

Shading

CS425: Computer Graphics I

Fabio Miranda

<https://fmiranda.me>

Overview



- Shading models:
 - Flat
 - Gouraud
 - Phong

Shading



- How do we compute the color for the whole surface?
- Illumination models (e.g., Phong, Blinn-Phong) compute the color of sample points. How do we color the entire object?

Shading

- What we need in the shaders?

$$f_{Phong}(\mathbf{L}_{light}, \mathbf{l}, \mathbf{v}, \mathbf{n}) = \underbrace{k_{ambient}}_{\text{uniform}} \underbrace{\mathbf{L}_{ambient}}_{\text{texture}} + \sum_{m \in \text{lights}} \underbrace{k_{diffuse}}_{\text{uniform}} (\underbrace{\hat{\mathbf{l}}_m \cdot \hat{\mathbf{n}}}_{\text{dot product}}) \underbrace{\mathbf{L}_{m,diffuse}}_{\text{texture}} + \underbrace{k_{specular}}_{\text{uniform}} (\underbrace{\hat{\mathbf{h}}_m \cdot \hat{\mathbf{n}}}_{\text{dot product}})^n \underbrace{\mathbf{L}_{m,specular}}_{\text{texture}}$$

- Uniforms

Shading

- What we need in the shaders?

$$f_{Phong}(\mathbf{L}_{light}, \mathbf{l}, \mathbf{v}, \mathbf{n}) = k_{ambient} \mathbf{L}_{ambient} + \sum_{m \in lights} k_{diffuse} (\hat{\mathbf{l}}_m \cdot \hat{\mathbf{n}}) \mathbf{L}_{m,diffuse} + k_{specular} (\hat{\mathbf{h}}_m \cdot \hat{\mathbf{n}})^n \mathbf{L}_{m,specular}$$

- Uniforms
- Normals

Shading

- What we need in the shaders?

$$f_{Phong}(\mathbf{L}_{light}, \mathbf{l}, \mathbf{v}, \mathbf{n}) = k_{ambient} \mathbf{L}_{ambient} + \sum_{m \in lights} k_{diffuse} (\hat{\mathbf{l}}_m \cdot \hat{\mathbf{n}}) \mathbf{L}_{m,diffuse} + k_{specular} (\hat{\mathbf{h}}_m \cdot \hat{\mathbf{n}})^n \mathbf{L}_{m,specular}$$

- Uniforms
- Normals
- Light position

Shading

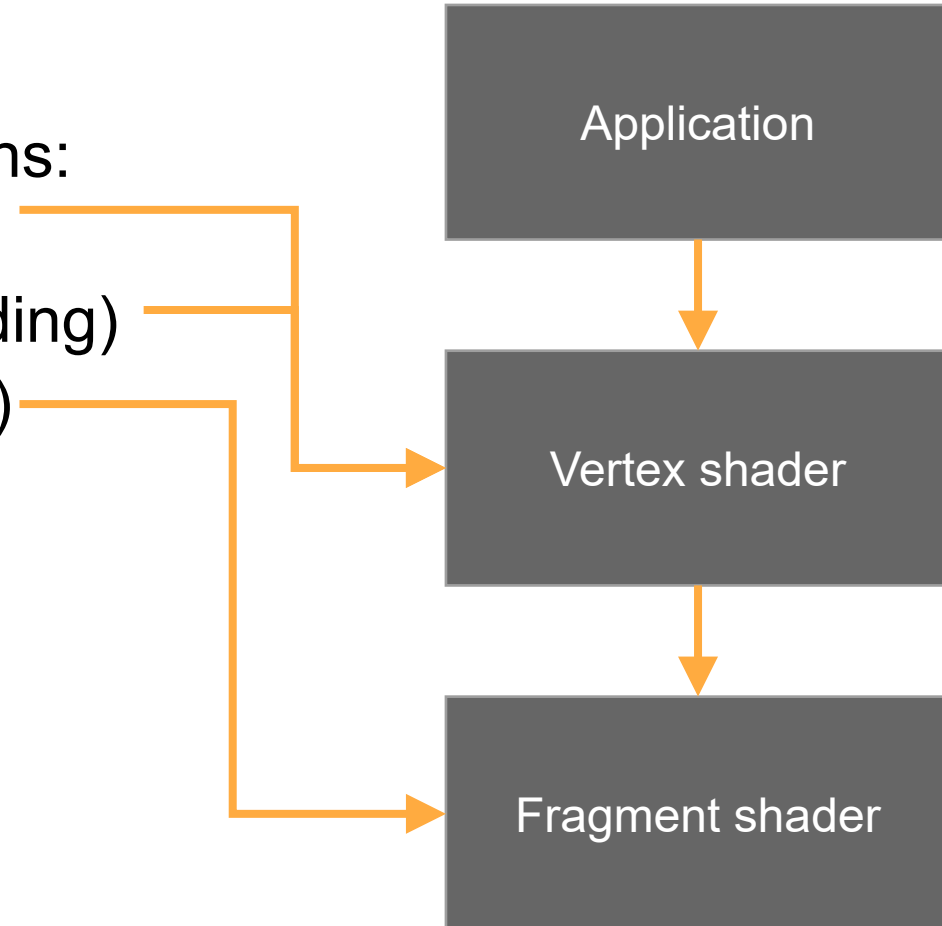
- What we need in the shaders?

$$f_{Phong}(\mathbf{L}_{light}, \mathbf{l}, \mathbf{v}, \mathbf{n}) = k_{ambient} \mathbf{L}_{ambient} + \sum_{m \in lights} k_{diffuse} (\hat{\mathbf{l}}_m \cdot \hat{\mathbf{n}}) \mathbf{L}_{m,diffuse} + k_{specular} (\hat{\mathbf{h}}_m \cdot \hat{\mathbf{n}})^n \mathbf{L}_{m,specular}$$

- Uniforms
- Normals
- Light position
- Half-vector: depends on \mathbf{l} and \mathbf{v}
 - Do we really need to send \mathbf{v} ? We can work on camera space!

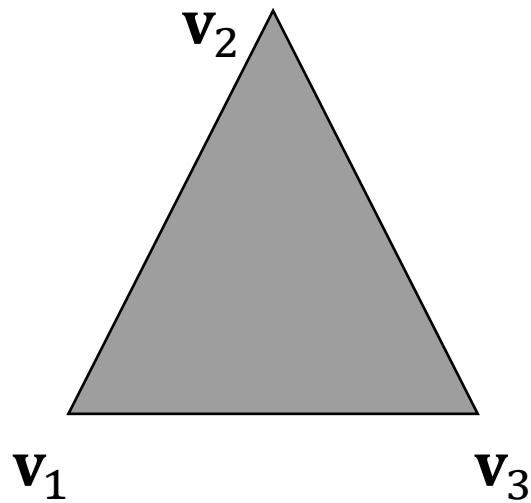
Shading: per-triangle, per-vertex, per-pixel

- We can compute shading operations:
 - Once per triangle (flat shading)
 - Once per vertex (Gouraud shading)
 - Once per pixel (Phong shading)

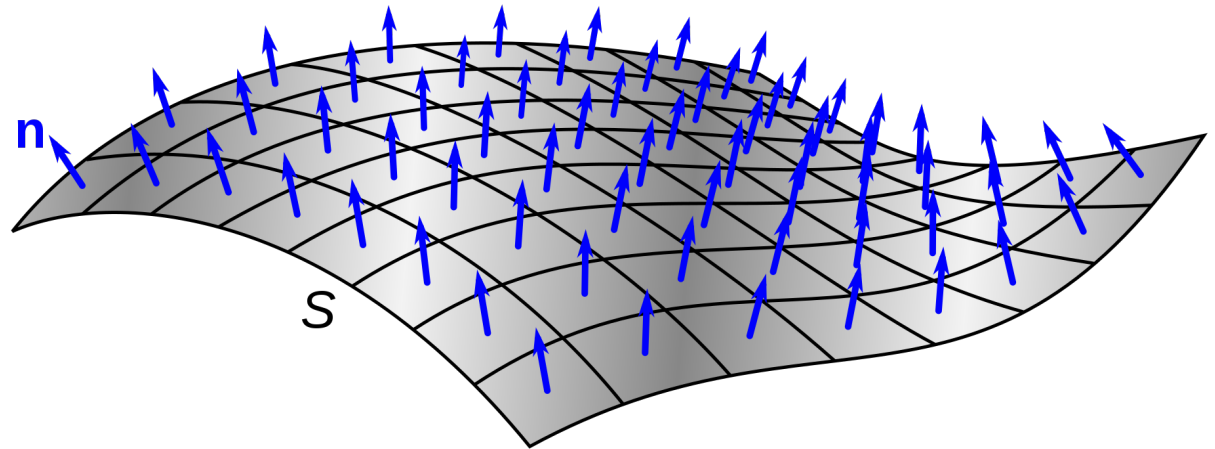


Computing normals

Triangle with vertices $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3$:

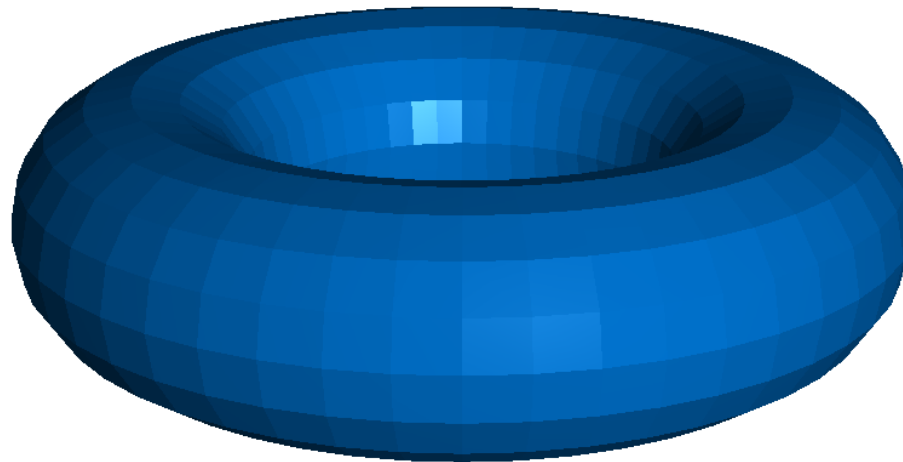


$$\mathbf{n} = (\mathbf{v}_2 - \mathbf{v}_1) \times (\mathbf{v}_3 - \mathbf{v}_1)$$



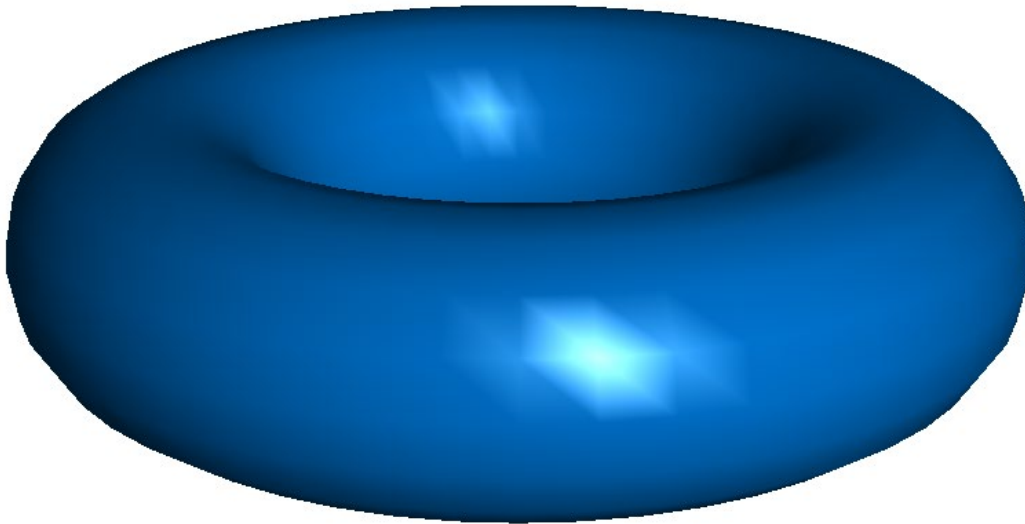
Flat shading

- Compute one color per polygon.
- All pixels in the same polygon are colored by the same color.

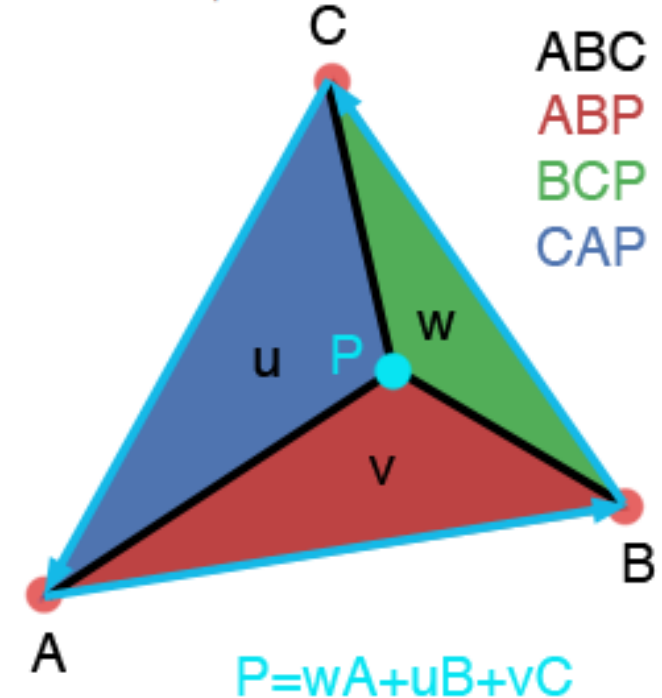


Gouraud shading

- Compute one color per vertex.
- Interpolate vertex **colors** across triangles.



© www.scratchapixel.com



Gouraud shading + WebGL

What uniforms in the vertex shader?

```
uniform float kAmbient;  
uniform float kDiffuse;  
uniform float kSpecular;  
uniform float specExponent;  
uniform vec3 colorAmbient;  
uniform vec3 colorDiffuse;  
uniform vec3 colorSpecular;
```

```
uniform vec3 lightPos; // or lightDir
```

What attributes in the vertex shader?

```
in vec3 pos;  
in vec4 color;  
in vec3 normal;
```

One normal per vertex.

Vertex shader:

- Blinn-Phong operations, send color result to fragment shader.

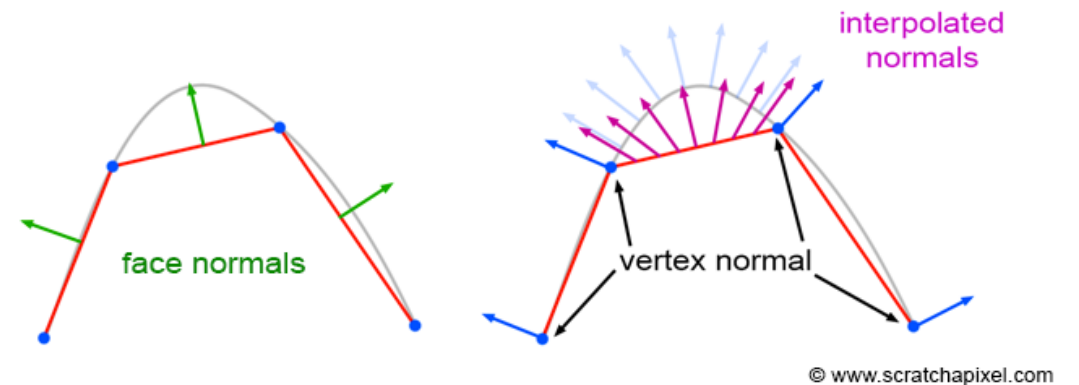
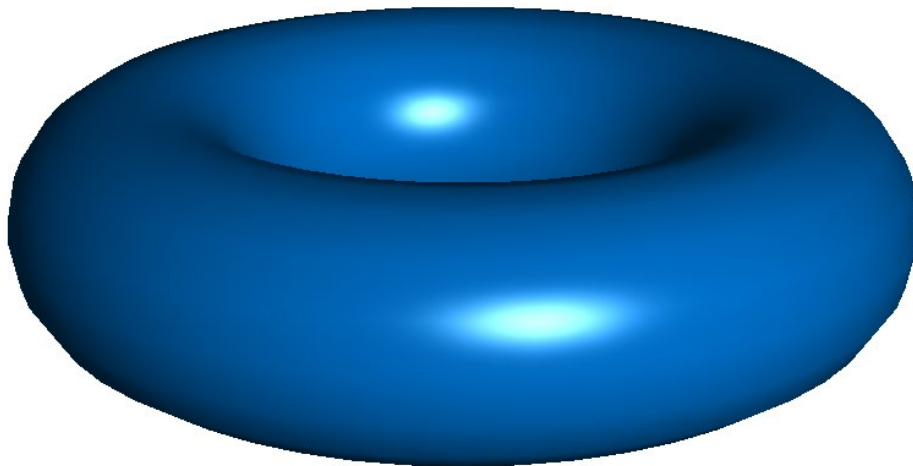
What about eye pos?

- No, if we perform everything in camera space.

```
posEyeSpace = modelViewMatrix * pos;
```

Phong shading

- One color per pixel.
- Interpolates vertex **normals** across triangles.
- Illumination model evaluated at each pixel.



Phong shading + WebGL

What uniforms in the frag shader?

```
uniform float kAmbient;  
uniform float kDiffuse;  
uniform float kSpecular;  
uniform float specExponent;  
uniform vec3 colorAmbient;  
uniform vec3 colorDiffuse;  
uniform vec3 colorSpecular;
```

```
uniform vec3 lightPos; // or lightDir
```

What varying in the frag shader?

```
in vec3 position;  
in vec3 normal;
```

Fragment shader:

- Blinn-Phong operations, send color result to framebuffer.

Again: camera space.

```
posEyeSpace = modelViewMatrix * pos;
```

$$f_{\text{Phong}}(\mathbf{L}_{\text{light}}, \mathbf{l}, \mathbf{v}, \mathbf{n}) = k_{\text{ambient}} \mathbf{L}_{\text{ambient}} + \sum_{m \in \text{lights}} k_{\text{diffuse}} (\hat{\mathbf{l}}_m \cdot \hat{\mathbf{n}}) \mathbf{L}_{m, \text{diffuse}} + k_{\text{specular}} (\hat{\mathbf{h}}_m \cdot \hat{\mathbf{n}})^n \mathbf{L}_{m, \text{specular}}$$

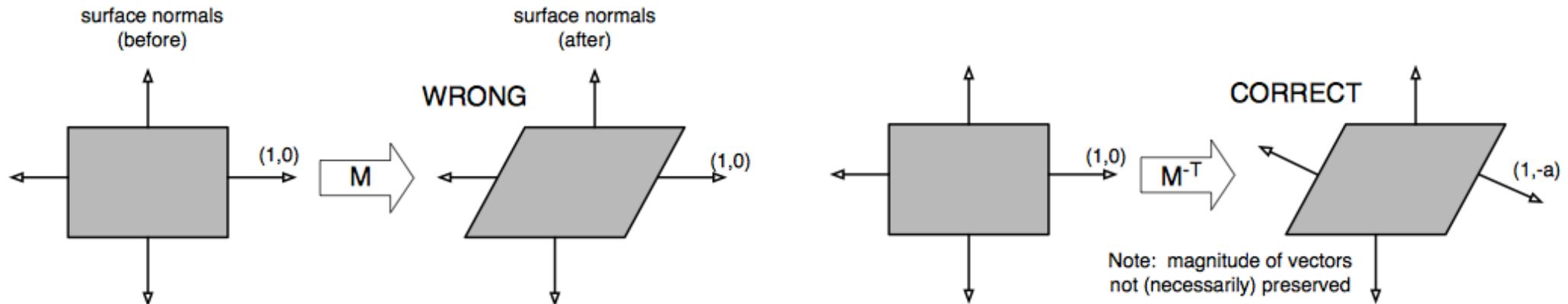
How can we transform normals?

- Vertex:

```
v = modelViewMatrix * vec4(position, 1);
```

- Normal:

```
n = uNormal * vec4(normal, 1);
```



$$\text{NormalMatrix} = (\text{ModelView}^{-1})^T$$

How can we transform normals?

$$\mathbf{t} = \mathbf{p}_2 - \mathbf{p}_1$$

$$\mathbf{t} * \mathbf{M} = (\mathbf{p}_2 - \mathbf{p}_1) * \mathbf{M}$$

$$\mathbf{t}' = \mathbf{p}_2' - \mathbf{p}_1'$$

\mathbf{n} and \mathbf{t} are perpendicular

\mathbf{n}' and \mathbf{t}' must also be perpendicular

We find normal matrix by solving: $\mathbf{n}' \cdot \mathbf{t}' = 0$

What is \mathbf{n}' ? $\mathbf{G} * \mathbf{n}$

What is \mathbf{t}' ? $\mathbf{M} * \mathbf{t}$

$$(\mathbf{G} * \mathbf{n}) \cdot (\mathbf{M} * \mathbf{t}) = 0$$

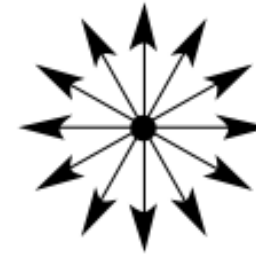
$$\mathbf{G} = (\mathbf{M}^{-1})^T$$

Point and directional light



Directional Light

```
uniform vec3 lightDir;
```



Point Light

```
lightDir = normalize(lightPos - pos);
```

Comparison

