COSC 414/519I: Computer Graphics

2023W2

Shan Du

Computation of Vectors

- The illumination and reflection models that we have are sufficiently general that they can be applied to curved or flat surfaces, to parallel or perspective views, and to distant or near surfaces.
- Most of the calculations for rendering a scene involve the determination of the required vectors and dot products.

Computation of Vectors

- For each special case, simplifications are possible. For example, if the surface is a flat polygon, the normal is the same at all points on the surface. If the light source is far from the surface, the light direction is the same at all points.

Normal Vectors

- A plane can be described by

$$ax + by + cz + d = 0$$

It can also be written in terms of the normal to the plane n, and a point p_0 on the plane

$$n \cdot (p - p_0) = 0$$

where p is any point (x, y, z) on the plane.

- Compare the two forms, we can see

$$n = \begin{bmatrix} a \\ b \\ c \end{bmatrix} \text{ or } n = \begin{bmatrix} a \\ b \\ c \\ 0 \end{bmatrix}$$

Normal Vectors

- If we have three non-collinear points - p_0, p_1, p_2 - that are in this plane, the vectors p_2- p_0 and p_1- p_0 are in the plane, and we can use their cross product to find the normal

$$n = (p_2 - p_0) \times (p_1 - p_0)$$

Normal Vectors

- For curved surfaces, how we compute normal depends on how we represent the surface.
- For a unit sphere centered at the origin, the equation is

$$f(x, y, z) = x^2 + y^2 + z^2 - 1 = 0$$

Or in vector form

$$f(p) = p \cdot p - 1 = 0$$

Normal Vectors

- The normal is given by the gradient vector, which is defined by the column matrix

$$n = \begin{bmatrix} \dfrac{\partial f}{\partial x} \\ \dfrac{\partial f}{\partial y} \\ \dfrac{\partial f}{\partial z} \end{bmatrix} = \begin{bmatrix} 2x \\ 2y \\ 2z \end{bmatrix} = 2p$$

Normal Vectors

- The sphere could also be represented in parametric form. In this form, the *x*, *y*, and *z* values of a point on the sphere are represented independently in terms of two parameters *u* and *v*:

$$x = x(u, v)$$
$$y = y(u, v)$$
$$z = z(u, v)$$

This form is preferable in computer graphics, especially for representing curves and surfaces.

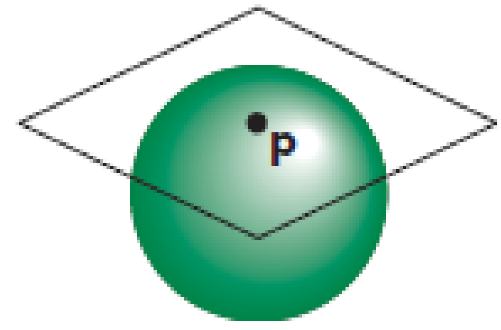
Normal Vectors

- For a particular surface, there may be multiple parametric representations. One parametric representation for the sphere is

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

As *u* and *v* vary in the range $-\frac{\pi}{2} < u < \frac{\pi}{2}, -\pi < v < \pi$, we get all the points on the sphere.

Normal Vectors



Tangent plane to sphere.

- When we are using the parametric form, we can obtain the normal from the tangent plane, at a point $p(u,v) = [x(u,v)\ y(u,v)\ z(u,v)]^T$ on the surface.
- The tangent plane gives the local orientation of the surface at a point; we can derive it by taking the linear terms of the Taylor series expansion of the surface at *p*.

Normal Vectors

- The result is that at p, lines in the directions of the vectors represented by

$$\frac{\partial p}{\partial u} = \begin{bmatrix} \frac{\partial x}{\partial u} \\ \frac{\partial y}{\partial u} \\ \frac{\partial z}{\partial u} \end{bmatrix} \quad \frac{\partial p}{\partial v} = \begin{bmatrix} \frac{\partial x}{\partial v} \\ \frac{\partial y}{\partial v} \\ \frac{\partial z}{\partial v} \end{bmatrix}$$

lie in the tangent plane. We can use their cross product to obtain the normal $n = \frac{\partial p}{\partial u} \times \frac{\partial p}{\partial v}$.

Normal Vectors

- For the sphere, we find

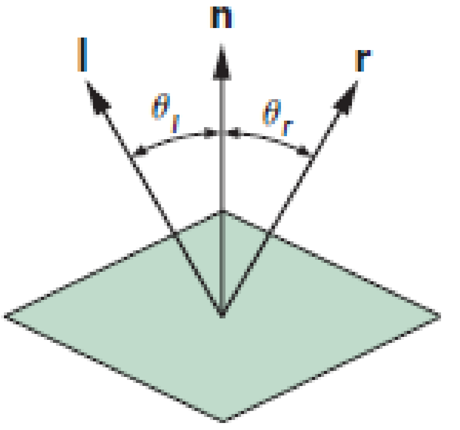
$$n = \cos u \begin{bmatrix} \cos u \sin v \\ \cos u \cos v \\ \sin u \end{bmatrix} = (\cos u)p$$

We are interested in only the direction of n; thus, we can divide $\cos u$ to obtain the unit normal to the sphere, $\mathrm{n} = p$.

- In WebGL, we will usually compute the vertex normals in the application and put them in a vertex array buffer just as we do for vertex positions.

Angle of Reflection

- We can use normal and the direction of the light source to compute the direction of a perfect reflection.



A mirror.

- For an ideal mirror: the angle of incidence is equal to the angle of reflection.
- The angle of incidence is the angle between the normal and the light source; the angle of reflection is the angle between the normal and the direction of reflection.

Angle of Reflection

- In 2D, only one angle satisfies the angle condition.
- But in 3D, there are infinite angles satisfying the condition.
- We must add the following statement: At a point *p* on the surface, the incoming light ray, the reflected light ray, and the normal at the point must all lie in the same plane.

Angle of Reflection

- The above two conditions are sufficient to determine *r* from *n* and *l*.
- We assume both *n* and *l* are normalized to unit length $|l| = |n| = 1$. We also want $|r| = 1$.
- If $\theta_i = \theta_r$, then $\cos\theta_i = cos\theta_r$. Using the dot product, $\cos\theta_i = l \cdot n = cos\theta_r = n \cdot r$.
- The coplanar implies that we can write *r* as a linear combination of *l* and *n*: $r = \alpha l + \beta n$.

Angle of Reflection

- Take the dot product with *n*, we have

$$n \cdot r = \alpha l \cdot n + \beta = l \cdot n$$

- We can get a second condition between α and β from our requirement that *r* also be of unit length; thus

$$1 = r \cdot r = \alpha^2 + 2\alpha\beta l \cdot n + \beta^2$$

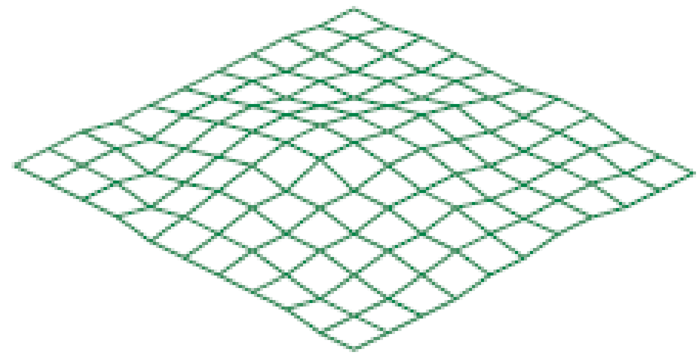
- Solving the two equations, we have

$$r = 2(l \cdot n)n - l$$

Polygonal Shading

- Decompose curved surface into many small, flat polygons.

- Each polygon has a well
defined normal vector.

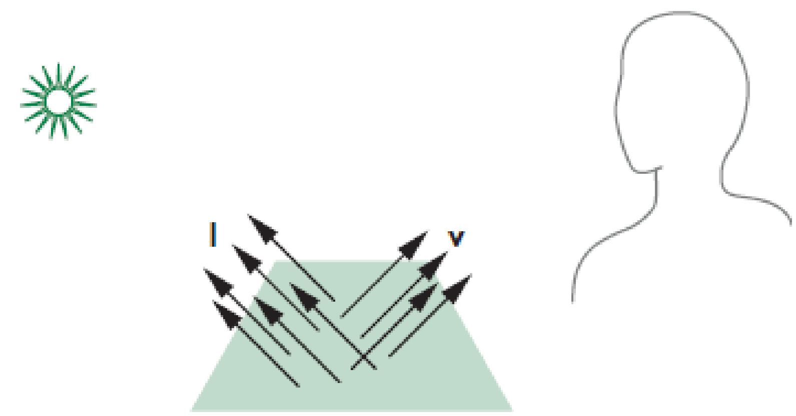


Polygonal mesh.

- We have three ways to shade the polygon: flat shading, smooth or Gouraud shading, and Phong shading.

Flat Shading

- The three vector – *l*, *n*, and *v* – can vary as we move from point to point on a surface.
- For a flat polygon, *n* is constant.
- If we assume a distant viewer, *v* is constant over the polygon.
- If the light is distant (at infinity), *l* is constant.

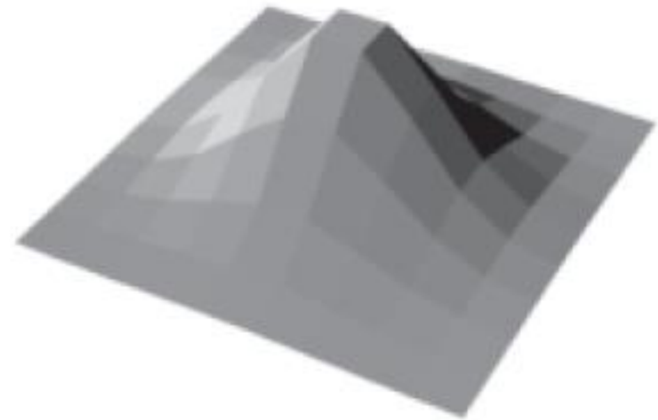


Distant source and viewer.

Flat Shading

- If the three vectors are constant, then the shading calculation needs to be carried out only once for each polygon, and each point on the polygon is assigned the same shade.
- This technique is flat, or constant shading.
- Flat shading will show differences in shading among the polygons in our mesh.
- If the light sources and viewer are near the polygon, the vectors *l* and *v* will be different for each polygon.

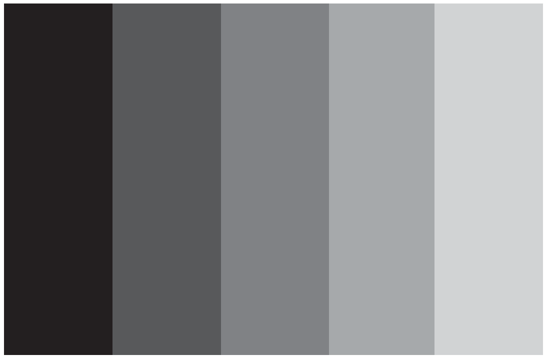
Flat Shading


Flat shading of polygonal mesh.

- However, if our polygon mesh has been designed for a smooth surface, flat shading will almost always be disappointing because we can see small differences in shading between adjacent polygons.
- The human visual system is sensitive to small differences in light intensity, due to a property know as lateral inhibition.

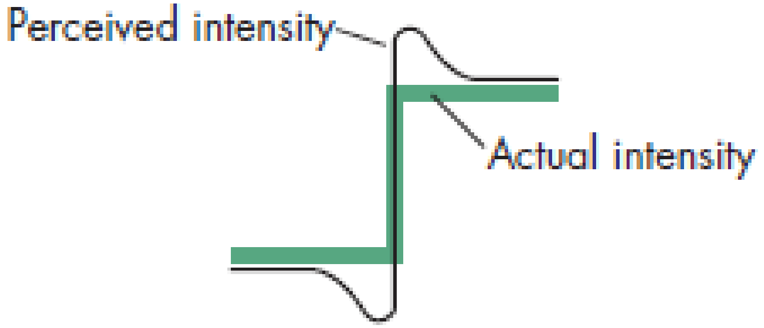
Flat Shading

- We can perceive the increases in brightness as overshooting on one side of an intensity step and undershooting on the other.



Step chart.

- We see strips, known as Mach Bands, along the edges.



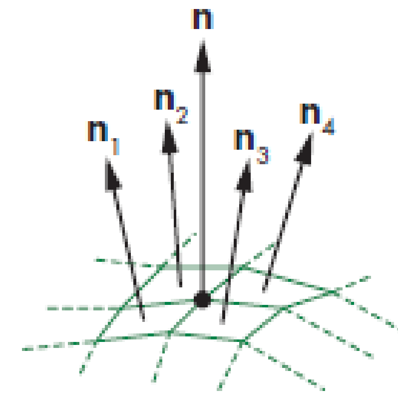
Perceived and actual intensities at an edge.

Smooth and Gouraud Shading

- Need to look for smoother shading techniques that do not produce large differences in shades at the edges of polygons.
- Suppose that the lighting calculation is made at each vertex using the material properties and the vector *n*, *v*, and *l* computed for each vertex, thus, each vertex will have its own color that the rasterizer can use to interpolate a shade for each fragment.

Smooth and Gouraud Shading

- Consider the mesh, because multiple polygons meet at interior vertices of the mesh, each of which has its own normal, the normal at the vertex is discontinuous.



Normals near interior vertex.

- Gouraud proposed to define the normal at the vertex through interpolation to achieve smoother shading.

Smooth and Gouraud Shading

- Four polygons meet. Each has its own normal. In Gouraud shading, the normal at a vertex is the normalized average of the normals of the polygons that share the vertex.
- The vertex normal is given by

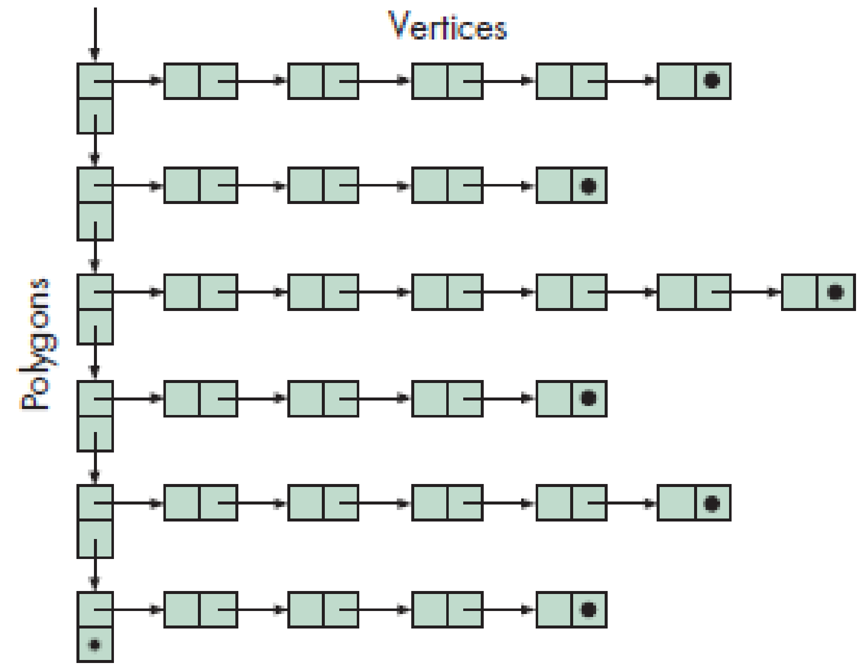
$$n = \frac{n_1 + n_2 + n_3 + n_4}{|n_1 + n_2 + n_3 + n_4|}$$

Smooth and Gouraud Shading

- How do we find the normals that we should average together?
  - If our program just specifies a list vertices (and other properties), we do not have the necessary information about which polygons share a vertex.
  - We need a data structure to represent a mesh.
  - Traversing the data structure can generate the vertices with the averaged normals.
  - Such a data structure should contain, at a minimum, polygons, vertices, normals, and material properties.

Smooth and Gouraud Shading

- The key information that must be represented in the data structure is which polygons meet at each vertex.



Mesh data structure.