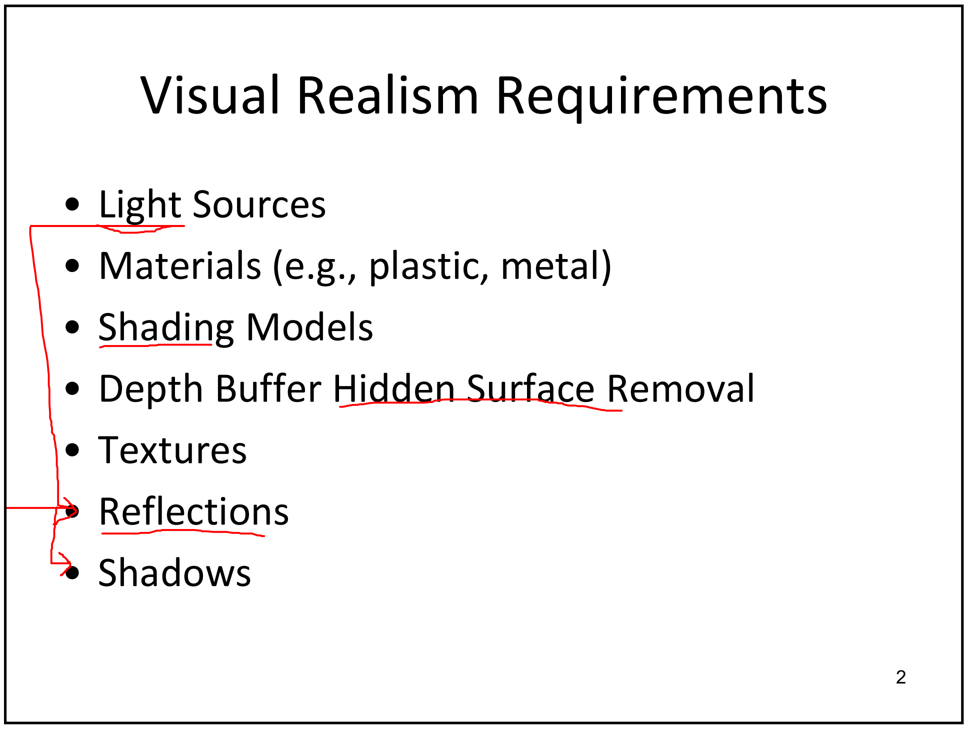
Rendering

Lighting &Shading

1

1



Visual Realism Requirements

- Light Sources
- Materials (e.g., plastic, metal)
- Shading Models
- Depth Buffer Hidden Surface Removal
- Textures
- Reflections
- Shadows

2

2

Rendering

- Rendering is the process of generating an image from a model, by means of a software program.
  - The model is a description of three dimensional objects in a strictly defined language or data structure.
  - It would contain geometry, viewpoint, texture, lighting information.
- Rendering is also used to describe the process of calculating effects in a video editing file to produce final video output.

3

3

Rendering Objects

- We know how to **model** mesh objects, manipulate a jib camera, view objects, and make pictures.
- Now we want to make these objects look visually interesting, realistic, or both.
- We want to develop methods of **rendering** a picture of the objects of interest: *computing* how each pixel of a picture should look.

4

4

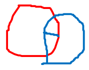
Rendering Objects (2)

- Much of rendering is based on different **shading models**, which describe how light from light sources interacts with objects in a scene.
  - It is impractical to simulate all of the physical principles of light scattering and reflection.
  - A number of approximate models have been invented that do a good job and produce various levels of realism.

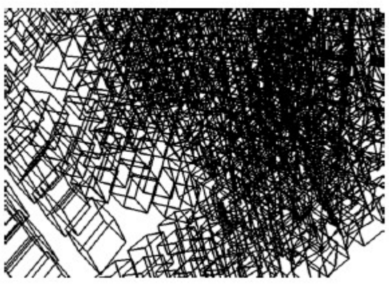
5

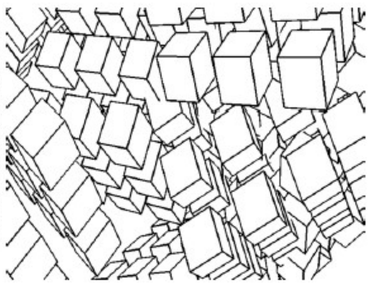
5

Rendering



- Rendering: deciding how a pixel should look
- Example: compare wireframe (left) to wire-frame with hidden surface removal (right)



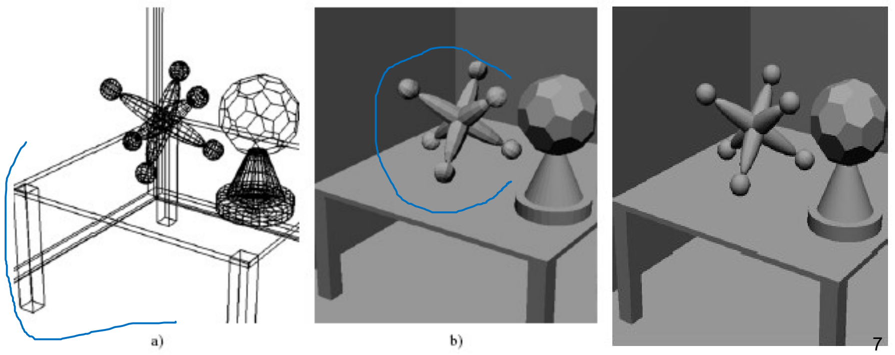


6

6

Rendering (2)

- Example: Compare mesh objects drawn using wire-frame, flat shading, smooth $\boxed{\text{(Gouraud)}}$ shading

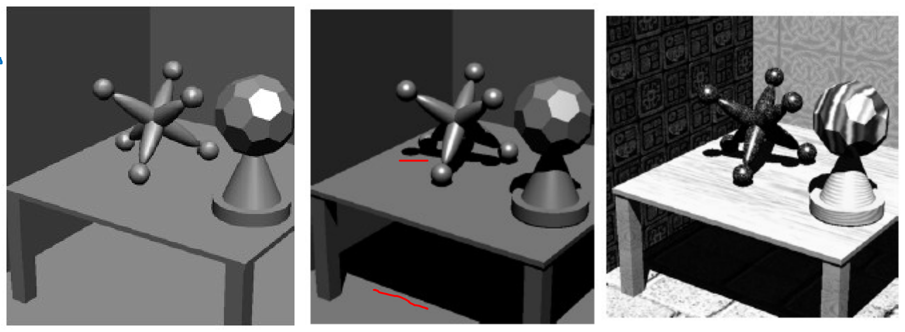


7

7

Rendering (3)

- Compare to images with specular highlights, shadows, textures



8

8

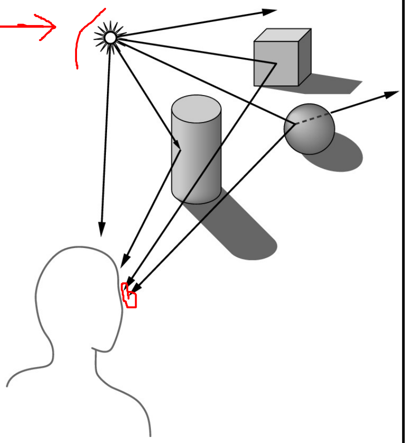
Lighting and Shading

9

9

Need for shading

- How do you make something look 3D?
- *Shading* that is appropriate for the lighting is the primary cue to 3D appearance



10

INTRODUCTION

- Shading is an important part of creating realistic objects.
  - It adds Shads and definition to otherwise flat representations.
- Illumination may fall unevenly across a polygon.
  - Can be calculated individually for each pixel.
    - Expensive to calculate.
- Faster shading algorithms exist.

11

11

SHADING

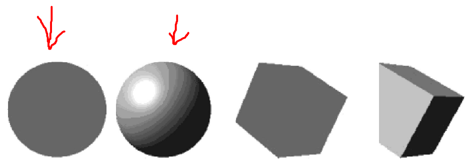
- There are several key determinants in the level of shading across a polygon.
  - Surface properties
    - Texture
    - Color
    - Material
  - Light sources
  - Relative positions and orientations

12

12

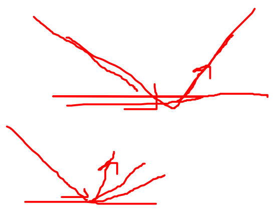
SHADING

- Shading is important as a mechanism for conveying information about 3D shapes in 2D.
- Representation of shapes with single colors render the images as 2D to our eyes.



Shading Models: Introduction

- Modelled in one of three ways.
  - Perfect specular reflection.
    - Light is reflected directionally.
  - Imperfect specular reflection.
    - Light is reflected imperfectly
  - Perfect diffuse reflection
    - Light is scattered in all directions.
- These three models can be used to determine the shading of polygons.



Shading Models: Introduction

- Assume to start with that light has no color, only brightness: R = G = B
- Assume we also have a point source of light (sun or lamp) and general ambient light, which doesn't come directly from a source, but through windows or scattered by the air.
  - Ambient light comes equally from all directions.
  - Point-source light comes from a single point.

15

15

Shading Models: Introduction (2)

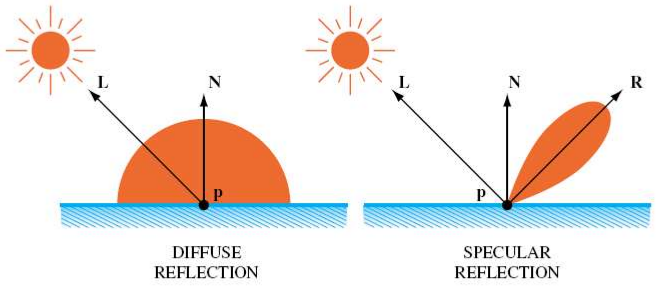
- When light hits an object, some light is absorbed (and turns into heat), some is reflected, and some may penetrate the interior (e.g., of a clear glass object).
- If all the light is absorbed, the object appears black and is called a blackbody.
- If all the light is transmitted, the object is visible only through refraction

16

16

Shading Models: Introduction (3)

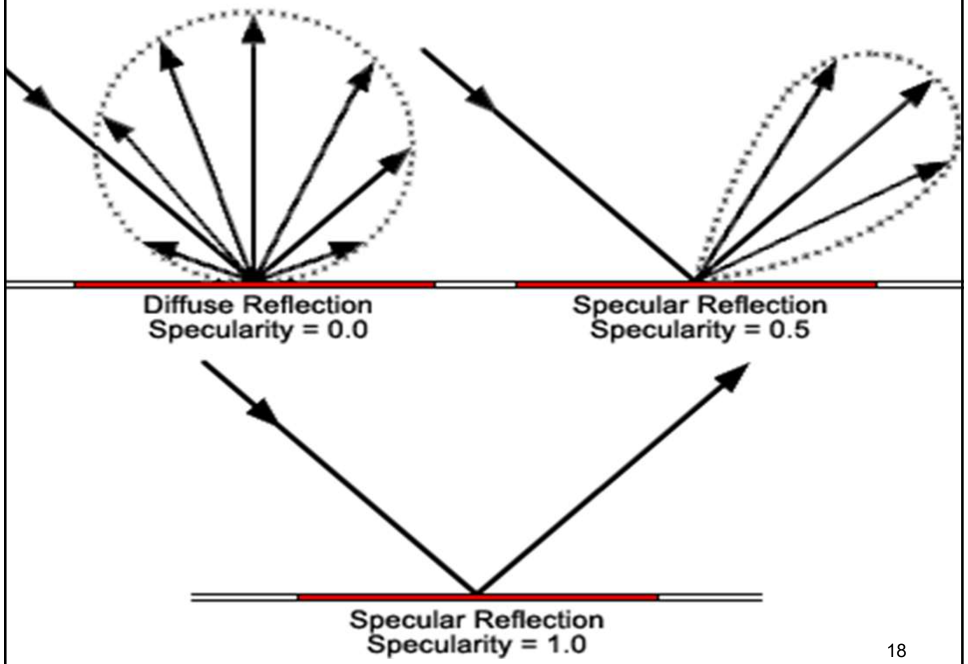
When light is reflected from an object, some of the reflected light reaches our eyes, and we see the object.



- **Diffuse** reflection: some of the light slightly penetrates the surface and is re-radiated uniformly in all directions. The light takes on some fraction of the color of the surface.
- **Specular** reflection: more mirror-like. Light is reflected directly from the object's outer surface, giving rise to highlights of approximately the same color as the source. The surface looks shiny.

17

17

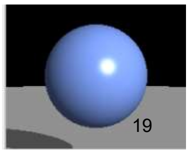


18

18

Shading Models: Introduction (4)

- In the simplest model, specular reflected light has the same color as the incident light. This tends to make the material look like plastic.
- In a more complex model, the color of the specular light varies over the highlight, providing a better approximation to the shininess of metal surfaces.



19

19

Shading Models: Introduction (5)

- Most surfaces produce some combination of diffuse and specular reflection, depending on surface characteristics such as roughness and type of material.
- The total light reflected from the surface in a certain direction is the sum of the diffuse component and the specular component.
  - For each surface point of interest, we compute the size of each component that reaches the eye.

20

20

Reflected Light Model

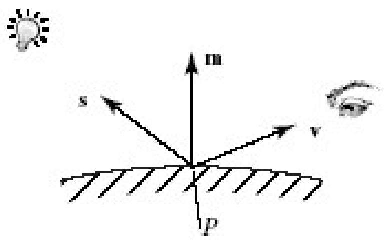
- Finding Reflected Light: a model
  - Model is not completely physically correct, but it provides fast and relatively good results on the screen.
  - Intensity of a light is related to its brightness. We will use I_s for intensity, where s is R or G or B.

21

21

Calculating Reflected Light

- To compute reflected light at point P, we need 3 vectors: normal **m** to the surface at P and vectors **s** from P to the source and **v** from P to the eye. We use world coordinates.

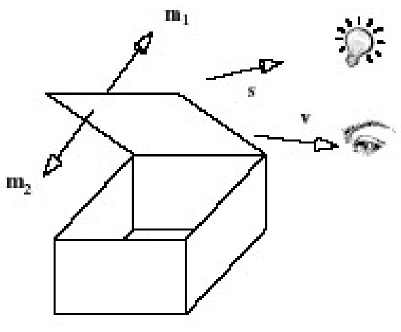


22

22

Calculating Reflected Light (2)

- Each face of a mesh object has an inside and an outside.
- Normally the eye sees only the outside (front, in Open-GL), and we calculate only light reflected from the outside.



23

23

Calculating Reflected Light (3)

- If the eye can see inside, we must also compute reflections from the inside (back, in OpenGL).
  - glLightModeli(GL_LIGHT_MODEL_TWO_ SIDES, GL_TRUE) calculates lighting for both front and back faces - in case you have open boxes, for example.

24

24

Calculating Diffuse Light

- Diffuse scattering is assumed to be independent of the direction from the point, *P*, to the location of the viewer's eye.

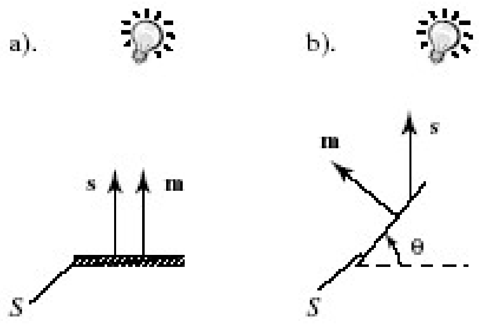
- The amount of light that illuminates the facet *does* depend on the orientation of the facet relative to the point source: the amount of light is proportional to the area of the facet that it sees: the area *subtended* by a facet.

25

25

Calculating Diffuse Light (2)

- The intensity depends on the projection of the face perpendicular to **s** (Lambert's law).
- Left :I_d; right: $I_d \cos\theta$



26

26

Calculating Diffuse Light (3)

- For ϑ near 0^o, brightness varies only slightly with angle, because the cosine changes slowly there. As ϑ approaches 90^o, the brightness falls rapidly to 0.
- We know $\cos \vartheta = \frac{\boldsymbol{s}.\boldsymbol{m}}{|\boldsymbol{s}||\boldsymbol{m}|}$
- $I_d = I_s \rho_d \frac{\boldsymbol{s}.\boldsymbol{m}}{|\boldsymbol{s}||\boldsymbol{m}|}$
  - I_s is the intensity of the source.
  - ρ_d is the diffuse reflection coefficient and depends on the material the object is made of.

27

27

Calculating Diffuse Light (4)

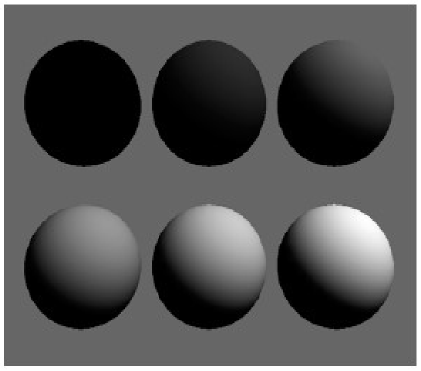
- $\mathbf{s} \cdot \mathbf{m} < 0$ implies $I_d = 0$.
- So to take all cases into account, we use
- $I_d = I_s \rho_d \max [(\mathbf{s} \cdot \mathbf{m})/(|\mathbf{s}||\mathbf{m}|), 0]$.

28

28

Example: Spheres Illuminated with Diffuse Light.

- Spheres with reflection coefficients from 0 to 1 by 0.2.
- The source intensity is 1.0. Background intensity is 0.4.
- The sphere is totally black when $\rho_d = 0.0$, and the shadow in its bottom half (where $(\mathbf{s}\cdot\mathbf{m})/(|\mathbf{s}||\mathbf{m}|) < 0$) is also black.



29

29

Calculating Diffuse Light (4)

- The light intensity falling on facet *S* from the point source is known to decrease as the inverse square of the distance between *S* and the source. We could try to incorporate this is our model.
- Experiments show that using this law yields pictures with exaggerated depth effects.
- Also, we sometimes model light sources as if they lie "at infinity". Using an inverse square law in such a case would quench the light entirely.
- So we ignore distance effects in calculating light intensity.

30

30

Calculating the Specular Component

- Real objects do not scatter light uniformly in all directions; a specular component is added to the shading model.
- Specular reflection causes highlights, which can add significantly to realism of a picture when objects are shiny.

31

31

Calculating the Specular Component (2)

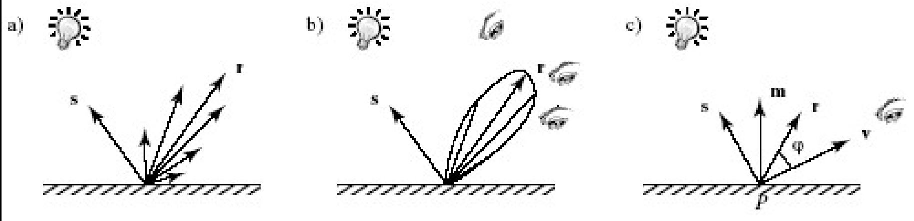
- A simple model for specular light was developed by Phong. It is easy to apply.
  - The highlights generated by the Phong model give an object a plastic-like or glass-like appearance.
  - The Phong model is less successful with objects that are supposed to have a shiny metallic surface, although you can roughly approximate them with OpenGL by careful choices of certain color parameters.

32

32

Calculating the Specular Component (3)

- Most of the light reflects at equal angles from the (smooth and/or shiny) surface, along direction **r,** the reflected direction.



33

33

Calculating the Specular Component (2)

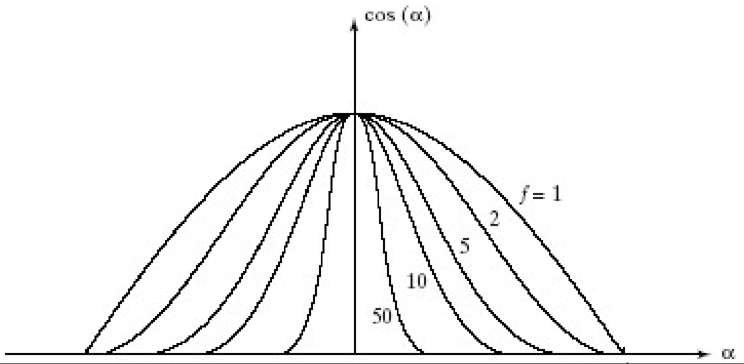
- We found that **r** = -**s** + 2 **m** (**s**·**m**)/(|**m**|^2) (mirror reflection direction).
- For surfaces that are not mirrors, the amount of reflected light decreases as the angle ϕ between **r** and **v** (vector from reflection point to eye) increases.
- For a simplified model, we say the intensity decreases as $\cos^f$ ϕ, where f is chosen experimentally between 1 and 200.

34

34

Calculating the Specular Component (3)

- The effect of f : large f values give concentrated highlights; smaller ones give larger dimmer highlights.



35

35

Calculating the Specular Component (4)

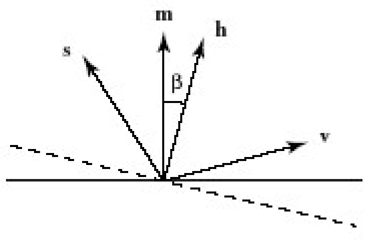
- cos ϕ = **r·v**/(|**r**||**v**|)
- $I_{sp} = I_s \rho_s (\mathbf{r}\cdot\mathbf{v}/(|\mathbf{r}||\mathbf{v}|))^f$.
  - ρ_s is the specular reflection coefficient, which depends on the material.
- If **r·v** < 0, there is no reflected specular light.
- $I_{sp} = I_s \rho_s \max[(\mathbf{r}\cdot\mathbf{v}/(|\mathbf{r}||\mathbf{v}|))^f, 0]$.

36

36

Speeding up Calculations for Specular Light

- Find the halfway vector **h** = **s** + **v**.
- Then the angle β between **h** and **m** approximately measures the falloff of intensity. To take care of errors, we use a different f value, and write



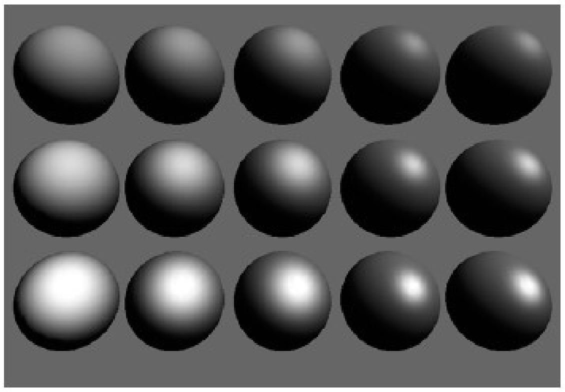
$I_{sp} = I_s \rho_s \max[(\mathbf{h}\cdot\mathbf{m}/(|\mathbf{h}||\mathbf{m}|))$

37

37

Calculating the Specular Component (5)

- From bottom to top, ρ_s = 0.75, 0.5, 0.25. From left to right, f = 3, 6, 9, 25, 200.
- ρ_a= 0.1
- ρ_d= 0.4

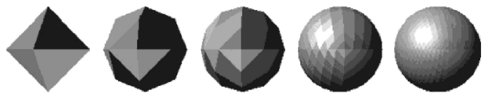


38

38

FLAT SHADING

- Flat shading works by applying a single pixel colour across an entire polygon.
  - It cannot handle specular reflection.
- Very quick and efficient, but realism is limited.
  - Especially for low polygon counts.
  - Separate polygons are clearly visible.



39

FLAT SHADING

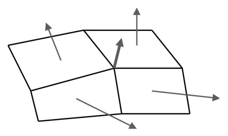
- The human eye is especially good at noticing edges.
  - Flat shading is thus acting against our biology.
- Better results can be obtained by interpolation of shading across a polygon's surface.
  - Common technique for this is Gouraud Shading.
  - Used when polygons are approximating curved surfaces.
  - This provides a linear colour gradient over the polygon.

40

GOURAUD SHADING

First, must calculate vertex intensity.

- Simple method is to average the light intensity of all polygons sharing a vertex.
- More complex method involves modelling light interaction at each vertex.
  - More computation, but more realistic output.

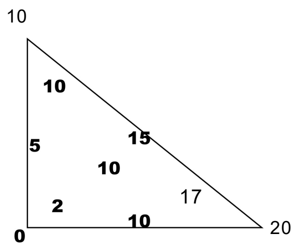


The arrows indicate suNace normals

41

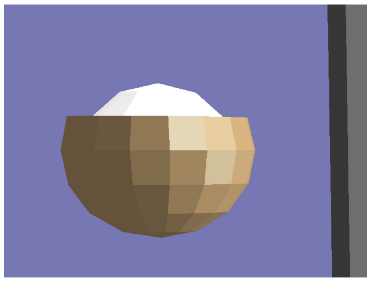
GOURAUD SHADING

Light intensity at each vertex used to calculate light intensity of pixels in polygon.

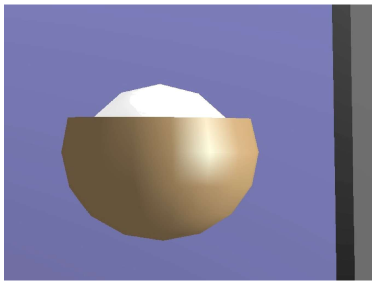


42

FLAT VERSUS GOURAUD



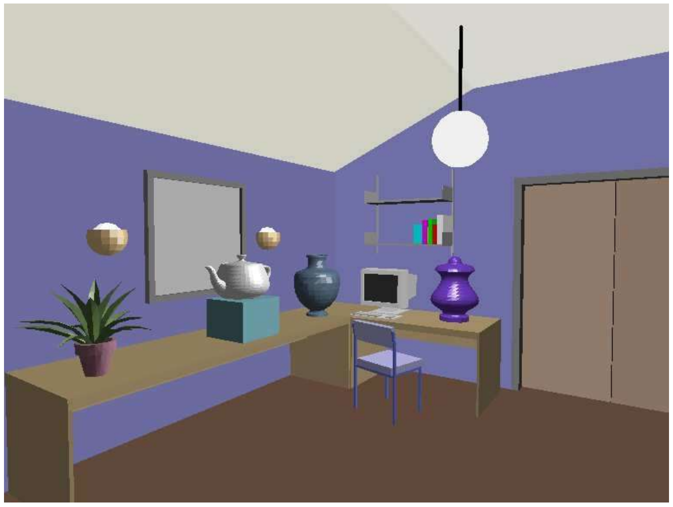
Flat Shading Model —note individual polygons easily identifiable.



Gouraud Shading Model — interpolation of pixel colours across polygons hides edges better.

43

FLAT SHADING



44

GOURAUD SHADING

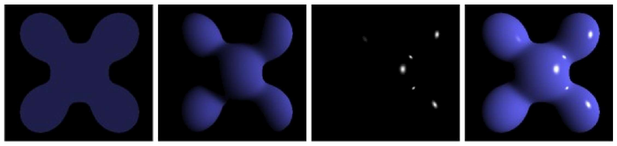
45

PHONG REFLECTION MODEL

The Phong Rehection Model works by estimating the colours of pixels.

Light described as the combination of:

- Ambient light
- Diffuse light
- Specular Light



46

PHONG SHADING

Problem of visible edges mitigated by Gouraud shading

- Noteliminated
- Minimum and maximum intensity will always occur at vertexes.

Calculation works on the basis of interpolation.

- Interpolation works slightly differently.

47

SURFACE NORMALS

When determining the way light interacts with a polygon, we base it on the surface normal.

- A hypothetical line that extends perpendicularly from the point.

With hat shading, we base the intersection on the surface normal of the middle pixel of a polygon.

- This gives a rough measure of light intensity.

With Gouraud, we base it on the intensity of each vertex.

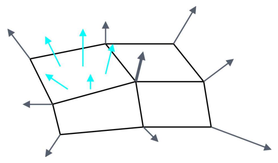
- More nuanced.

48

PHONG SHADING

Phong Shading interpolates the surface normals across a polygon.

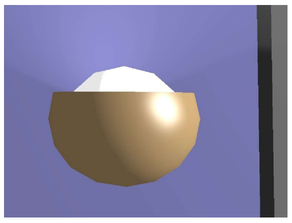
Intensity is estimated based on these interpolated normals.



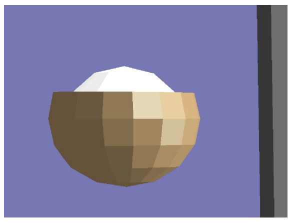
The arrows indicate suNace normals

49

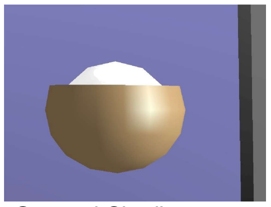
SHADING MODELS



Phong Shading



Flat Shading



Gouraud Shading

50

Ambient Light

- Our desire for a simple reflection model leaves us with far from perfect renderings of a scene.
  - E.g., shadows appear to be unrealistically deep and harsh.
- To soften these shadows, we can add a third light component called *ambient light*.

51

51

Ambient Light (2)

- The scenes we observe around us always seem to be bathed in some soft non-directional light.
- This light arrives by multiple reflections from various objects in the surroundings and from light sources that populate the environment, such as light coming through a window, fluorescent lamps, and the like.
- We assume a uniform background glow called **ambient light** exists in the environment.
- This ambient light source is not situated at any particular place, and it spreads in all directions uniformly.

52

52

Calculating Ambient Light

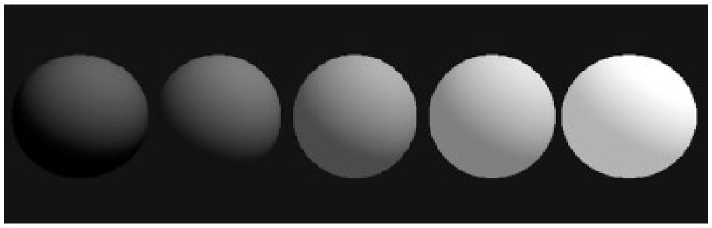
- The source is assigned an intensity, I_a.
- Each face in the model is assigned a value for its **ambient reflection coefficient**, ρ_a (often this is the same as the diffuse reflection coefficient, ρ_d), and the term $I_a \rho_a$ is simply added to whatever diffuse and specular light is reaching the eye from each point *P* on that face.
- I_a and ρ_a are usually arrived at experimentally, by trying various values and seeing what looks best.

53

53

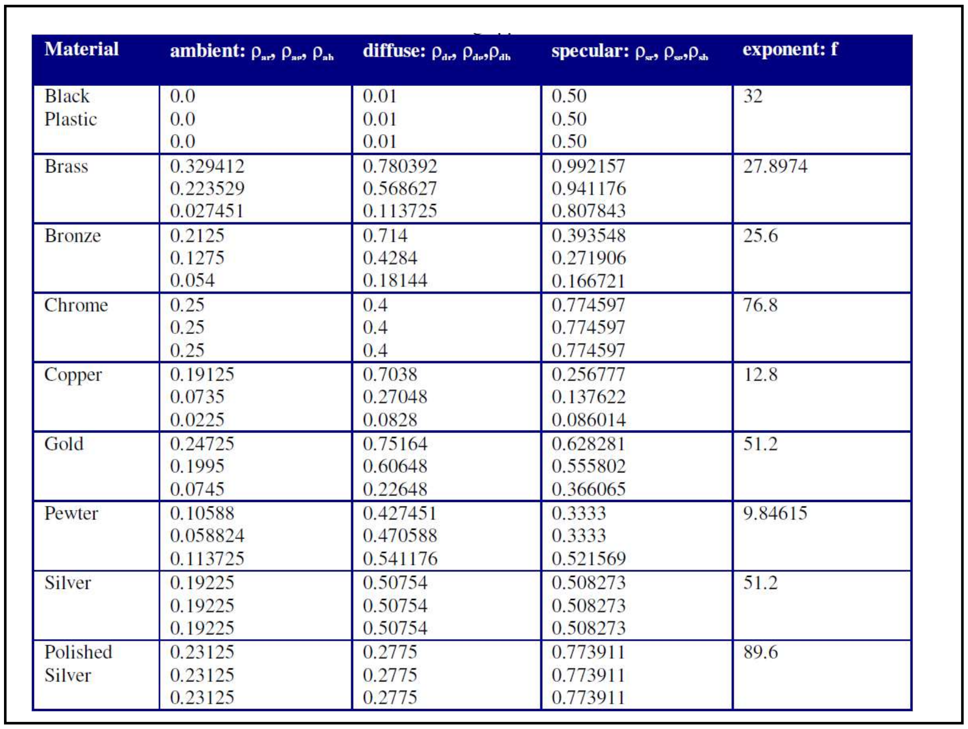
Adding Ambient Light to Diffuse Reflection

- The diffuse and ambient sources have intensity 1.0, and $\rho_d = 0.4$. $\rho_a = 0, 0.1, 0.3, 0.5, 0.7$ (left to right).
- Modest ambient light softens shadows; too much ambient light washes out shadows.



54

54



Material	ambient: $\rho_{ar}, \rho_{ag}, \rho_{ab}$	diffuse: $\rho_{dr}, \rho_{dg}, \rho_{db}$	specular: $\rho_{sr}, \rho_{sg}, \rho_{sb}$	exponent: f
Black Plastic	0.0 0.0 0.0	0.01 0.01 0.01	0.50 0.50 0.50	32
Brass	0.329412 0.223529 0.027451	0.780392 0.568627 0.113725	0.992157 0.941176 0.807843	27.8974
Bronze	0.2125 0.1275 0.054	0.714 0.4284 0.18144	0.393548 0.271906 0.166721	25.6
Chrome	0.25 0.25 0.25	0.4 0.4 0.4	0.774597 0.774597 0.774597	76.8
Copper	0.19125 0.0735 0.0225	0.7038 0.27048 0.0828	0.256777 0.137622 0.086014	12.8
Gold	0.24725 0.1995 0.0745	0.75164 0.60648 0.22648	0.628281 0.555802 0.366065	51.2
Pewter	0.10588 0.058824 0.113725	0.427451 0.470588 0.541176	0.3333 0.3333 0.521569	9.84615
Silver	0.19225 0.19225 0.19225	0.50754 0.50754 0.50754	0.508273 0.508273 0.508273	51.2
Polished Silver	0.23125 0.23125 0.23125	0.2775 0.2775 0.2775	0.773911 0.773911 0.773911	89.6

55