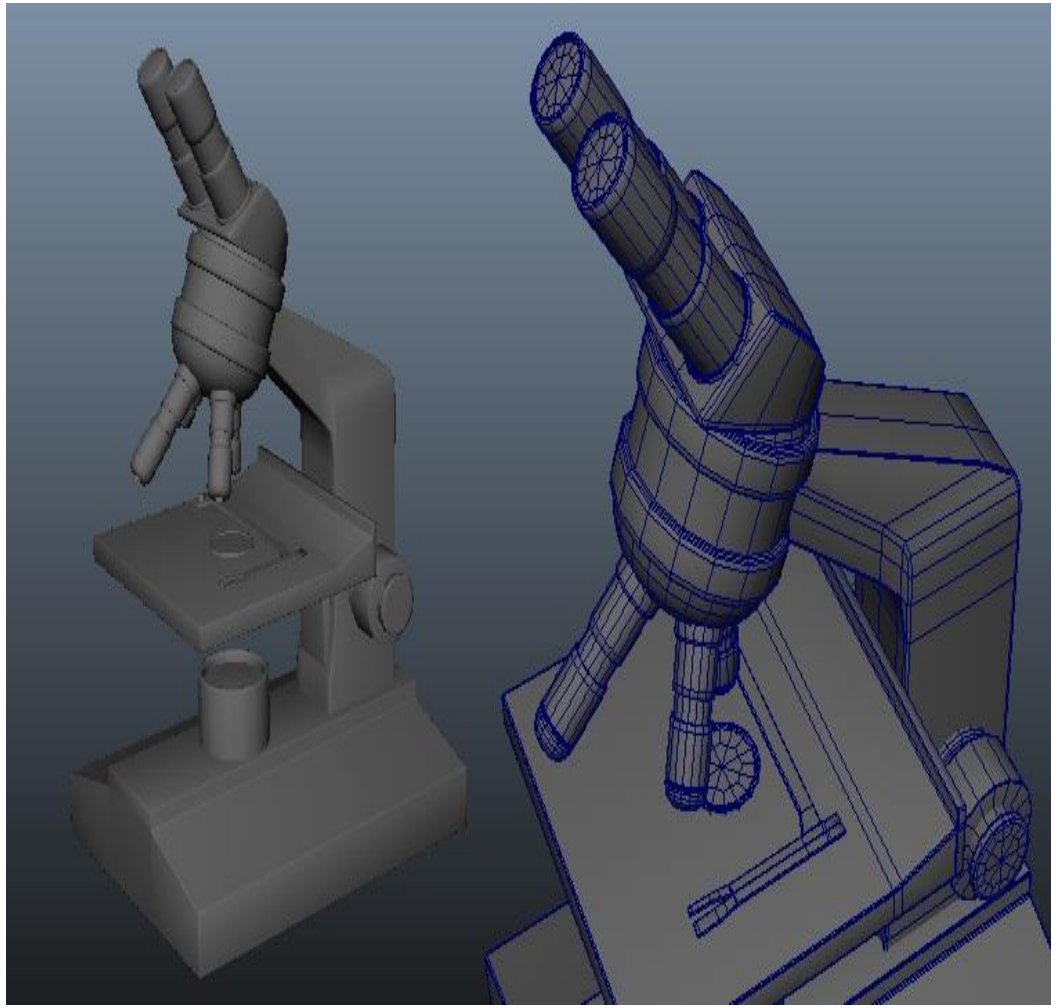
Extrusion modeling

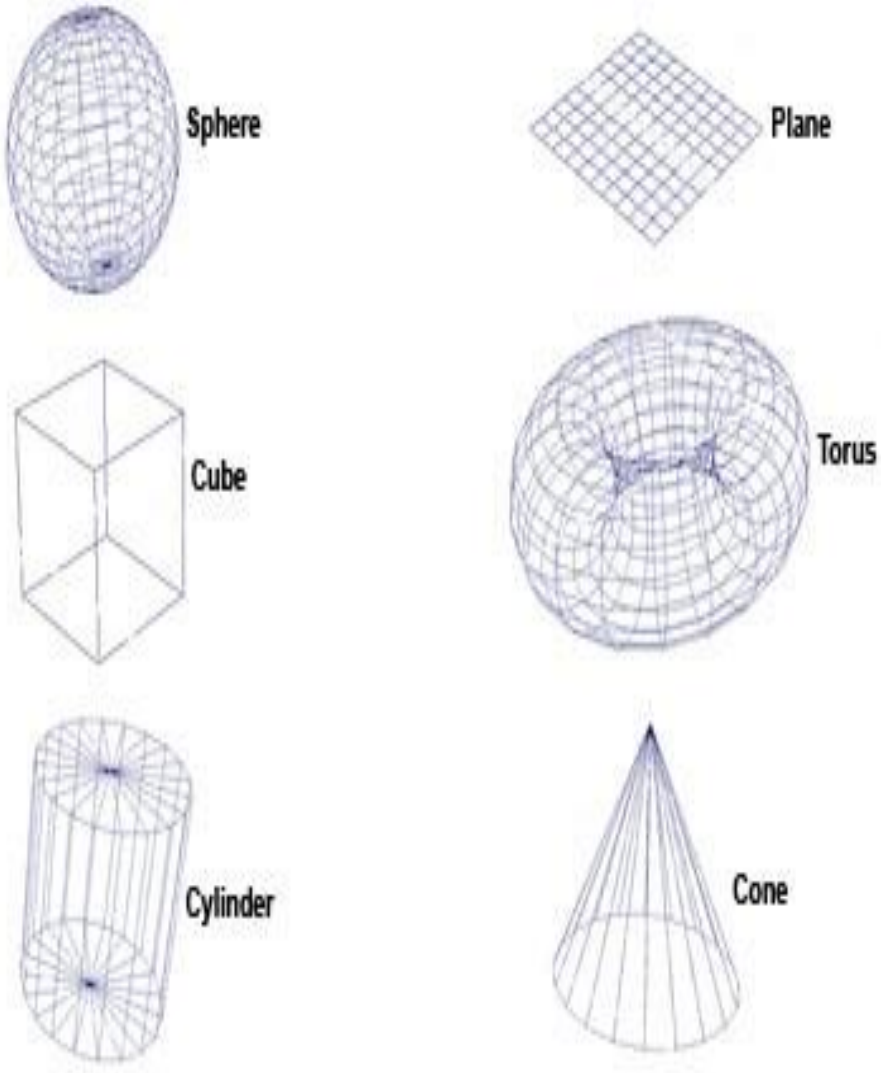
- This method is especially common for creating faces and heads. In general, the artist will model half of the head and then duplicate the vertices, invert their location relative to some plane, and connect the two pieces together. This ensures that the model will be symmetrical.



Construction Of Polygonal Meshes

- Another common method of creating a polygonal mesh is by connecting together various **primitives**, which are predefined polygonal meshes created by the modeling environment. Common primitives include:
- Cubes
- Pyramids
- Cylinders
- 2D primitives, such as squares, triangles, and disks
- Specialized or esoteric primitives, such as the Utah Teapot or Suzanne, Blender's monkey mascot.
- Spheres




Sphere
Plane
Cube
Torus
Cylinder
Cone

polygon meshes representation

Polygon meshes may be represented in a variety of ways, using different methods to store the vertex, edge and face data. These include:

1. Vertex-vertex meshes
2. Face-vertex meshes
3. Winged-edge meshes
4. Render dynamic meshes

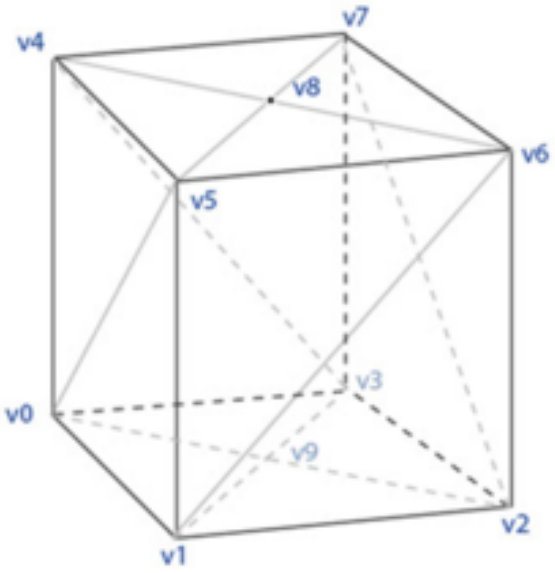
1- Vertex-vertex meshes

- Represent an object as a set of vertices connected to other vertices. This is the simplest representation, but not widely used since the face and edge information is implicit. Thus, it is necessary to traverse the data in order to generate a list of faces for rendering. In addition, operations on edges and faces are not easily accomplished.

Vertex-Vertex Meshes (VV)

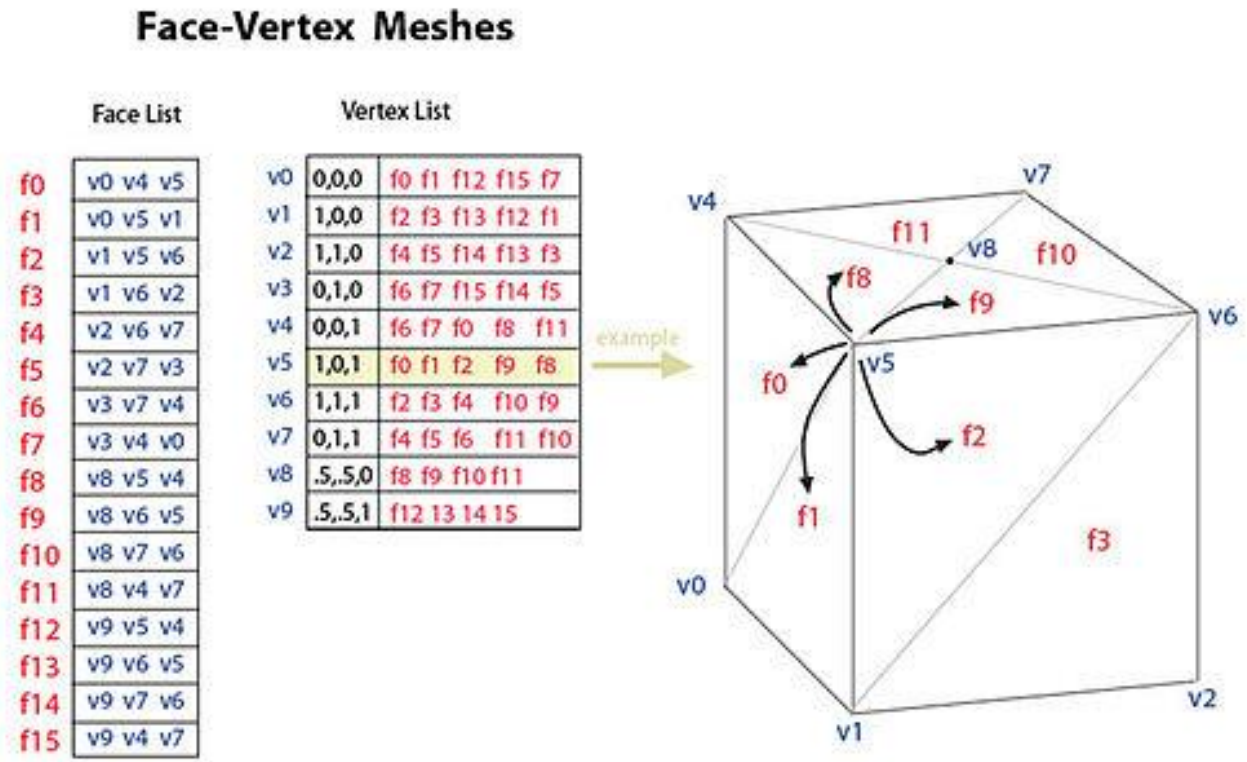
Vertex List

v0	0,0,0	v1 v5 v4 v3 v9
v1	1,0,0	v2 v6 v5 v0 v9
v2	1,1,0	v3 v7 v6 v1 v9
v3	0,1,0	v2 v6 v7 v4 v9
v4	0,0,1	v5 v0 v3 v7 v8
v5	1,0,1	v6 v1 v0 v4 v8
v6	1,1,1	v7 v2 v1 v5 v8
v7	0,1,1	v4 v3 v2 v6 v8
v8	.5,.5,1	v4 v5 v6 v7
v9	.5,.5,0	v0 v1 v2 v3



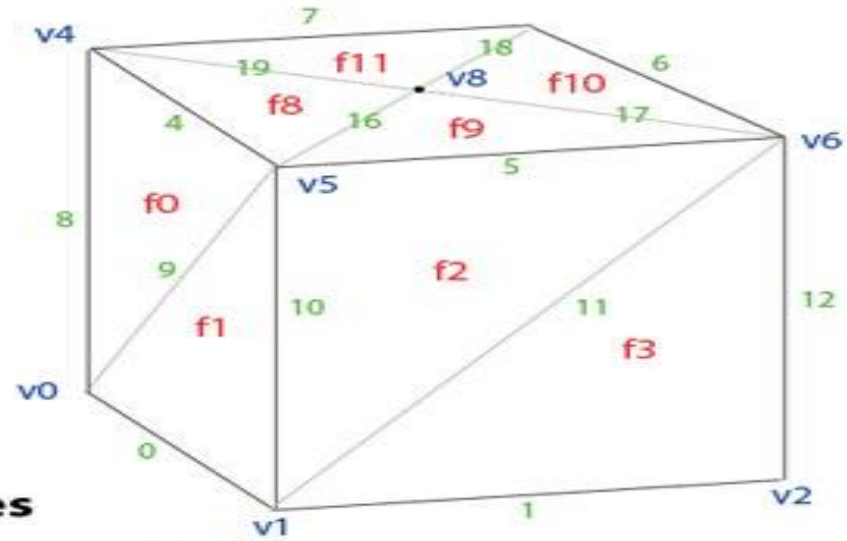
2- Face-vertex meshes

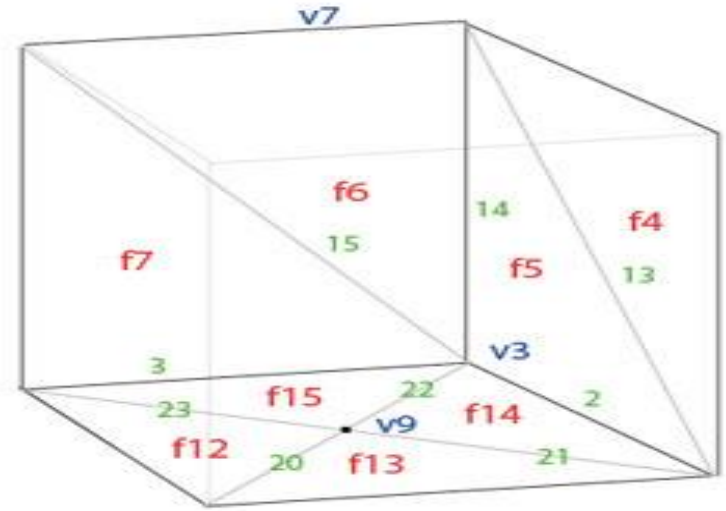
- Represent an object as a set of faces and a set of vertices. This is the most widely used mesh representation, being the input typically accepted by modern graphics hardware.



3- Winged-edge meshes

- Introduced by Baumgart 1975, **winged-edge meshes** explicitly represent the vertices, faces, and edges of a mesh. This representation is widely used in modeling programs to provide the greatest flexibility in dynamically changing the mesh geometry, because split and merge operations can be done quickly.





Winged-Edge Meshes

Face List

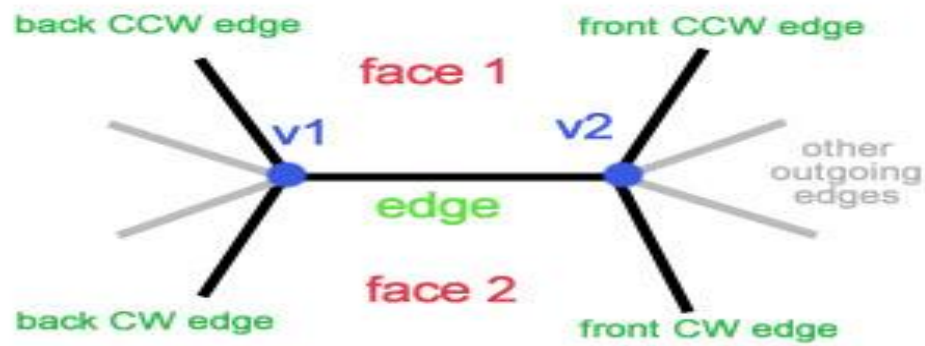
f0	4 8 9
f1	0 10 9
f2	5 10 11
f3	1 12 11
f4	6 12 13
f5	2 14 13
f6	7 14 15
f7	3 8 15
f8	4 16 19
f9	5 17 16
f10	6 18 17
f11	7 19 18
f12	0 23 20
f13	1 20 21
f14	2 21 22
f15	3 22 23

Edge List

e0	v0 v1	f1 f12	9 23 10 20
e1	v1 v2	f3 f13	11 20 12 21
e2	v2 v3	f5 f14	13 21 14 22
e3	v3 v0	f7 f15	15 22 8 23
e4	v4 v5	f0 f8	19 8 16 9
e5	v5 v6	f2 f9	16 10 17 11
e6	v6 v7	f4 f10	17 12 18 13
e7	v7 v4	f6 f11	18 14 19 15
e8	v0 v4	f7 f0	3 9 7 4
e9	v0 v5	f0 f1	8 0 4 10
e10	v1 v5	f1 f2	0 11 9 5
e11	v1 v6	f2 f3	10 1 5 12
e12	v2 v6	f3 f4	1 13 11 6
e13	v2 v7	f4 f5	12 2 6 14
e14	v3 v7	f5 f6	2 15 13 7
e15	v3 v4	f6 f7	14 3 7 15
e16	v5 v8	f8 f9	4 5 19 17
e17	v6 v8	f9 f10	5 6 16 18
e18	v7 v8	f10 f11	6 7 17 19
e19	v4 v8	f11 f8	7 4 18 16
e20	v1 v9	f12 f13	0 1 23 21
e21	v2 v9	f13 f14	1 2 20 22
e22	v3 v9	f14 f15	2 3 21 23
e23	v0 v9	f15 f12	3 0 22 20

Vertex List

v0	0,0,0	8 9 0 23 3
v1	1,0,0	10 11 1 20 0
v2	1,1,0	12 13 2 21 1
v3	0,1,0	14 15 3 22 2
v4	0,0,1	8 15 7 19 4
v5	1,0,1	10 9 4 16 5
v6	1,1,1	12 11 5 17 6
v7	0,1,1	14 13 6 18 7
v8	.5,.5,0	16 17 18 19
v9	.5,.5,1	20 21 22 23



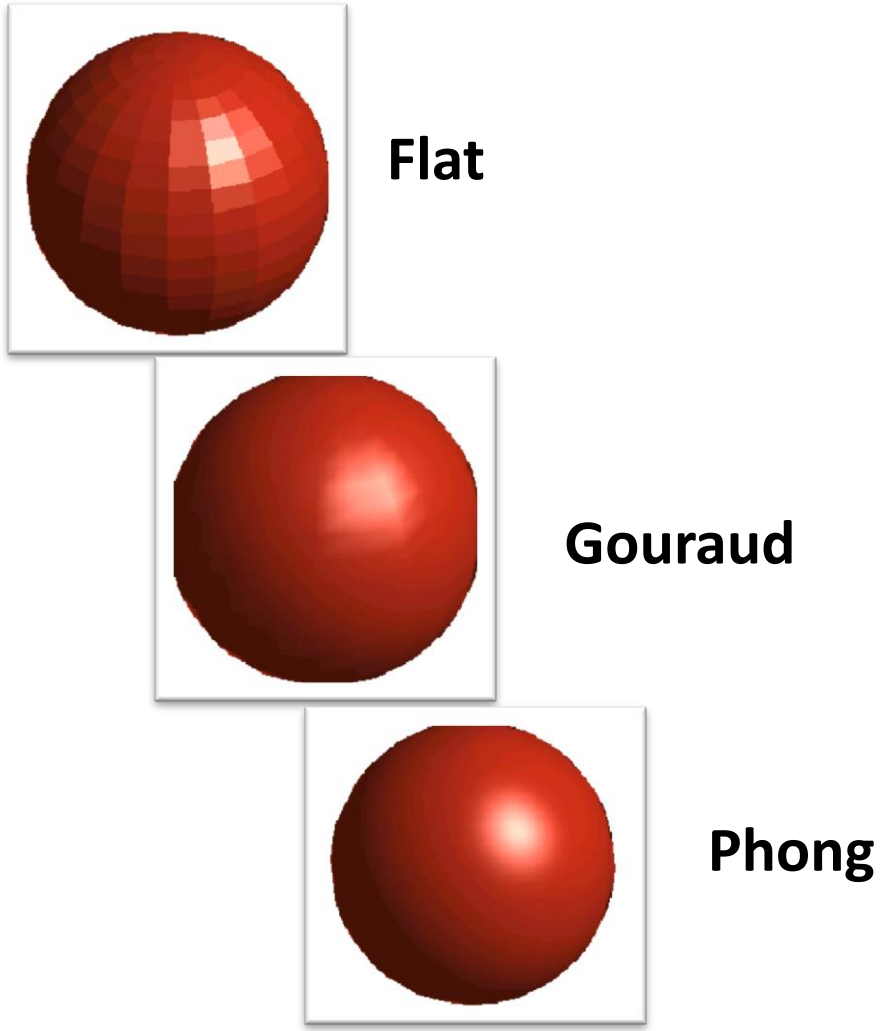
Winged Edge Structure

4- Render dynamic meshes

- Winged-edge meshes are not the only representation which allows for dynamic changes to geometry. A new representation which combines winged-edge meshes and face-vertex meshes is the **render dynamic mesh**, which explicitly stores both, the vertices of a face and faces of a vertex (like Face Vertics meshes), and the faces and vertices of an edge (like winged-edge).

Shading Models

- Flat Shading (less computation needed)
- Gouraud shading
- Phong Shading (heavy computation needed)

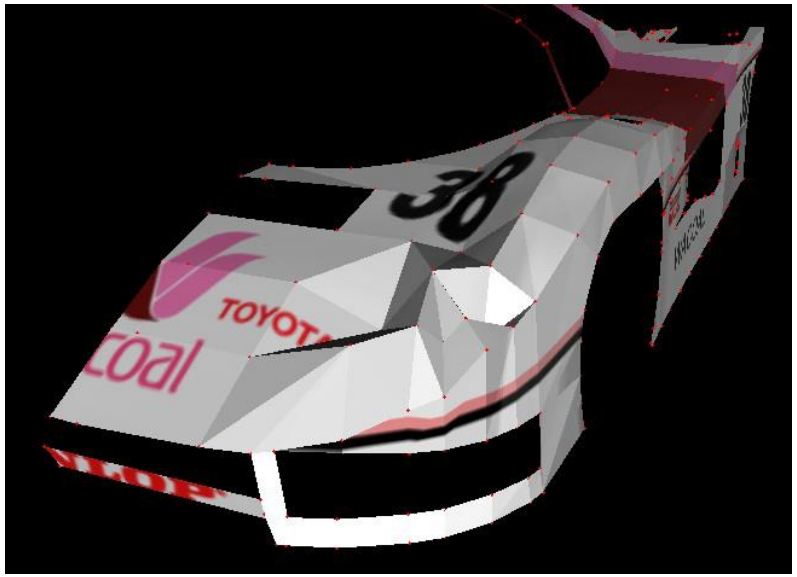


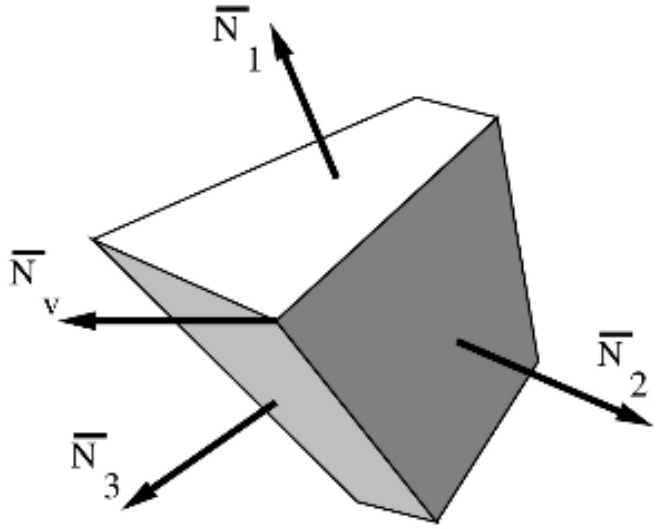
How to compute the normal vectors?

- The mesh is a set of polygons
- We can compute the normal vectors for each polygon (triangle)
- We can color the whole polygon using the same normal
- vector (flat shading)

Flat Shading

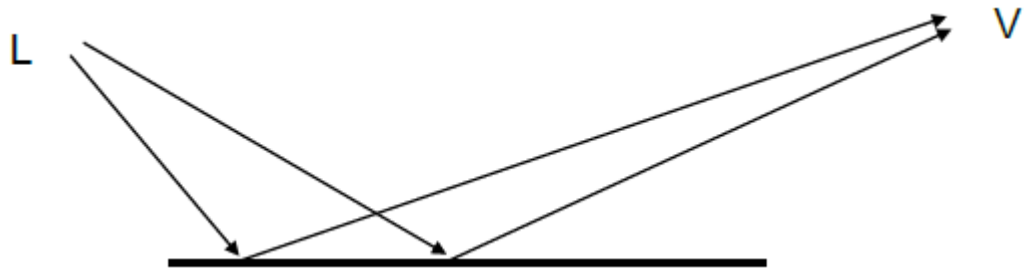
- Compute the color at the middle of the polygon
- All points in the same polygon are colored by the same color
- One illumination calculation per polygon Assign all pixels inside each polygon the same color





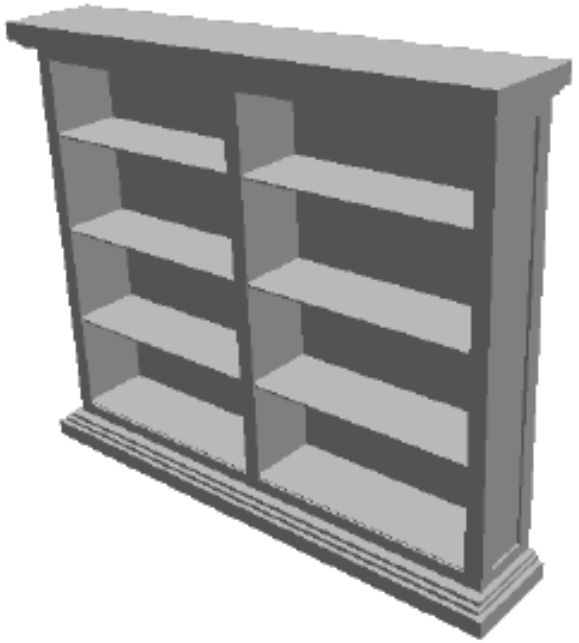
Flat Shading

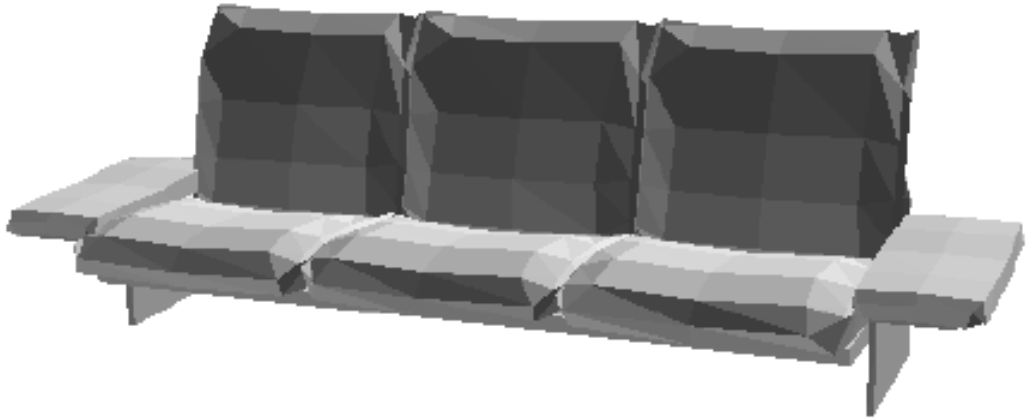
- A fast and simple method.
- Gives reasonable result only if all of the following assumptions are valid: The object is really a polyhedron.
    - Light source is far away from the surface so that N•L is constant over each polygon.
    - Viewing position is far away from the surface so that V•R is constant over each polygon.



Flat Shading

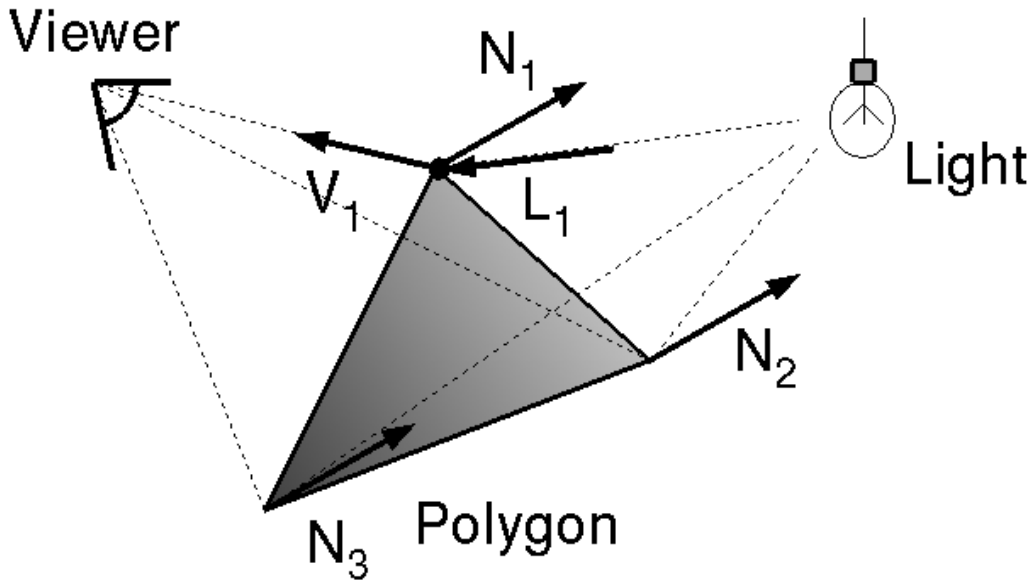
- Objects look like they are composed of polygons
- OK for polyhedral objects
- Not so good for ones with smooth surfaces





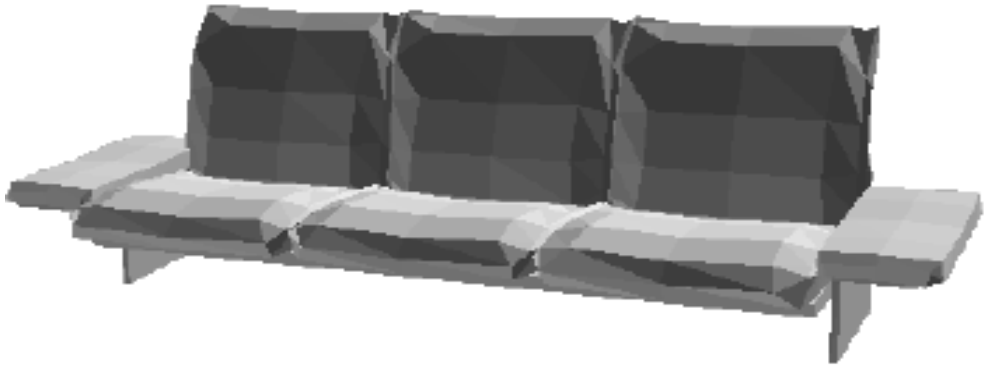
Gouraud Shading (Smooth Shading)

- Compute the color at each vertex first
- Compute the color inside the polygon by interpolating the colors of the vertices composing the polygon

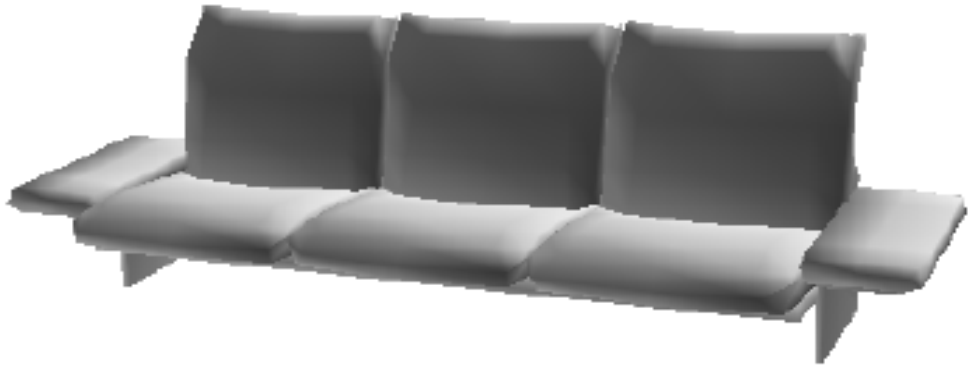


Gouraud Shading (Smooth Shading)

- Produces smoothly shaded polygonal mesh Piecewise linear approximation
- Need fine mesh to capture subtle lighting effects



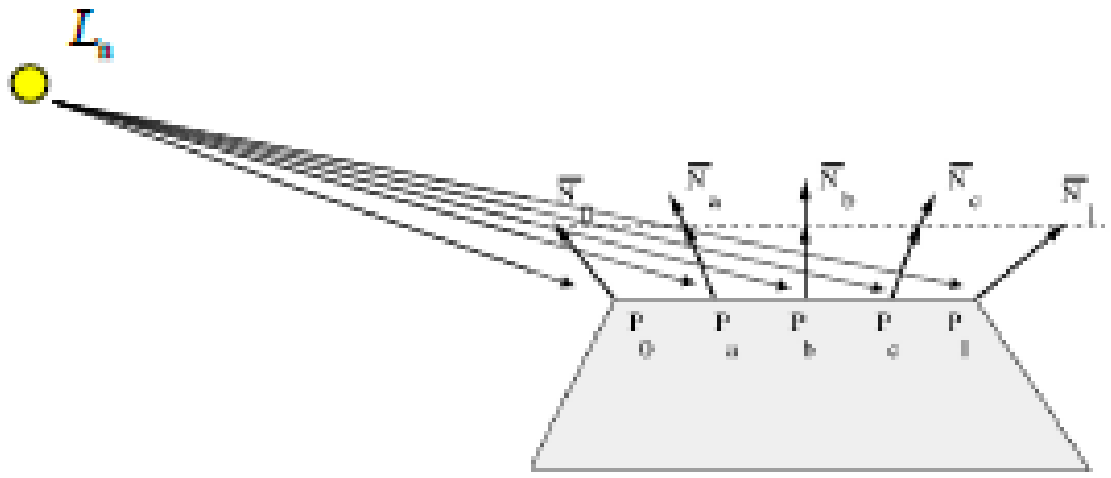
Flat Shading



Gouraud Shading

Phong Shading

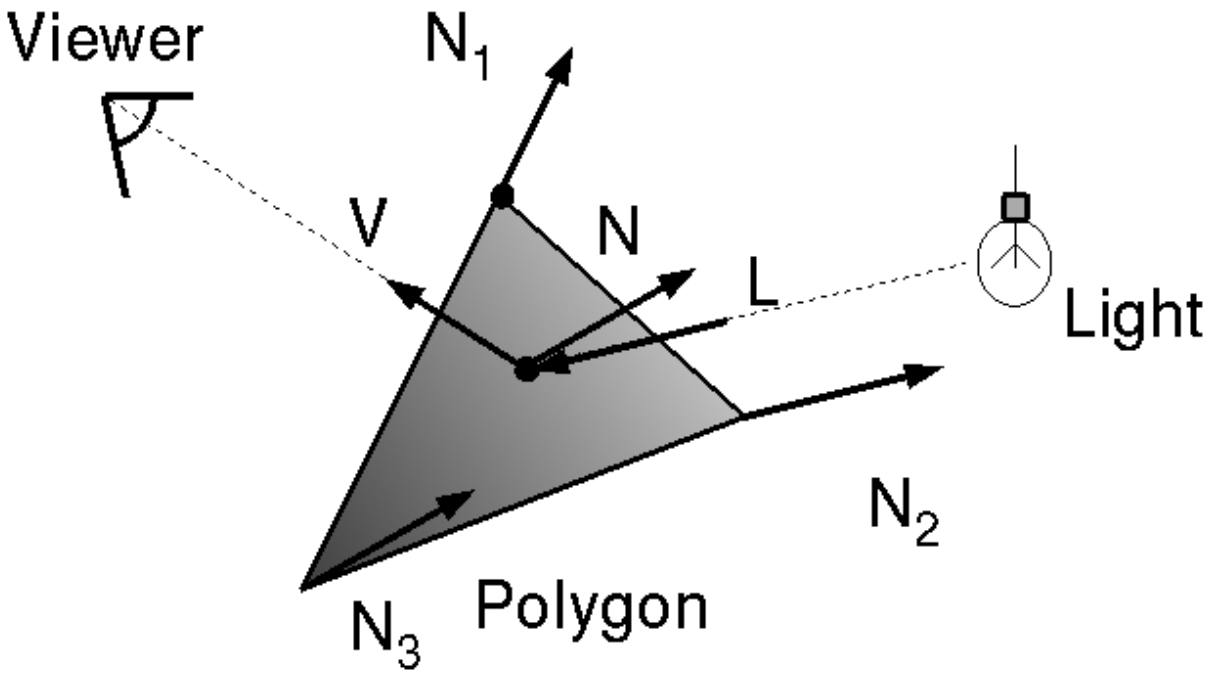
- Interpolating the normal vectors at the vertices
- Do the computation of illumination at each point in the polygon
- Apply the illumination model along each scan line to calculate pixel intensities for each surface point



Phong shading model

Phong Shading

- One lighting calculation per pixel.
- Approximate surface normals for points inside polygons by bilinear interpolation of normals from vertices



What's Bad About Polygons?

- What are some disadvantages of polygonal representations?

Polygons Aren’t Great

- They are always an approximation to curved surfaces
  - But can be as good as you want, if you are willing to pay in size
  - Normal vectors are approximate
  - They throw away information
  - Most real-world surfaces are curved, particularly natural surfaces
- They can be very unstructured
- It is difficult to perform many geometric operations
  - Results can be unduly complex, for instance

Thank you