UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

Modelos de Iluminação

Disciplina: Computação Gráfica (BCC35F)
Curso: Ciência da Computação

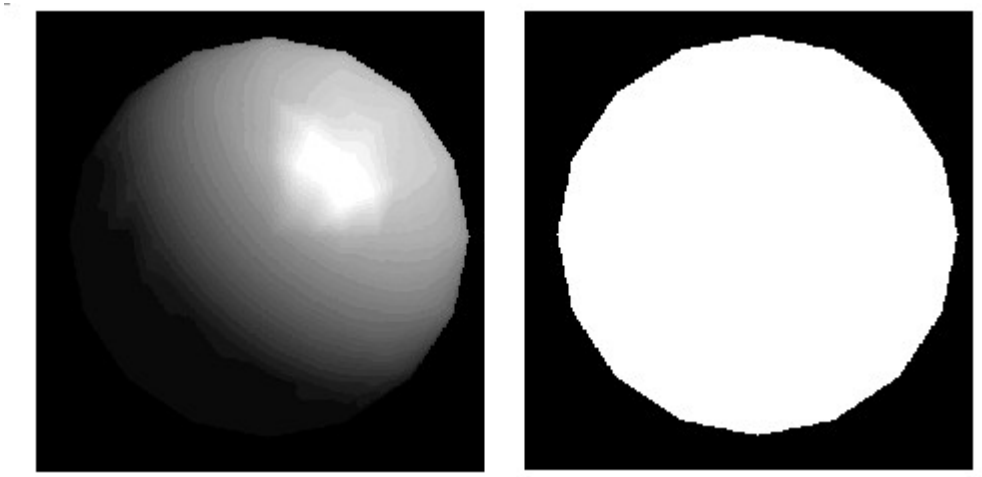
Prof. Walter T. Nakamura
waltertakashi@utfpr.edu.br

Campo Mourão - PR

Baseados nos materiais elaborados pelas professoras Aretha Alencar (UTFPR) e Rosane Minghim (USP)

Introdução

- Imagens **realísticas** são criadas usando **projeções perspectivas**, aplicando-se efeitos de iluminação natural às superfícies visíveis por meio de um **modelo de iluminação** (*shading model*)
- Modelos de iluminação são usados para calcular a cor de uma posição iluminada na superfície do objeto
- De forma geral, modelar os **efeitos da luz** sobre um objeto é um processo **complexo**, que envolve princípios físicos

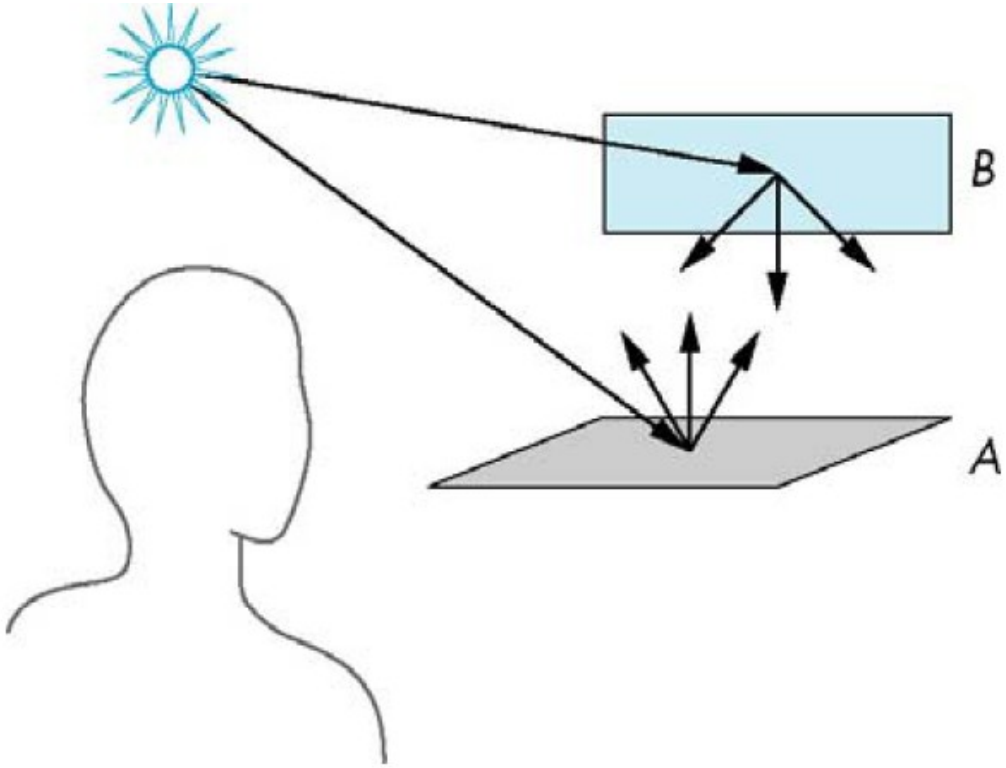


Introdução

- Os **modelos físicos** envolvem vários fatores, como **propriedades** dos materiais, posição do objeto em relação a luz e outros objetos, e características das fontes de luz:
  - Objetos podem ser opacos ou mais ou menos transparentes, podem ser finos ou grosseiros
  - Fontes de luz podem ter vários formatos, cores e posições
- Os **modelos de iluminação** em computação gráfica são na maioria das vezes **aproximações** das leis físicas que descrevem efeitos de luz sobre superfícies

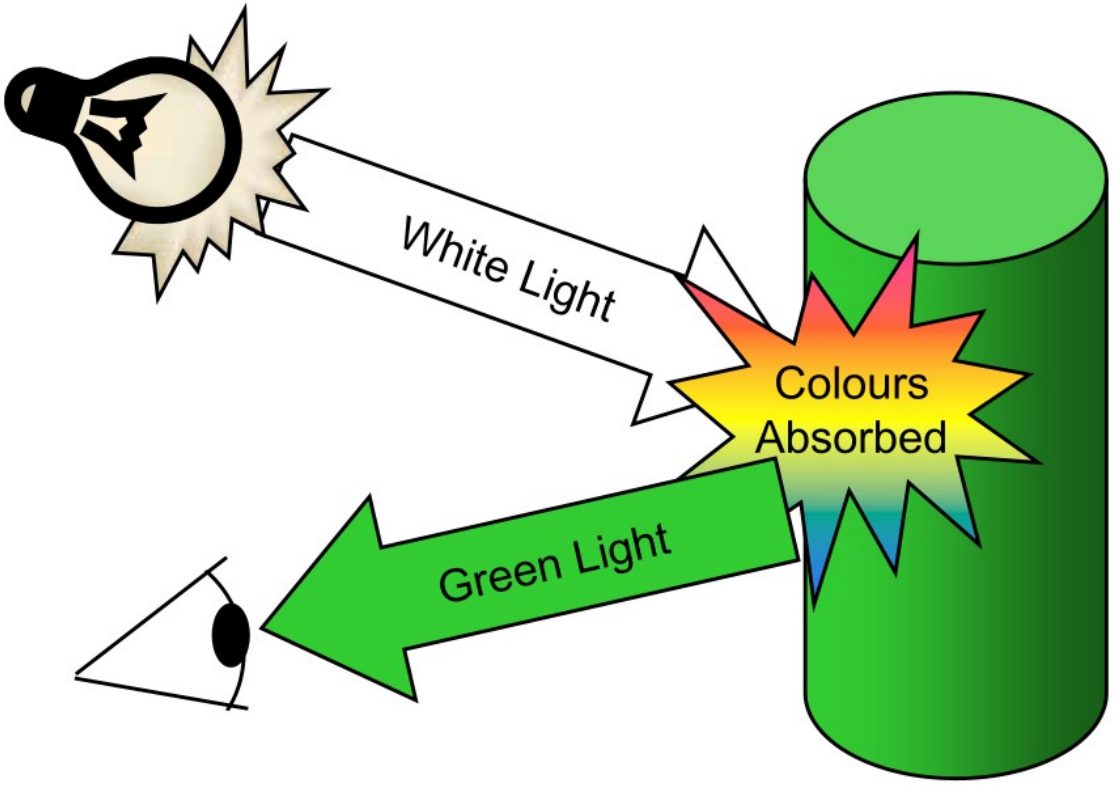
Realidade Física

- Quando a luz atinge uma superfície
  - Parte é absorvida
  - Parte é refletida
- Cores visíveis incluem todas as cores de um arco-íris



Realidade Física

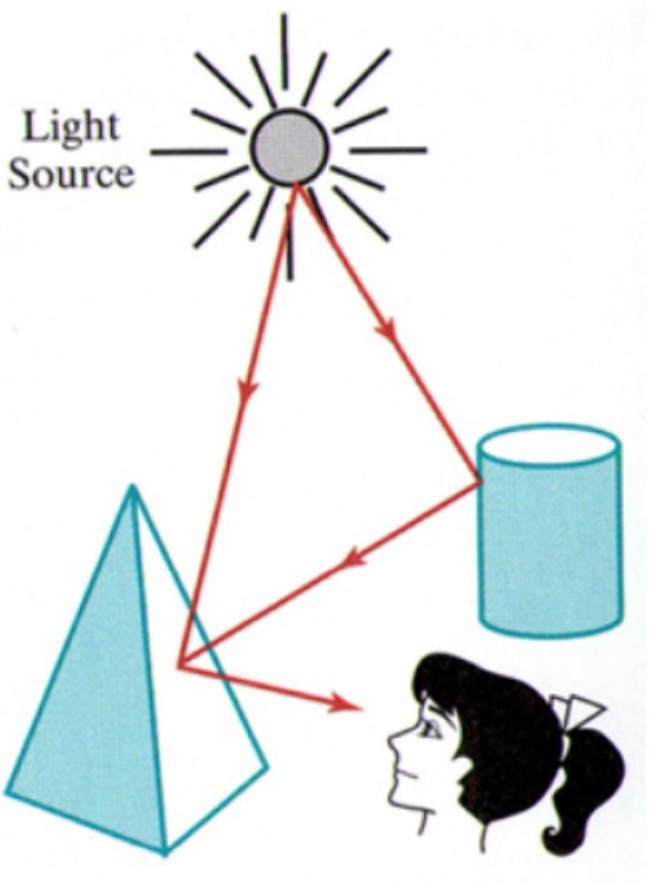
- Fisicamente, pode-se dizer que os objetos possuem a cor da luz que emana de suas superfícies



- Se um objeto azul é iluminado por uma luz vermelha, ele aparecerá na cor_____?

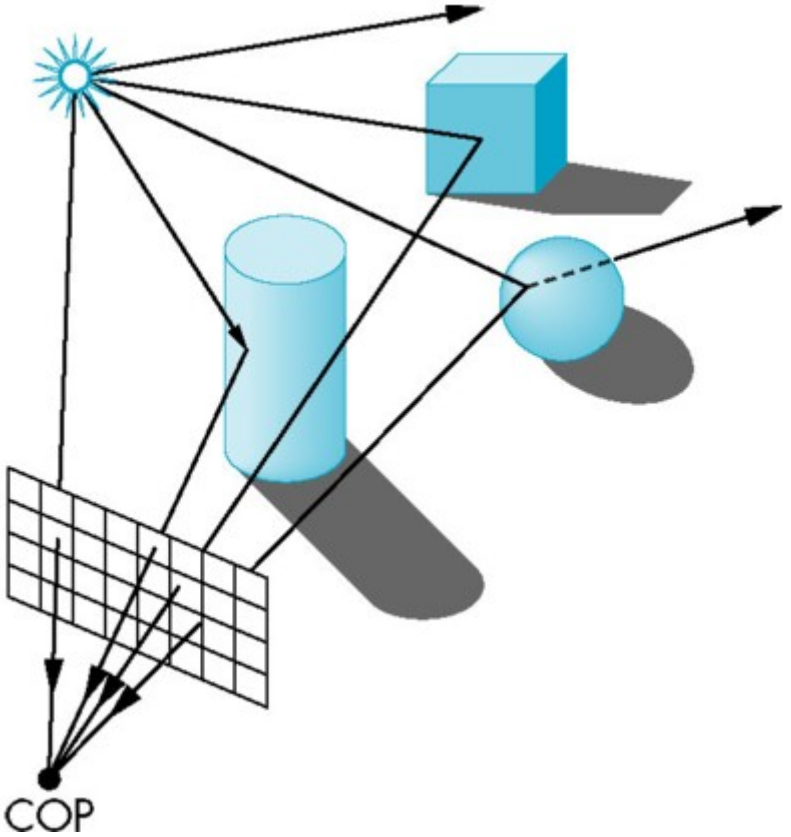
Modelo de Iluminação

- Segue os raios a partir da fonte de luz
- Somente as luzes que alcançam os olhos são vistos
  - Luz direta é vista na cor da fonte de luz
  - Luz indireta depende da interação entre o material e a luz



Iluminação na Computação Gráfica

- Substituímos o observador por um plano de projeção
- Raios que alcançam o centro de projeção (COP) após passar pelo plano de visão são visualizados
- A cor dos pixels é determinada pelo modelo de interação

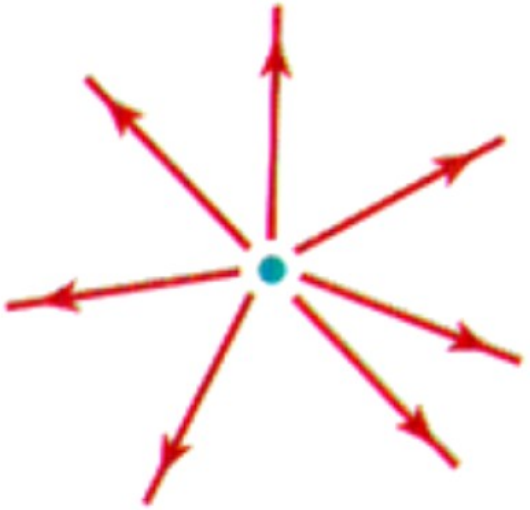


Fontes de Luz

- Qualquer objeto que emite energia brilhante é uma **fonte de luz** que contribui para os efeitos de luz dos outros objetos na cena
- **Fontes de luz** podem ter diferentes formas e características (posição, cor, direção de emissão, formato, etc.)
- Em aplicações gráficas de **tempo real**, um modelo simples de iluminação normalmente é aplicado por causa do **custo computacional**
  - Propriedades da emissão de luz são definidas usando **valores distintos** para cada componente de cor RGB, descrevendo suas intensidades

Fontes de Luz Pontuais

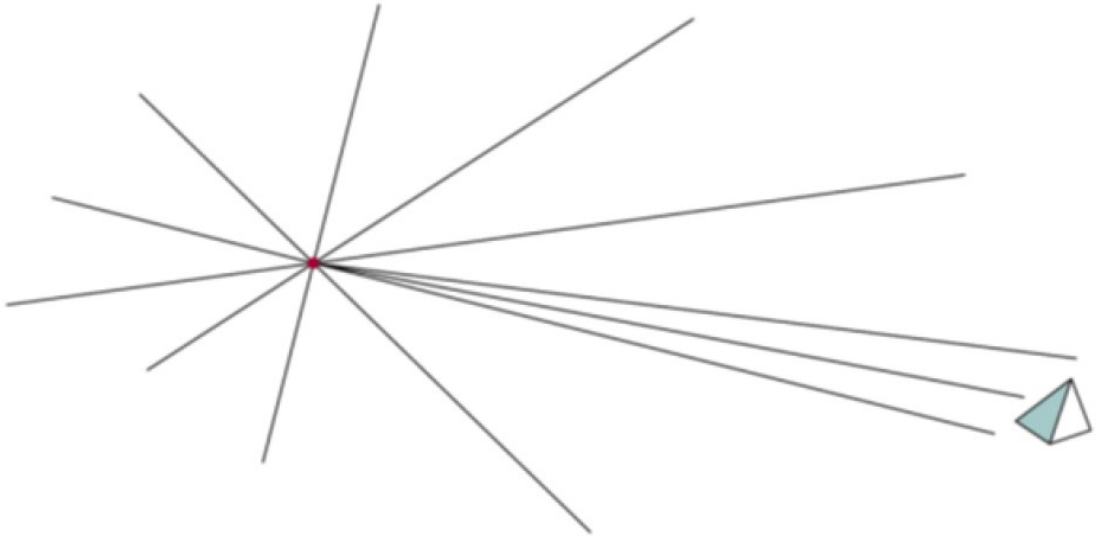
- O modelo mais simples de emissão de luz é **fonte de luz pontual** com uma única cor:
  - Definida por uma **posição** e a **cor** da luz emitida:



- Os raios de luz são gerados em **direções radiais divergentes** a partir do ponto de luz:
  - Indicado para aproximar efeitos de luz quando a **fonte de luz** é pequena em comparação com os objetos da cena.

Fontes de Luz Infinitamente Distantes

- Uma **fonte de luz grande** (ex: o Sol), que está bem longe da cena, pode ser aproximada como um ponto emissor bem distante dos objetos
  - A iluminação é provida em uma **única direção**:

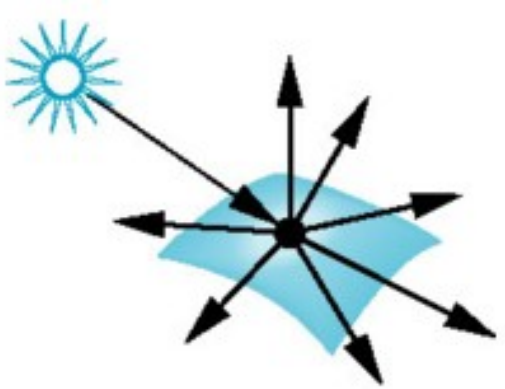


- Uma fonte de luz distante é simulada definindo sua **cor** e uma **direção** da emissão dos raios, não é necessário especificar uma posição

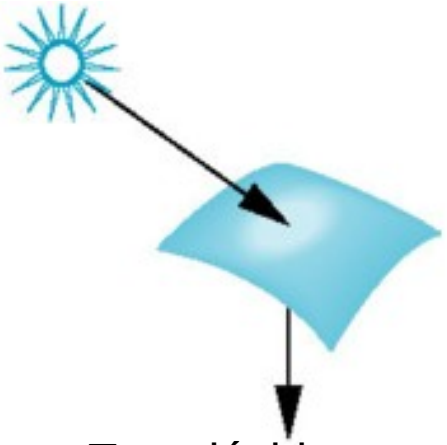
Interação Luz-Material

- Determina se um objeto aparecerá vermelho ou marrom, claro ou escuro, opaco ou brilhoso
- A reflexão da luz em uma superfície pode ser:

Especular



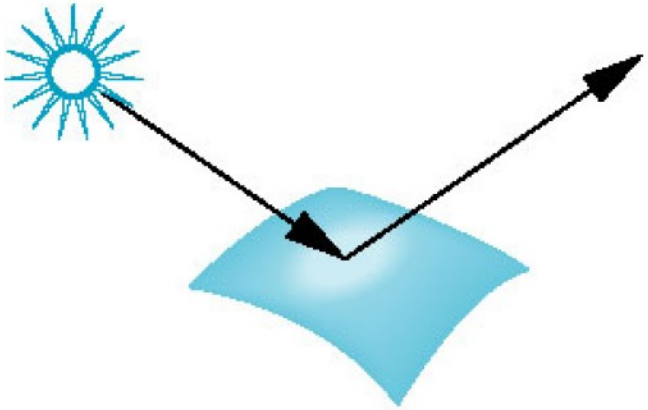
Difusa



Translúcida

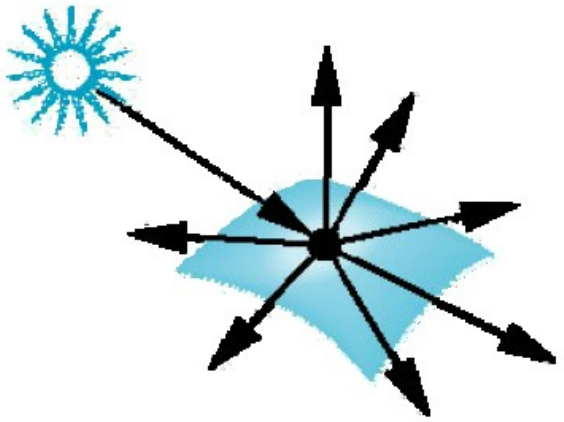
Superfícies Especulares

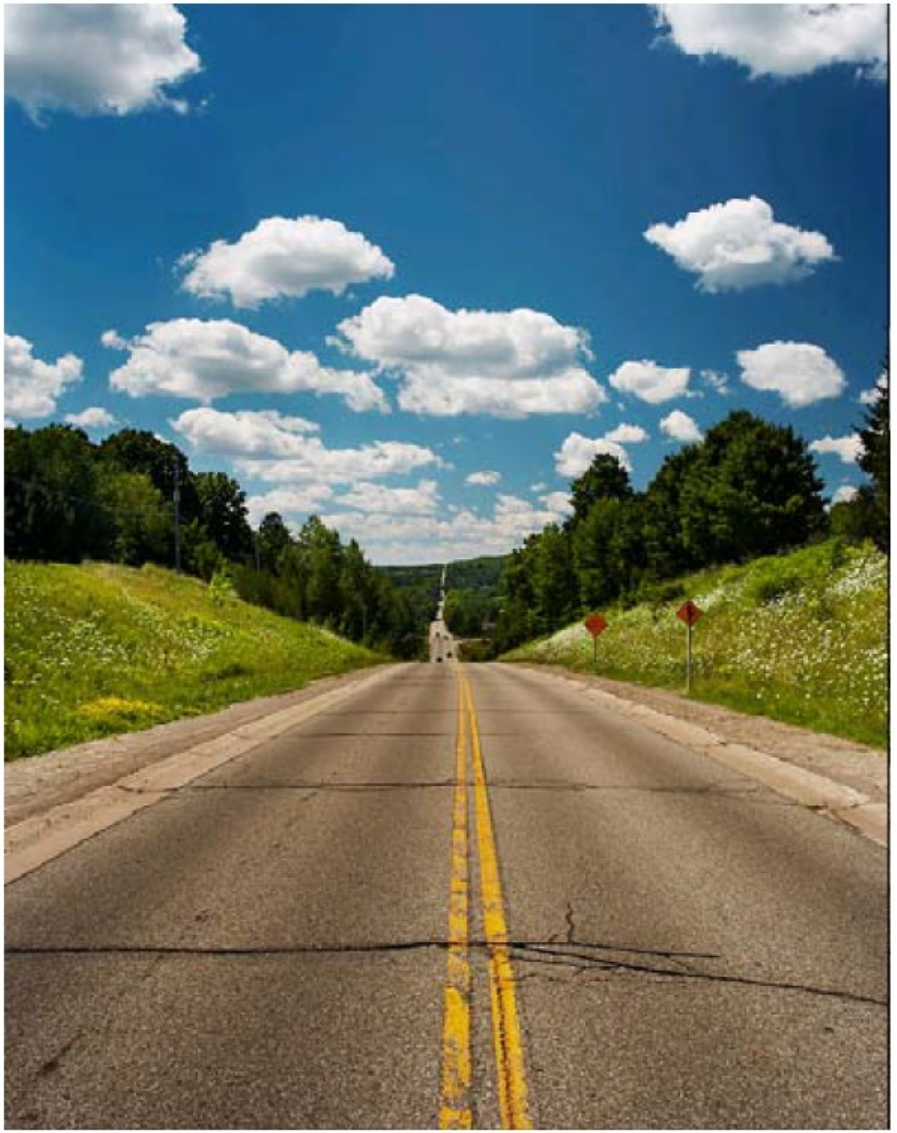
- Suave, brilhosa, lustroso
- É como um espelho
- Reflexão focada em uma direção
  - Ângulo de incidência = ângulo de reflexão



Superfícies Difusas

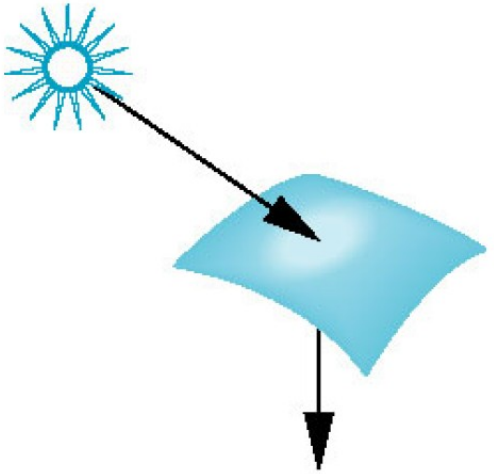
- Áspero (granulado, fosco)
- A luz é refletiva quase uniformemente em todas as direções





Superfícies Translúcidas

- Permite que parte da luz atravesse
- Similar a um vidro ou à água



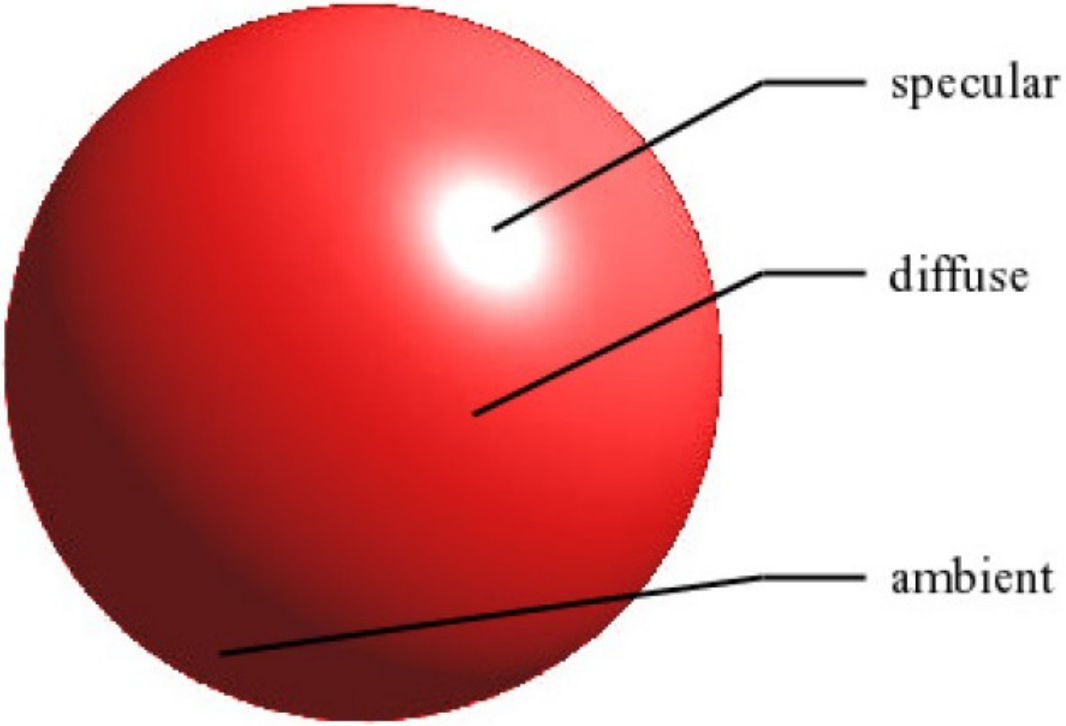
Resumo da Interação Luz-Material

- Superfícies podem ser
  - Especulares
  - Difusas
  - Translúcidas
- Uma superfície nunca é perfeitamente difusa, perfeitamente translúcida ou perfeitamente especular
- Pode ter todas essas propriedades
- Para modelar essas propriedades, nós devemos compreender como a luz é refletida em cada tipo de superfície

Modelo Básico de Iluminação

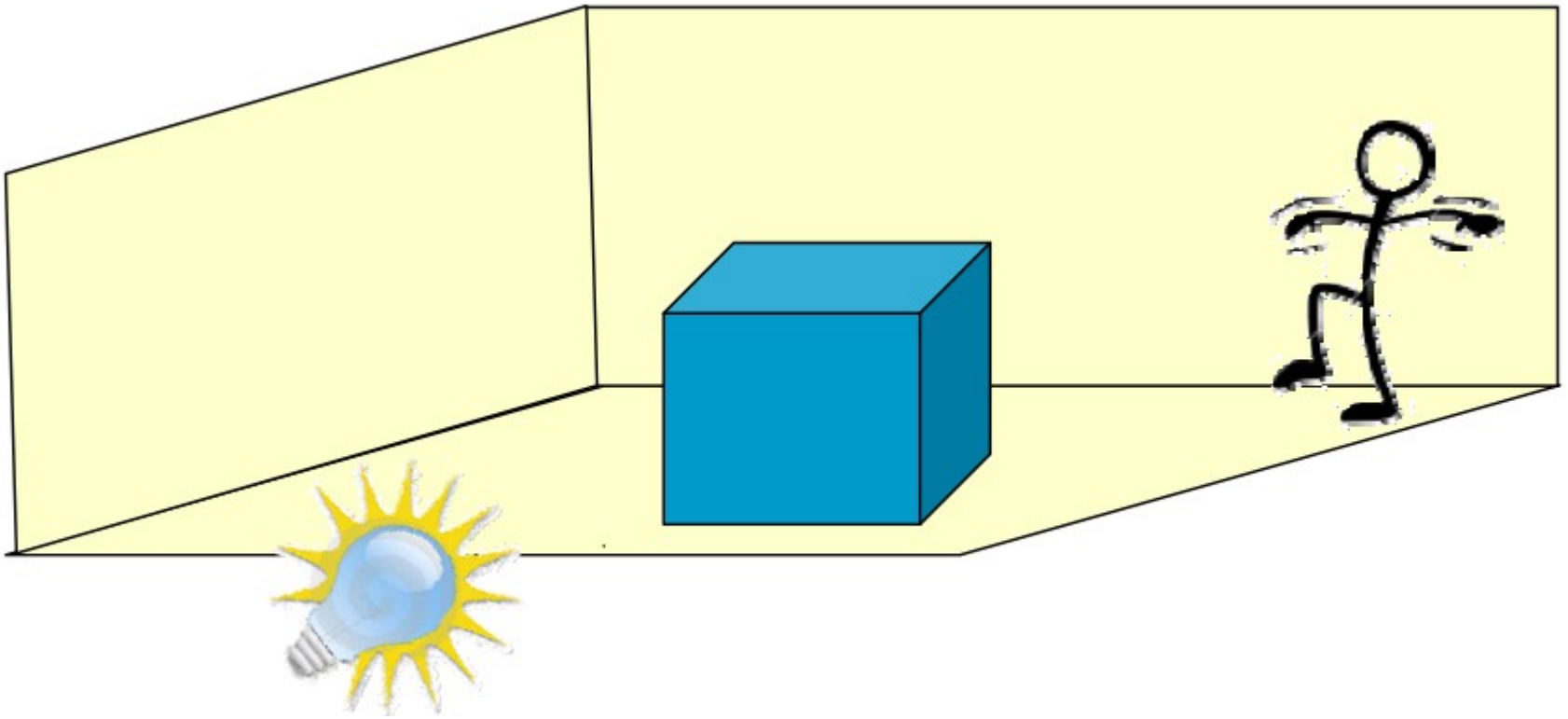
- Nós consideraremos um modelo de iluminação básico que fornece re sultados razoavelmente bons e é usado na maioria dos sistemas gráficos
- Os componentes importantes são:
  - Reflexão difusa
  - Reflexão especular
  - Luz ambiente

Exemplo



A luz total refletida pela superfície é a soma das contribuições das fontes de luz e a luz refletida

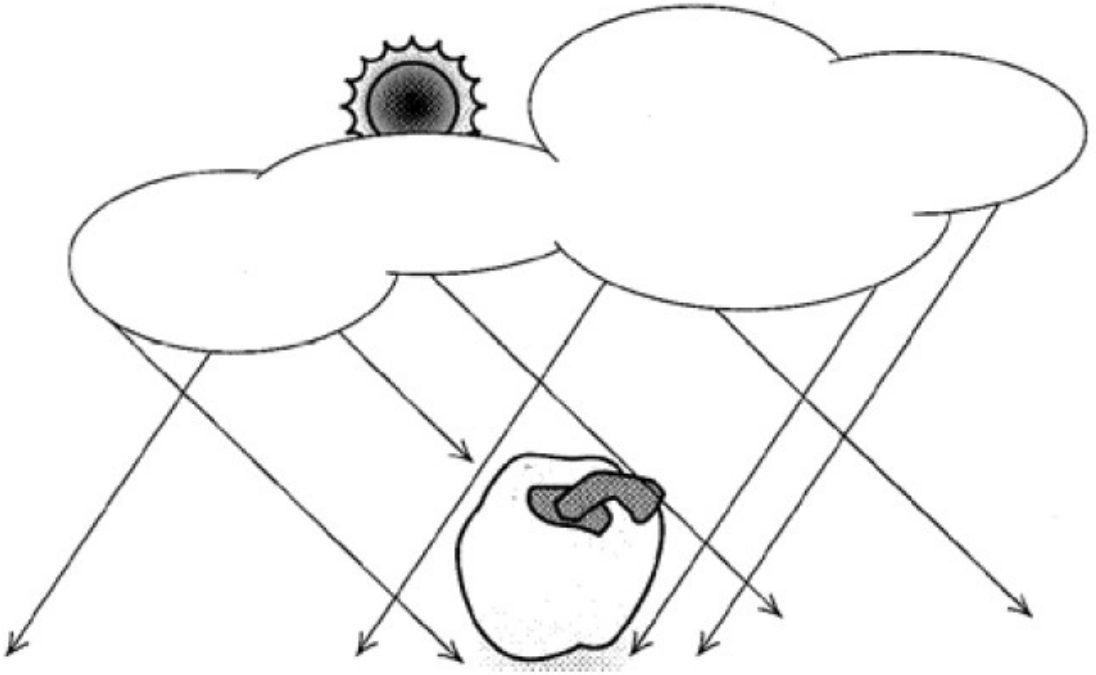
Luz Ambiente



- Ande ao redor de uma caixa preta. Nenhum raio de luz bate diretamente. Você conseguiria ver a face de trás?
- Esse tipo de luz indireta é chamada de luz ambiente

Luz Ambiente

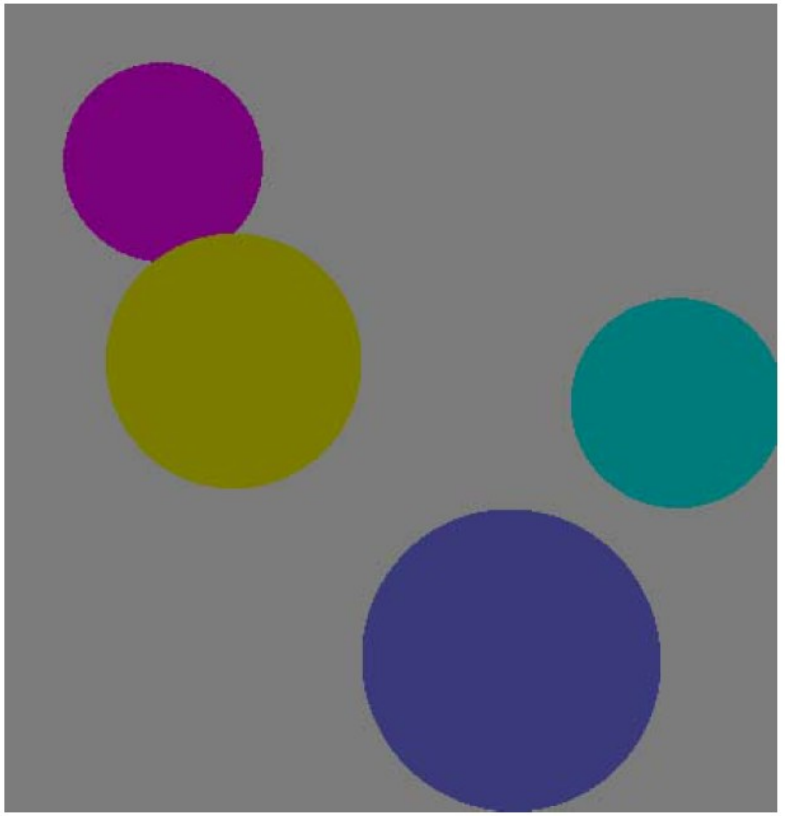
- É como a luz do Sol em dia totalmente nublado



- Nível de luz uniforme e independente da posição
- Determina o quão brilhoso um objeto parece quando nenhuma fonte de luz pode alcançá-lo diretamente

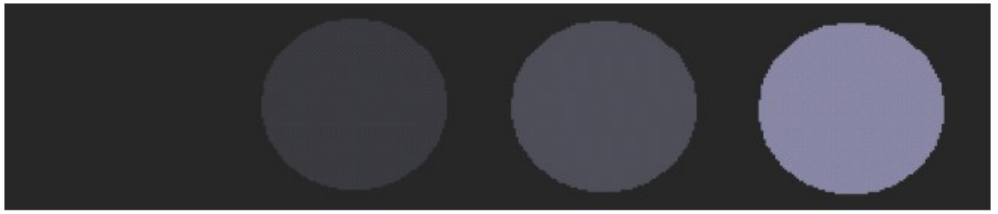
Luz Ambiente

- Também chamada luz de fundo
- Não criada por nenhuma fonte de luz
- Uma iluminação constante em todas as direções
- Contribui por luz dispersa em um ambiente
- Quando usado sozinho, não produz imagens muito interessantes



Luz Ambiente

- Para incorporar uma luz de fundo, nós definimos um nível de brilho geral I_a para uma cena



- Diferentes superfícies podem refletir uma quantidade diferente de luz ambiente de acordo com suas propriedades reflexivas
  - Nós modelamos isso por um fator constante para cada superfície:

$$k_a \times I_a$$

Considerações sobre Cores RGB

- A intensidade de cada cor RGB é especificada por um vetor de três elementos

$$I_a = (I_{aR}, I_{aG}, I_{aB})$$

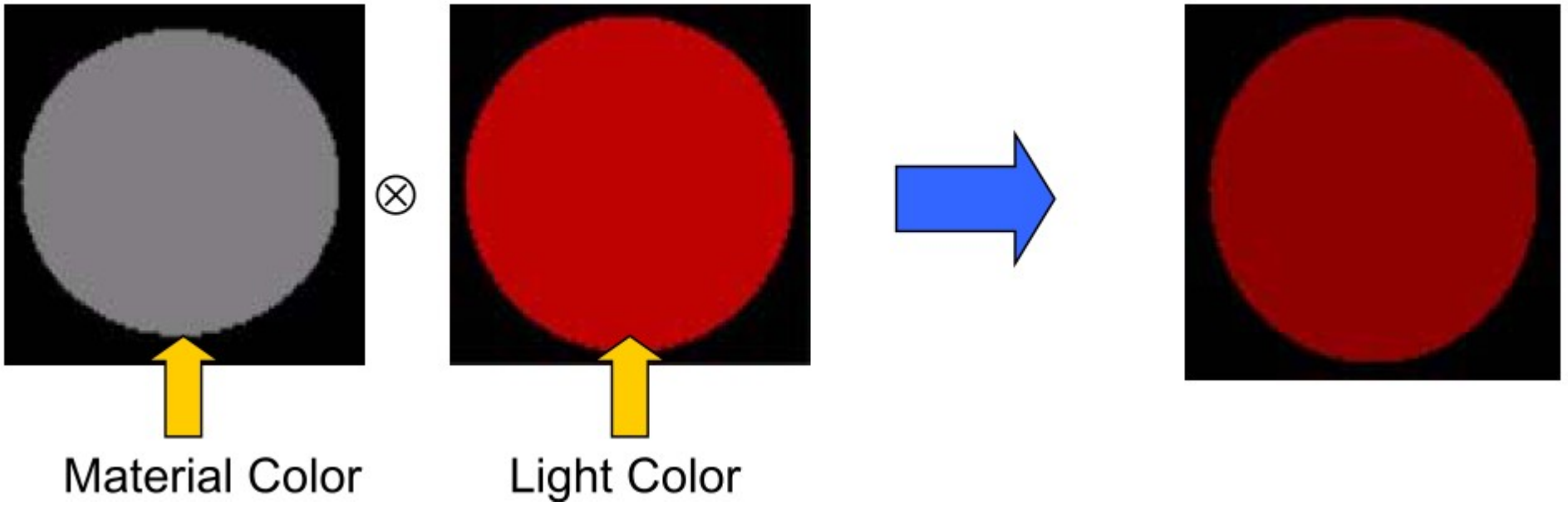
- Similarmente, a reflexão também é dada como um vetor:

$$k_a = (k_{aR}, k_{aG}, k_{aB})$$

- As intensidades de luz ambiente das cores RGB são:

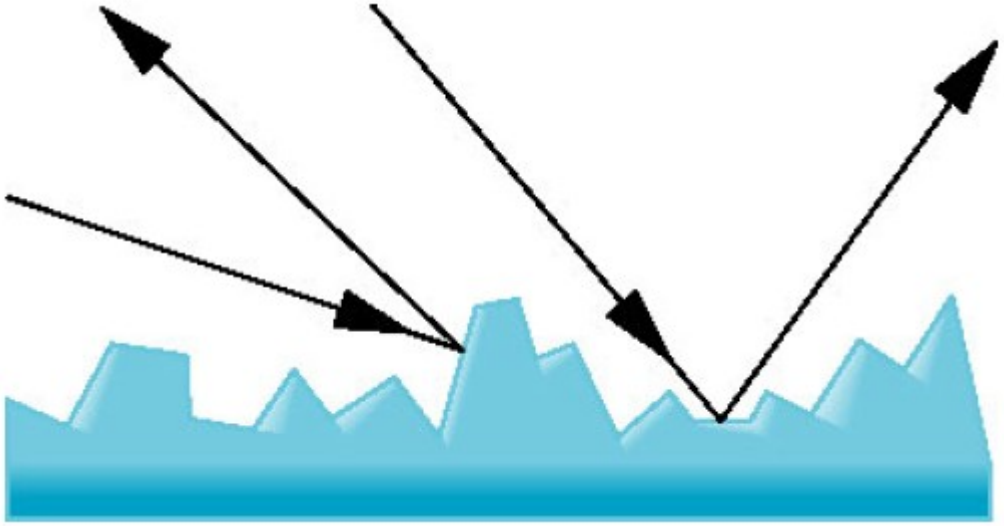
$$(k_{aR} I_{aR}, k_{aG} I_{aG}, k_{aB} I_{aB})$$

Iluminação: Luz Ambiente



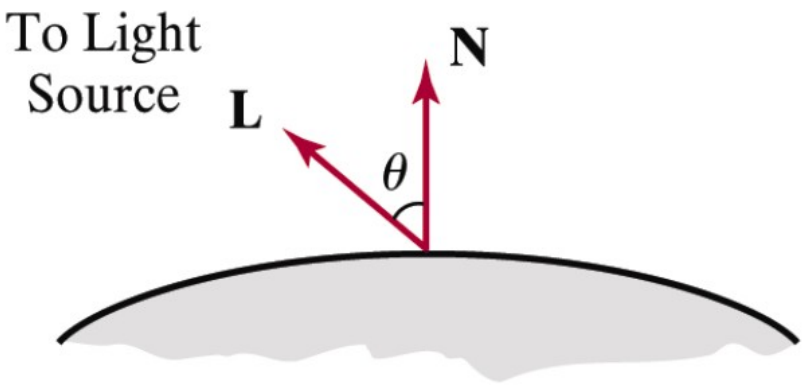
Reflexão Difusa

- Luz espalhada com igual intensidade em todas as direções (reflexão difusa ideal)
- Luz de um ponto é independente da direção de visão (igualmente brilhante em todas as direções)



Reflexão Difusa

- O ângulo entre a direção da luz de entrada e a normal de uma superfície é referido como o ângulo de incidência, denotado θ

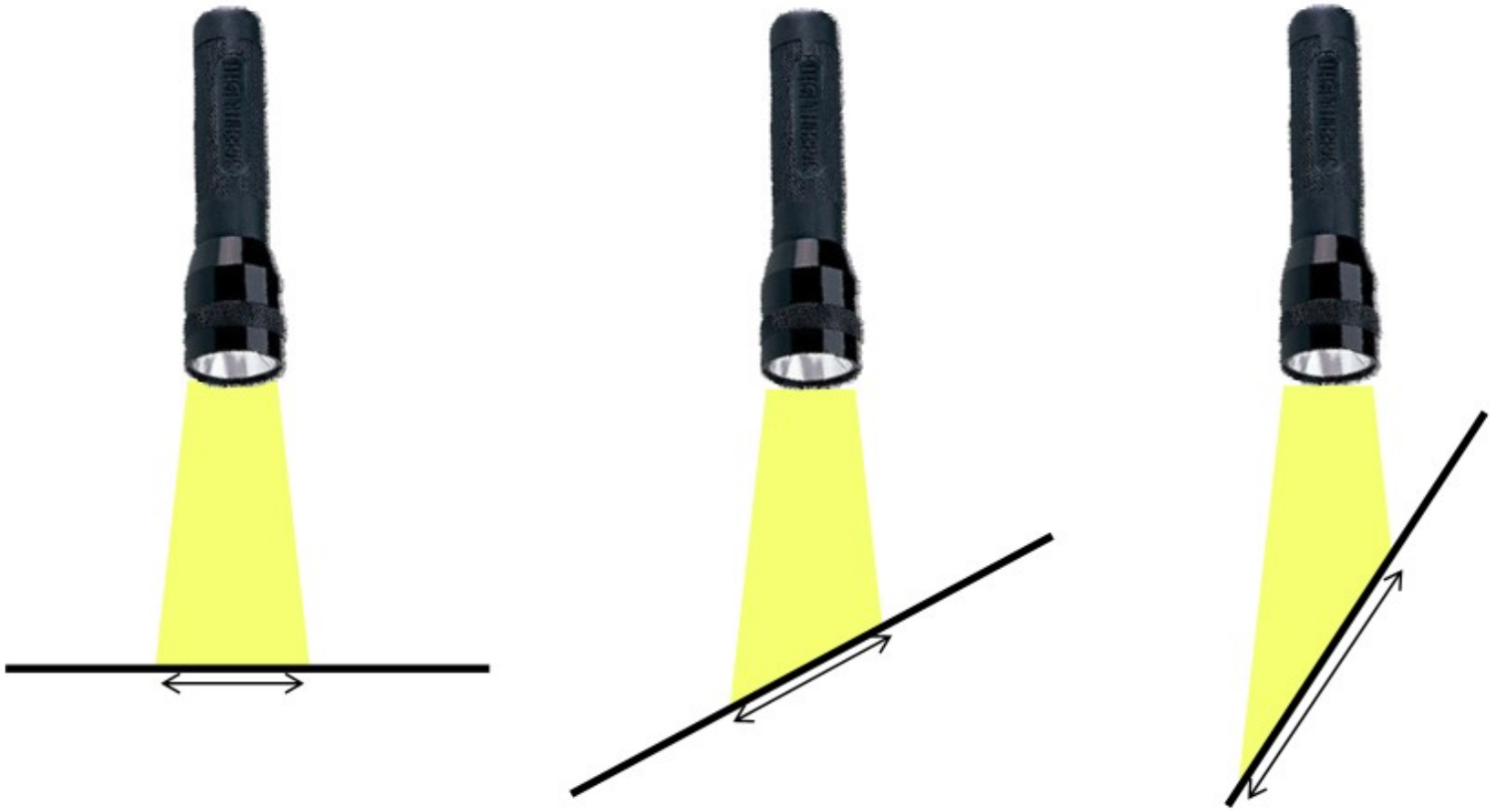


L = unit vector
to light source

N = unit vector
normal to surface

- Lei da reflexão: o ângulo de incidência é igual ao ângulo de reflexão e as direções de L, N e R (reflexão) são co-planares

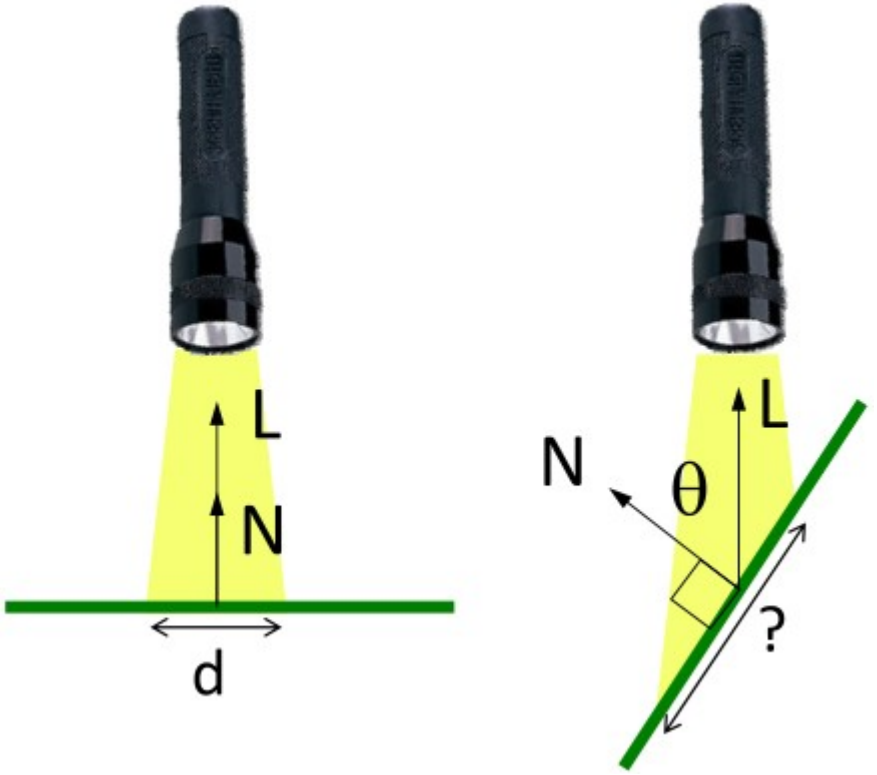
Reflexão Difusa



- A quantidade de luz incidente depende da orientação da superfície relativa à direção da fonte de luz

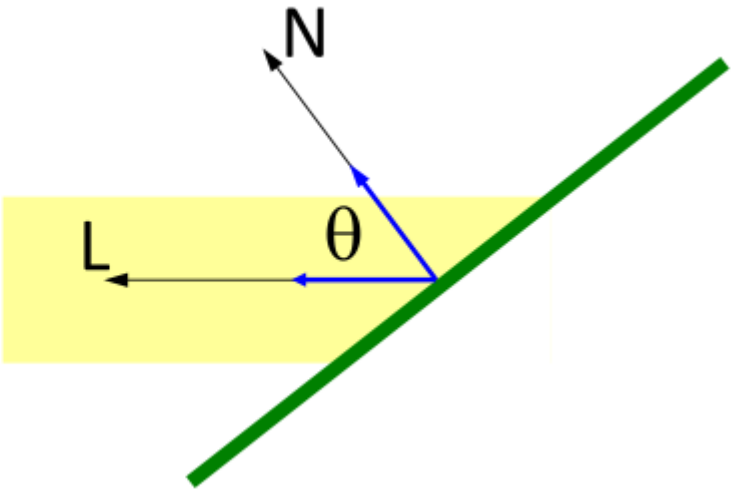
Reflexão Difusa

- A quantidade de luz incidente sobre uma superfície depende do ângulo de incidência
  - À medida que θ aumenta, o brilho da superfície diminui



Relembrando: Produto Escalar

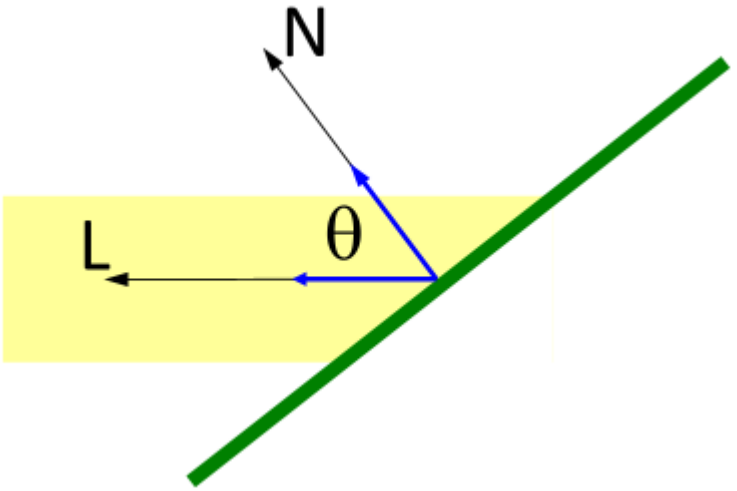
$$N \cdot L = |N||L| \cos \theta$$



- Se N e L são vetores unitários, então $N \cdot L = cos\ \theta$

Reflexão Difusa

- Se a superfície tem brilho I ao ser atingido pela luz, ele tem brilho $I \times cos(\theta)$ ao ser atingido em um ângulo θ

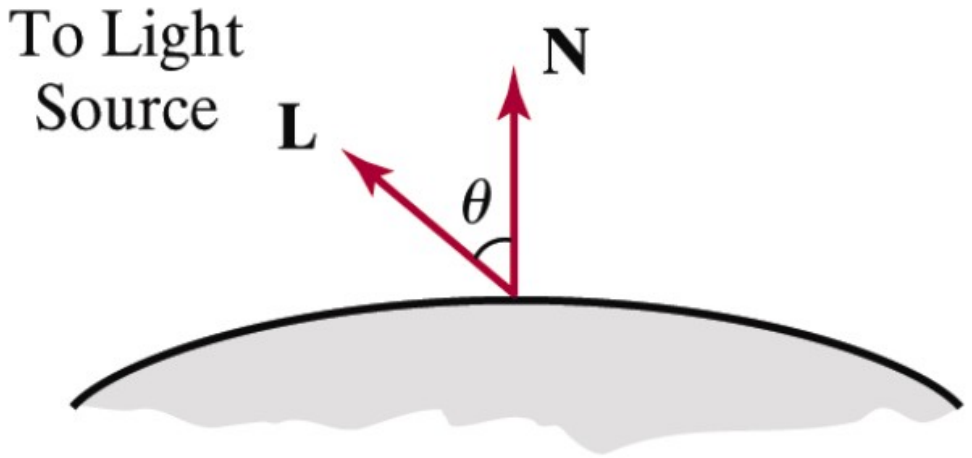


$N \cdot L = \cos \theta$

Reflexão Difusa

- Um parâmetro k_d definido para cada superfície determina a fração da luz incidente dispersa como reflexões difusas a partir daquela superfície
- Esse parâmetro é conhecido como coeficiente de reflexão difusa ou refletividade difusa
- k_d é atribuído um valor entre 0.0 e 1.0
  - 0.0 para superfície opaca que absorve quase toda luz
  - 1.0 para superfície brilhante que reflete quase toda luz

Reflexão Difusa

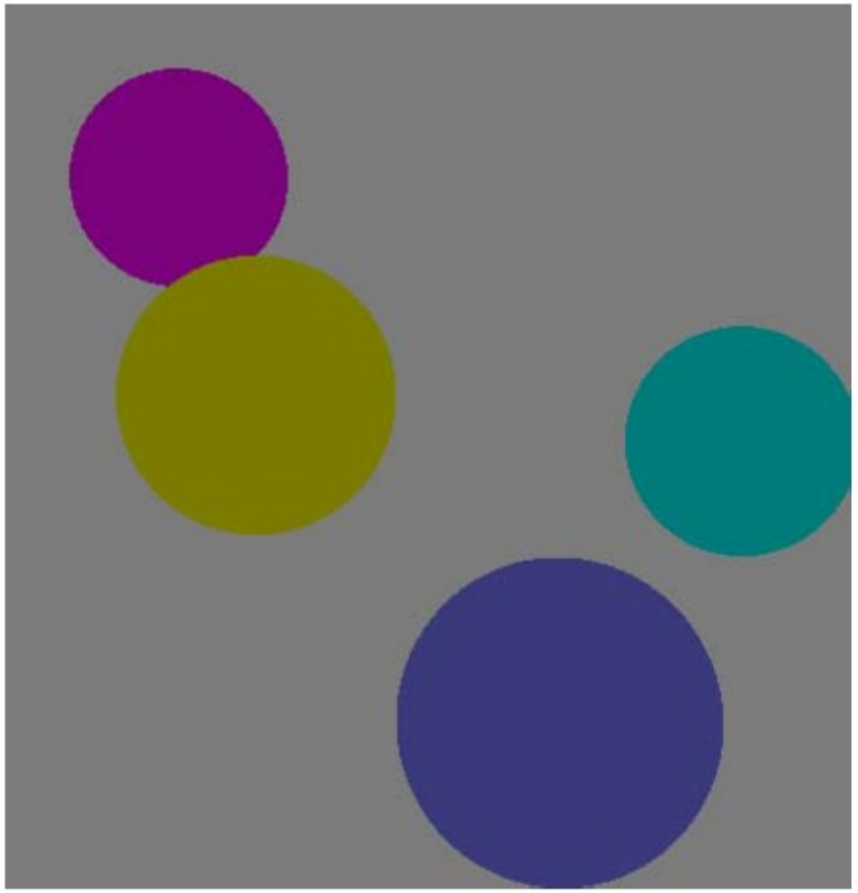


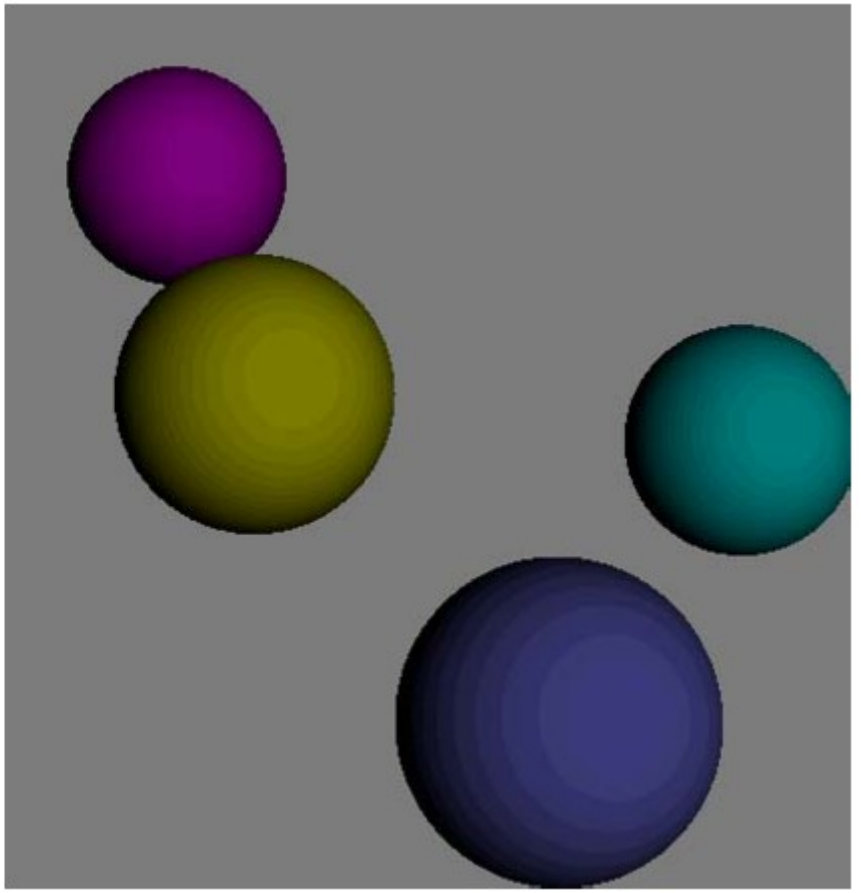
$$I_d = \begin{cases} k_d I(N \cdot L) & \text{if } N \cdot L > 0 \\ 0 & \text{if } N \cdot L \leq 0 \end{cases}$$

Reflexão Difusa

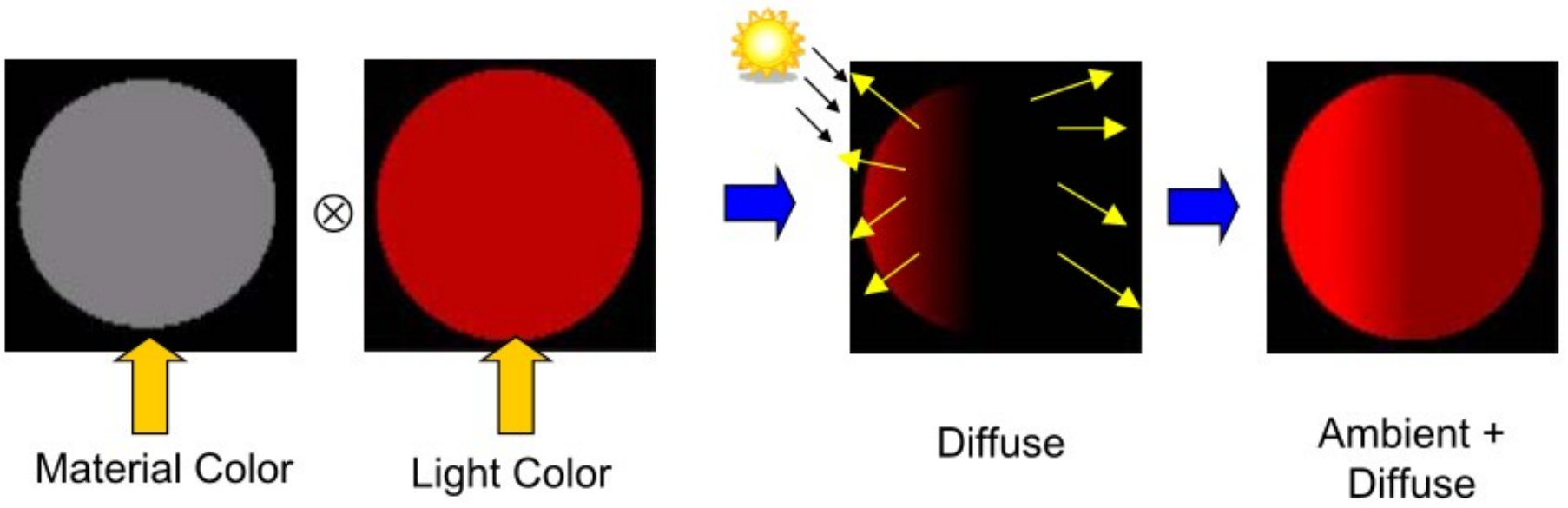
Iluminação das esferas com coeficientes de reflexão difusa de 0.0, 0.25, 0.5, 0.75 e 1 respectivamente

Luz Ambiente vs Reflexão Difusa





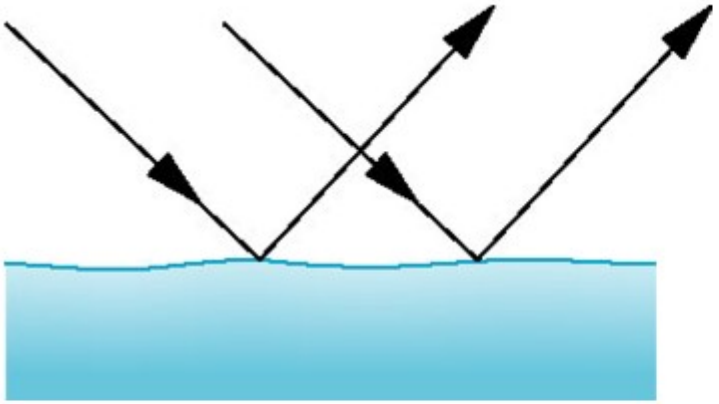
Iluminação Difusa: O que está faltando?

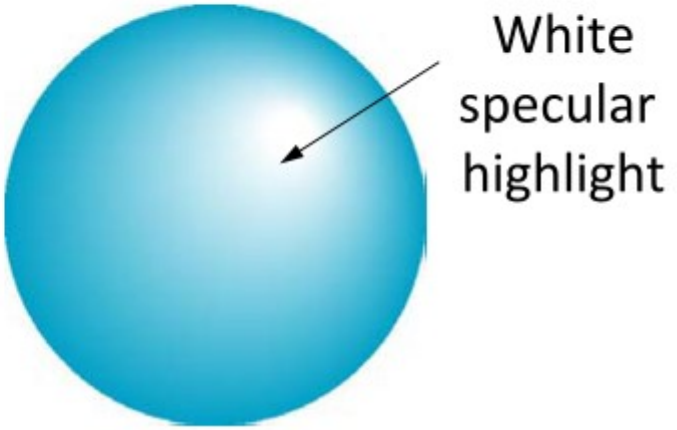


- Luz ambiente + reflexão difusa produzem imagens sombreadas que parecem tridimensionais
  - Mas as superfícies parecem opacas como giz
  - O que está faltando?

Reflexão Especular

- A intensidade depende de onde o observador está

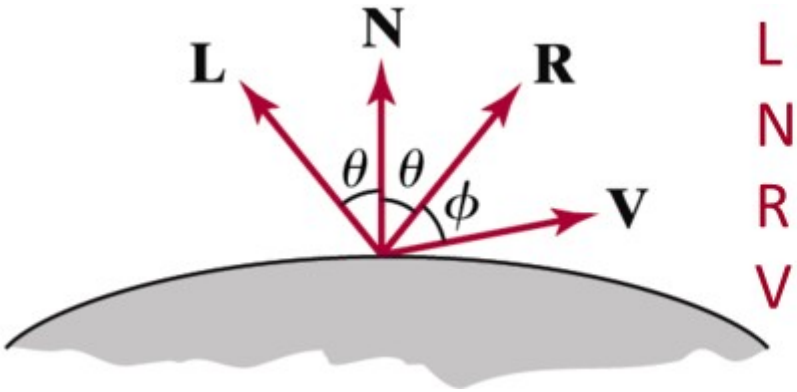




- O destaque especular branco é a reflexão da luz branca da origem em direção ao observador

Reflexão Especular

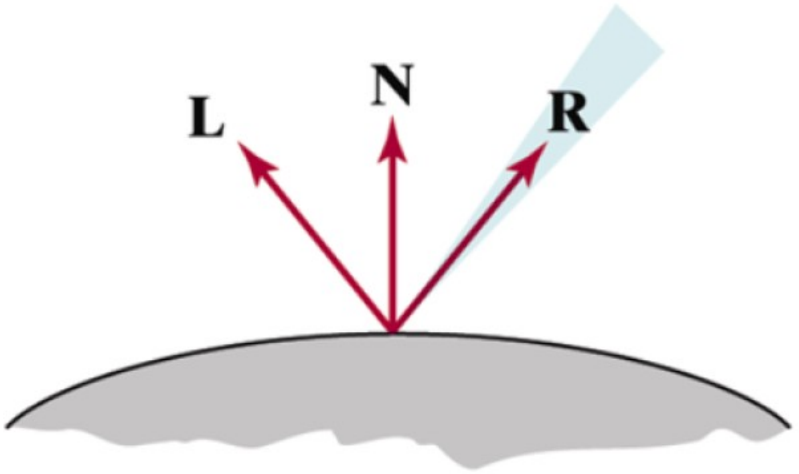
- O local branco que vimos na superfície brilhosa é o resultado da incidência da luz refletida em uma região concentrada ao redor do ângulo de reflexão especular
- O ângulo da reflexão especular é igual ao ângulo da luz incidente



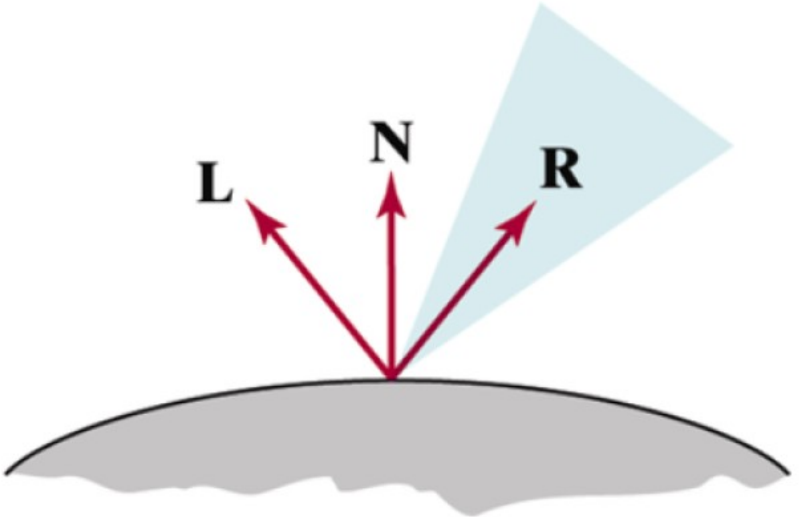
L = vector to light source
N = vector normal to surface
R = direction of reflected light
V = vector to viewer

Reflexão Especular

- Um espelho perfeito reflete a luz somente na direção da reflexão especular
- Outros objetos exibem reflexões especulares em um intervalo finito de posições de visualização ao redor do vetor *R*



Shiny Surface
(Large n_s)

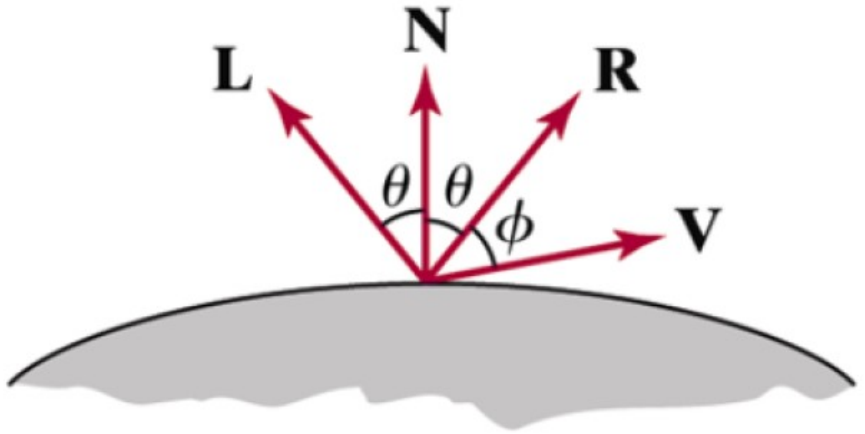


Dull Surface
(Small n_s)

Modelo de Reflexão Especular de Phong

- Define a intensidade da reflexão especular como proporcional ao ângulo φ entre o vetor de observação e o vetor de reflexão especular:

$$I_s = I \times k_s \times cos^{\alpha}\varphi$$



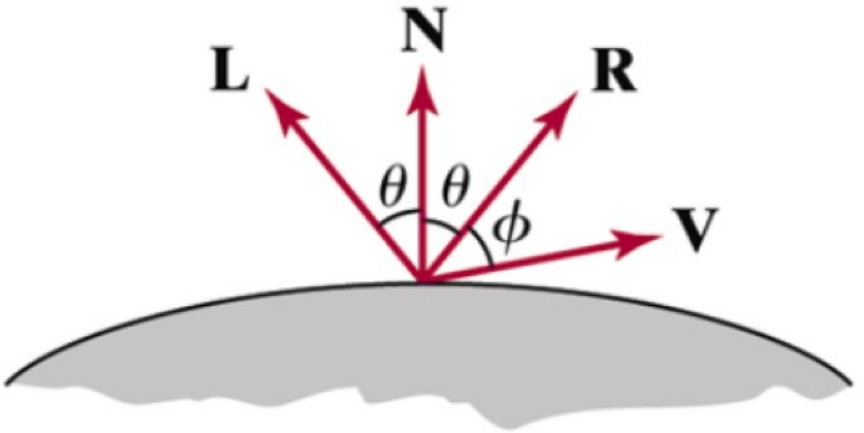
α = expoente de reflexão especular (shininess)
k_s = refletividade especular do material

Modelo de Reflexão Especular de Phong

- Define a intensidade da reflexão especular como proporcional ao ângulo φ entre o vetor de observação e o vetor de reflexão especular:

$$I_s = I \times k_s \times cos^{\alpha}\varphi$$

α = shininess
k_s = refletividade do material



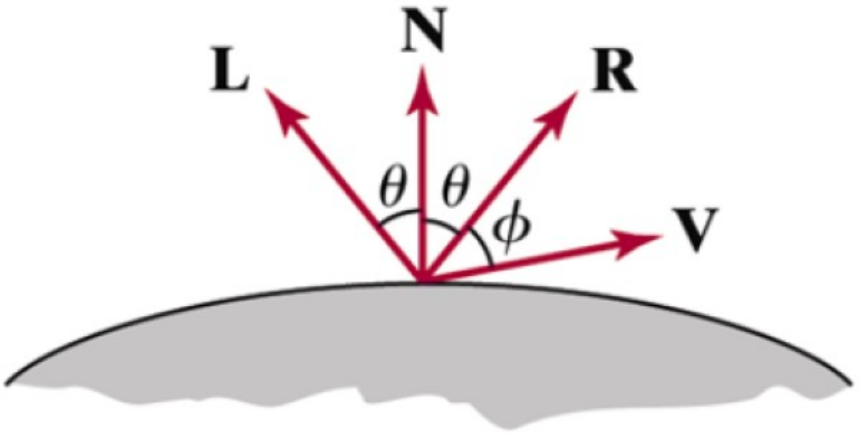
- O shininess α é determinado pelo tipo de superfície
  - Superfícies brilhosas possuem valores algos (>100)
  - Superfícies ásperas possuem valores próximos de 1
- Quanto maior o α, mais concentrada é a luz ao redor de R. Para espelhos, $\alpha \rightarrow$ infinito

Modelo de Reflexão Especular de Phong

- Define a intensidade da reflexão especular como proporcional ao ângulo φ entre o vetor de observação e o vetor de reflexão especular:

$$I_s = I \times k_s \times cos^{\alpha}\varphi$$

α = shininess
k_s = refletividade do material

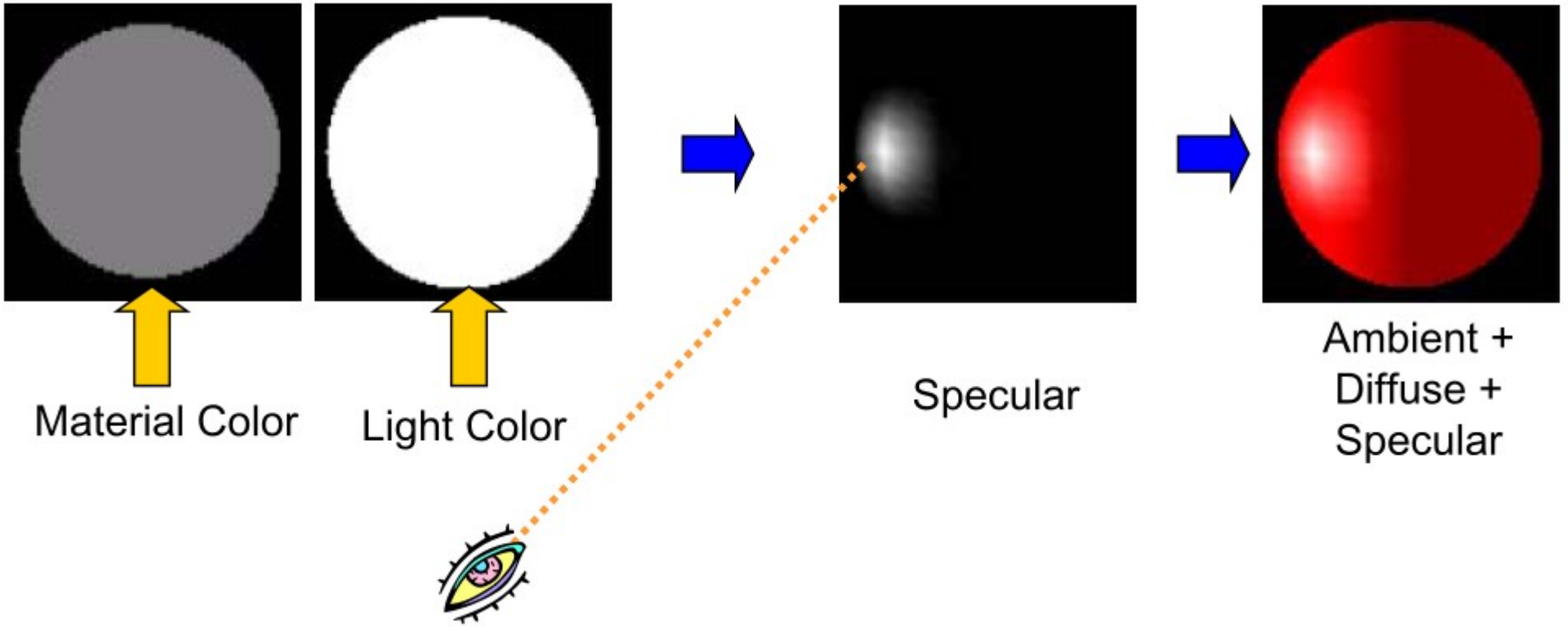


- Relembrando que $R \cdot V = cos\ \varphi$

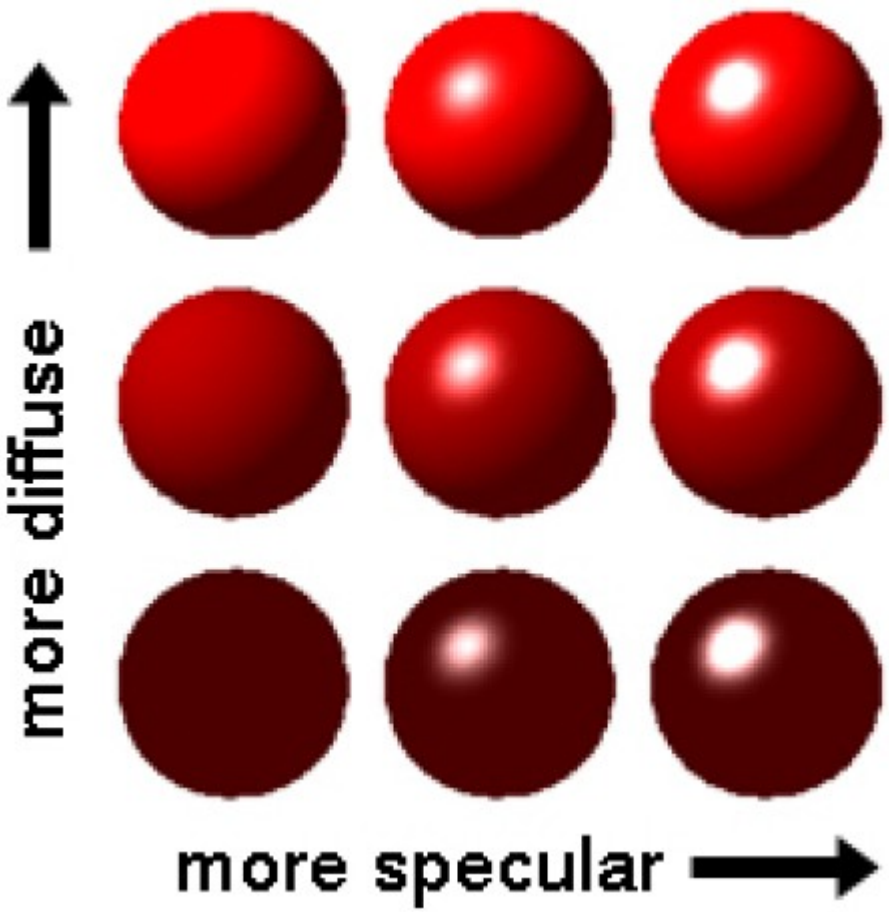
$$I_s = \begin{cases} k_s I\ (V \cdot R)^{\alpha} & \text{if } V \cdot R > 0 \text{ and } N \cdot L > 0 \\ 0.0 & \text{if } V \cdot R < 0 \text{ or } N \cdot L \leq 0 \end{cases}$$

Iluminação Especular

- Cria uma superfície brilhante (a superfície reflete perfeitamente)
- Depende do ponto de visão



Exemplo de Reflexão Difusa + Especular

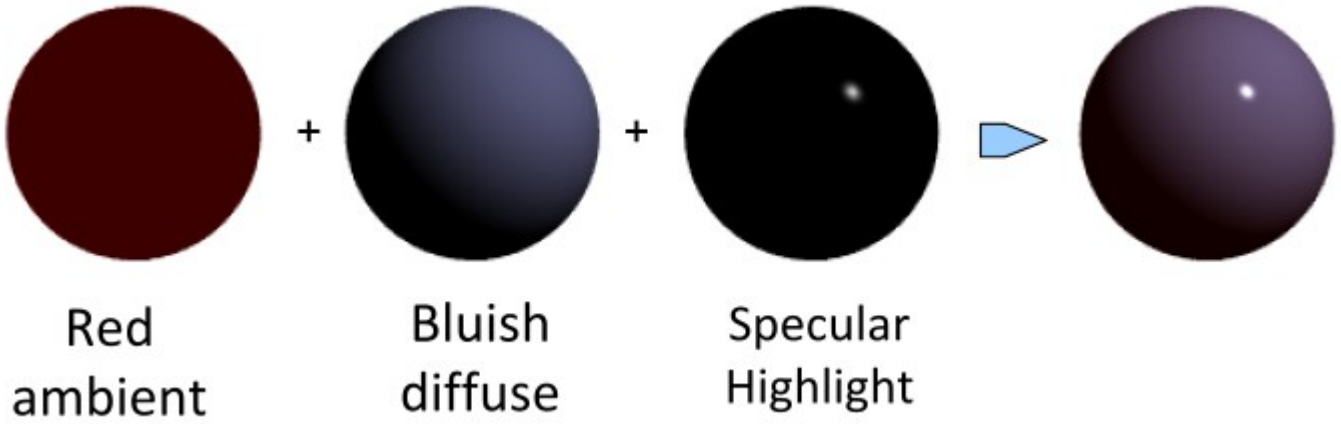


Resumo do Modelo de Reflexão de Phong

- Devido à luz ambiente, nada pode ficar completamente preto
- Reflexões espelhadas são possíveis
- Podem ser computadas muito rapidamente
- Aproximação muito boa de superfícies difusas
- Fisicamente impreciso
- Expresso em termos de geometria vetorial

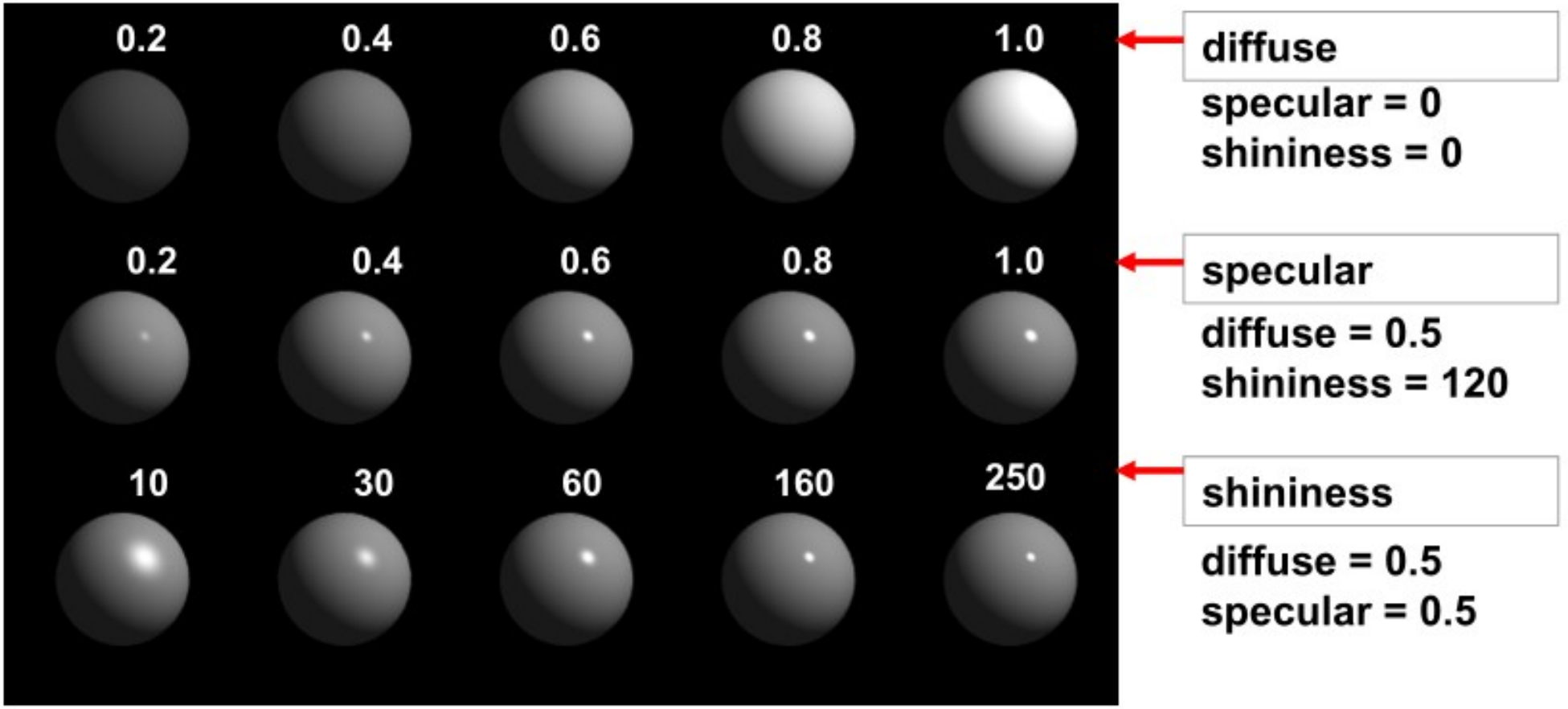
Misturando Tudo

- A intensidade da luz a partir de um ponto é a soma dos componentes difuso, especular e ambiente:



$$I = I_a + I_d + I_s$$

Misturando Tudo



Funções de Iluminação e Rendering OpenGL

- Fornece funções para
  - Definir pontos de fonte de luz e refletores;
  - Selecionar os coeficientes de reflexão da superfície;
  - Escolher valores para diversos parâmetros no modelo básico de iluminação
- Implementa o modelo de reflexão de Phong

Funções de Iluminação e Rendering OpenGL

- As **rotinas de iluminação** são ativadas usando:

```
glEnable(GL_LIGHTING);
```

- **Múltiplas fontes de luz** podem ser adicionadas a uma cena e **várias propriedades** podem ser associadas a cada fonte usando:

```
glLight*(light_name, light_property, property_value);
```

- Um sufixo `i`, `iv`, `f` ou `fv` é adicionado ao nome da função dependendo do **tipo de dado** do valor da propriedade

Funções de Iluminação e Rendering OpenGL

- O parâmetro `light_name` recebe um identificador:

```
GL_LIGHT0, GL_LIGHT1, GL_LIGHT2,..., GL_LIGHT7
```

- Depois de todos os parâmetros de uma luz terem sido definidos, essa deve ser ligada usando:

```
glEnable(light_name);
```

Propriedades da Luz

- É definida pela função

```
glLight*(lightName, lightProperty, propertyValue);
```

- O parâmetro `lightProperty` pode ser:

  - `GL_POSITION`
  - `GL_AMBIENT`
  - `GL_DIFFUSE`
  - `GL_SPECULAR`
  - `GL_CONSTANT_ATTENUATION`
  - `GL_LINEAR_ATTENUATION`
  - `GL_QUADRATIC_ATTENUATION`
  - `GL_SPOT_DIRECTION`
  - `GL_SPOT_CUTOFF`
  - `GL_SPOT_EXPONENT`

Especificando a Posição e Tipo de uma Fonte de Luz

- Para designar a **posição** de uma fonte de luz usa-se o flag `GL_POSITION` e passa-se um vetor de 4 elementos
  - Os primeiros 3 elementos do vetor definem sua posição em coordenadas do mundo.
- Quarto elemento do vetor é usado para definir o **tipo de fonte de luz**:
  - Fonte próxima da cena (posição):
    - Quarto elemento do vetor diferente de 0.0
  - Fonte distante da cena (direção):
    - Quarto elemento do vetor igual a 0.0
    - O raio de luz está na direção da linha partindo do ponto (x, y, z) até a origem.

Especificando a Posição e Tipo de uma Fonte de Luz

- O seguinte exemplo define duas fontes de luz, uma local e uma distante:

```
//fonte local
GLfloat light0_pos[4] = {2.0, 0.0, 3.0, 1.0};
//fonte distante
GLfloat light1_pos[4] = {0.0, 1.0, 0.0, 0.0};
//define posição da luz local
glLightfv(GL_LIGHT0, GL_POSITION, light0_pos);
glEnable(GL_LIGHT0);
//define direção da luz distante
glLightfv(GL_LIGHT1, GL_POSITION, light1_pos);
glEnable(GL_LIGHT1);
```

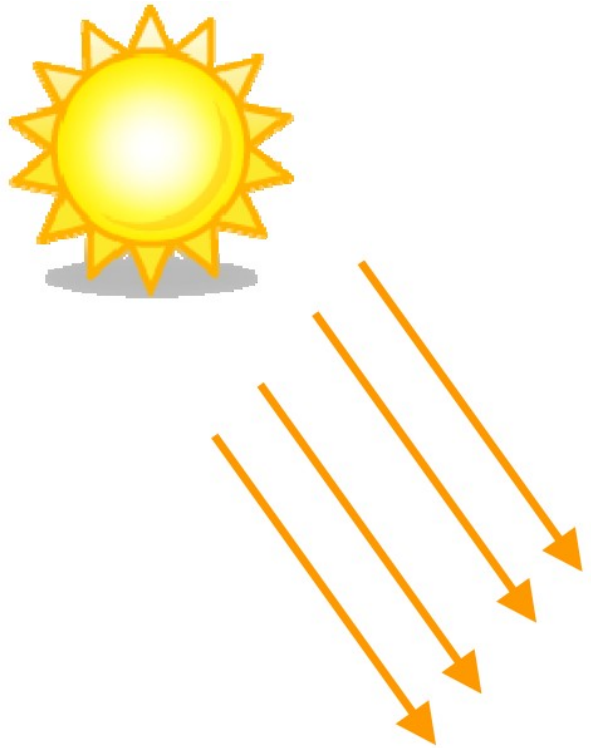
- Os valores padrão de uma fonte de luz são (0.0, 0.0, 1.0, 0.0)
  - Fonte de luz distante e luz na direção negativa de z

Especificando a Posição e Tipo de uma Fonte de Luz

- Para uma fonte local, as luzes emitidas irradiam em todas as direções e a posição da luz é usada no cálculo da iluminação
  - A direção da luz muda para cada objeto
  - Em OpenGL é chamado de luz posicional

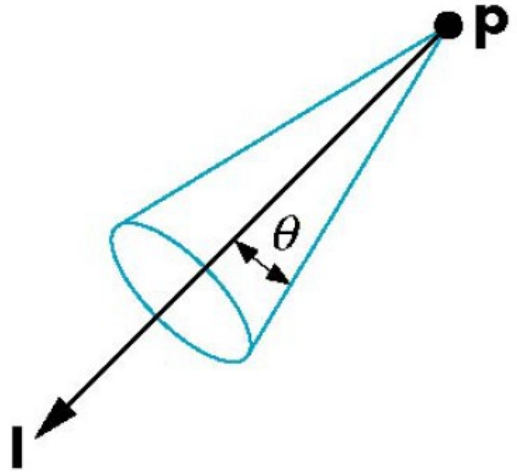
Especificando a Posição e Tipo de uma Fonte de Luz

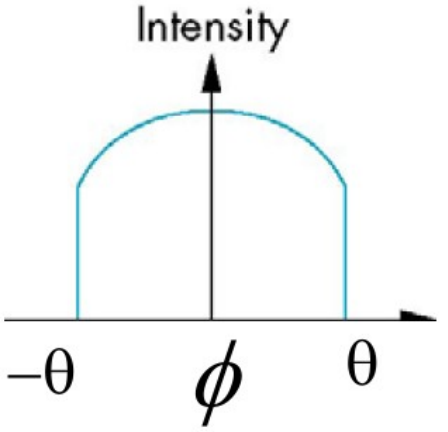
- Para uma fonte de luz no infinito, a luz emitida irradia em somente uma direção e esta direção é aplicada para todos os objetos na cena
  - A direção da luz é constante para cada objeto
  - Em OpenGL é chamada de luz direcional



Especificando a Posição e Tipo de uma Fonte de Luz: Refletores (Spotlights)

- Também chamados de fontes de luz direcionais
- Limita a luz a uma região em formato de cone
- Utilize `glLightv` para definit
  - A direção: `GL_SPOT_DIRECTION`
  - O ponto de corte (θ): `GL_SPOT_CUTOFF`
  - Atenuação (α): `GL_SPOT_EXPONENT`
    - Proporcional a $\cos^{\alpha}\varphi$





Especificando a Posição e Tipo de uma Fonte de Luz: Refletores (Spotlights)

- **Exemplo:**

```
GLfloat dirVector [] = {1.0, 0.0, 0.0};
glLightf (GL_LIGHT3, GL_SPOT_DIRECTION, dirVector);
glLightf (GL_LIGHT3, GL_SPOT_CUTOFF, 30.0);
glLightf (GL_LIGHT3, GL_SPOT_EXPONENT, 2.5);
```

Especificando as Cores da Fonte de Luz

- A cor da luz é definida especificando as diferentes cores RGBA
  - O componente **alpha** só é usado quando se ativa a transparência na cena
- Para cada fonte de luz especifica-se sua contribuição para efeitos de luz ambiente, difusa e especular

```
GLfloat white[4] = {1.0, 1.0, 1.0, 1.0};
GLfloat black[4] = {0.0, 0.0, 0.0, 1.0};

glLightfv(GL_LIGHT0, GL_AMBIENT, black);
glLightfv(GL_LIGHT0, GL_DIFFUSE, white);
glLightfv(GL_LIGHT0, GL_SPECULAR, white);
```

- Por padrão, para a **luz 0** a propriedade de **luz ambiente** é preta e branca para as propriedades difusa e especular
  - Para as outras luzes todas as propriedades são pretas

Exemplo 1 – Especificando as Cores da Fonte de Luz

```
#include <GL/glut.h>

void lighting(){
    float position[4] = {2.0f, 2.0f, 2.0f, 1.0f};
    float white[4] = {1.0f, 1.0f, 1.0f, 1.0f};
    float black[4] = {0.0f, 0.0f, 0.0f, 1.0f};
    glLightfv(GL_LIGHT0, GL_POSITION, position);
    glLightfv(GL_LIGHT0, GL_AMBIENT, black);
    glLightfv(GL_LIGHT0, GL_DIFFUSE, white);
    glLightfv(GL_LIGHT0, GL_SPECULAR, white);

    //ativa a iluminação
    glEnable(GL_LIGHTING);
    glEnable(GL_LIGHT0);
}
```

Exemplo 1 – Especificando as Cores da Fonte de Luz

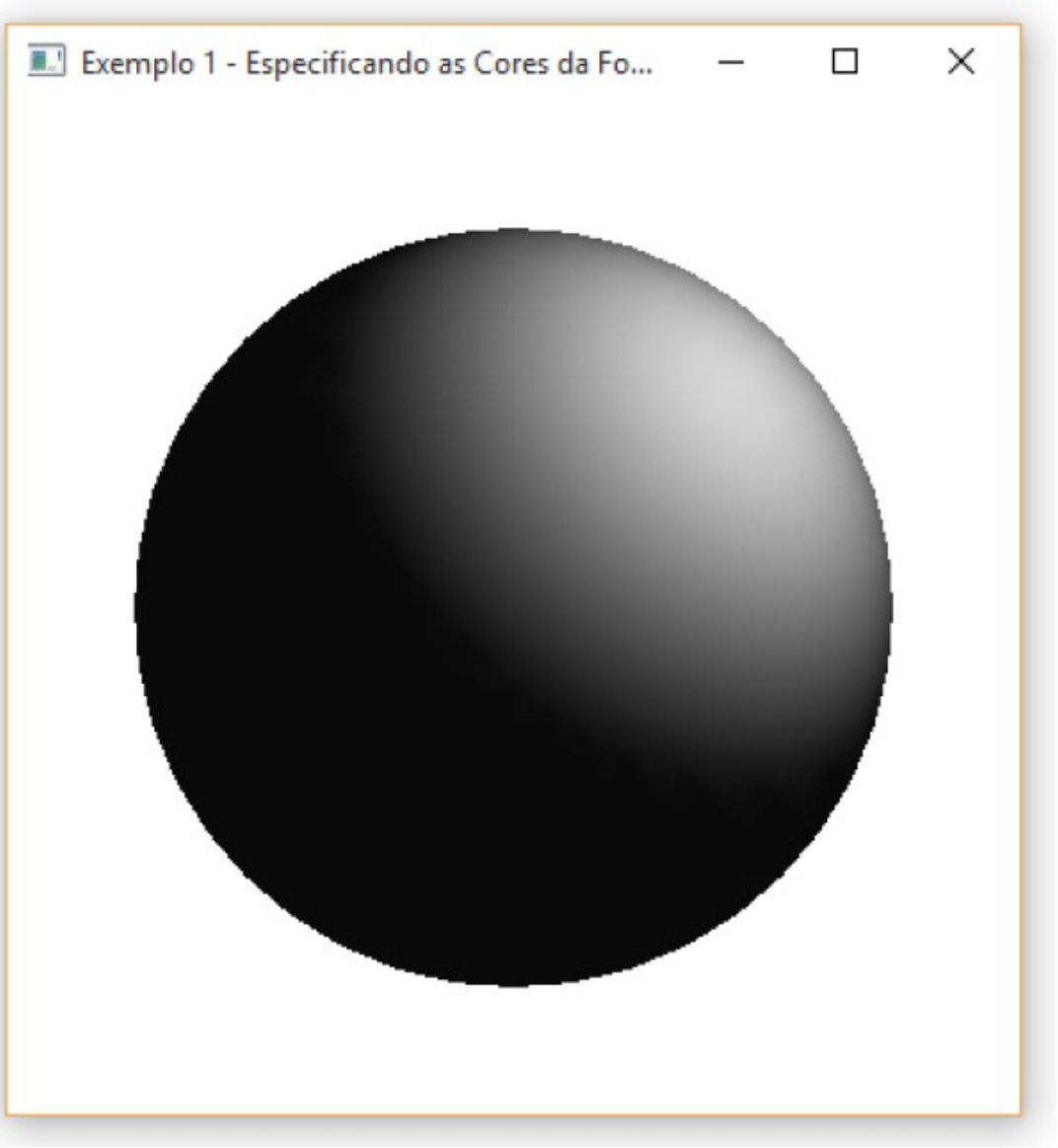
```
int init(){
    glClearColor(1.0f, 1.0f, 1.0f, 1.0f); //define a cor de fundo
    glEnable(GL_DEPTH_TEST); //habilita o teste de profundidade

    glMatrixMode(GL_MODELVIEW); //define que a matriz é a model view
    glLoadIdentity(); //carrega a matriz de identidade
    gluLookAt(0.0, 0.0, 1.0, //posição da câmera
              0.0, 0.0, 0.0, //para onde a câmera aponta
              0.0, 1.0, 0.0); //vetor view-up
    glMatrixMode(GL_PROJECTION); //define que a matriz é a de projeção
    glLoadIdentity(); //carrega a matriz de identidade
    glOrtho(-2.0, 2.0, -2.0, 2.0, -2.0, 2.0);

    lighting();
}
```

Exemplo 1 – Especificando as Cores da Fonte de Luz

UTFPR
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

```
void display(){
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    glMatrixMode(GL_MODELVIEW);
    glutSolidSphere(1.5, 40, 40);

    glFlush();
}

int main(int argc, char** argv){
    glutInit(&argc,argv);
    glutInitDisplayMode(GLUT_SINGLE| GLUT_RGB | GLUT_DEPTH);

    glutInitWindowPosition(200,0);
    glutInitWindowSize(400,400);
    glutCreateWindow("Exemplo 1 - Definindo as Cores da Fonte de Luz");

    init();
    glutDisplayFunc(display);
    glutMainLoop();
    return 0;
}
```

Exemplo 1 – Especificando as Cores da Fonte de Luz



Especificando os Coeficientes de Atenuação Radial

- É possível especificar os **coeficientes de atenuação radial** a_0, a_1, a_2 usando:

```
//define a0
glLightfv(GL_LIGHT0, GL_CONSTANT_ATTENUATION, 0.5);
//define a1
glLightfv(GL_LIGHT0, GL_LINEAR_ATTENUATION, 0.15);
//define a2
glLightfv(GL_LIGHT0, GL_QUADRATIC_ATTENUATION, 0.1);
```

- Os valores dos coeficientes podem ser **inteiros** ou de **ponto flutuante** positivos
  - Os valores padrão para a atenuação radial são $a_0 = 1$, $a_1 = 0$, $a_2 = 0$ (atenuação desativada)

Exemplo 2 - Especificando os Coeficientes de Atenuação Radial

UTFPR
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

```
void lighting(){
    //uma fonte de luz local
    float position[4] = {2.0f, 2.0f, 2.0f, 1.0f};
    float white[4] = {1.0f, 1.0f, 1.0f, 1.0f};
    float black[4] = {0.0f, 0.0f, 0.0f, 1.0f};

    glLightfv(GL_LIGHT0, GL_POSITION, position);
    glLightfv(GL_LIGHT0, GL_AMBIENT, black);
    glLightfv(GL_LIGHT0, GL_DIFFUSE, white);
    glLightfv(GL_LIGHT0, GL_SPECULAR, white);

    //ativa a atenuação
    glLightf(GL_LIGHT0, GL_CONSTANT_ATTENUATION, 0.5f); //define a0
    glLightf(GL_LIGHT0, GL_LINEAR_ATTENUATION, 0.5f);   //define a1
    glLightf(GL_LIGHT0, GL_QUADRATIC_ATTENUATION, 0.1f);   //define a2

    //ativa a iluminação
    glEnable(GL_LIGHTING);
    glEnable(GL_LIGHT0);
}
```

Exemplo 2 - Especificando os Coeficientes de Atenuação Radial

Parâmetros de Iluminação Global

- Vários **parâmetros de luz** podem ser definidos globalmente usando:

```
glLightModel*(param_name, param_value);
```

- O sufixo pode ser `i`, `iv`, `f` ou `fv` dependendo do tipo de parâmetro
- Podemos definir globalmente:
  - O nível de **luz ambiente**
  - Como os **brilhos especulares** são calculados
  - Se o **modelo de iluminação** deve ser aplicado na parte de trás dos polígonos

Parâmetros de Iluminação Global

- Por exemplo, para definir uma **luz ambiente** independente das fontes de luz existentes usamos:

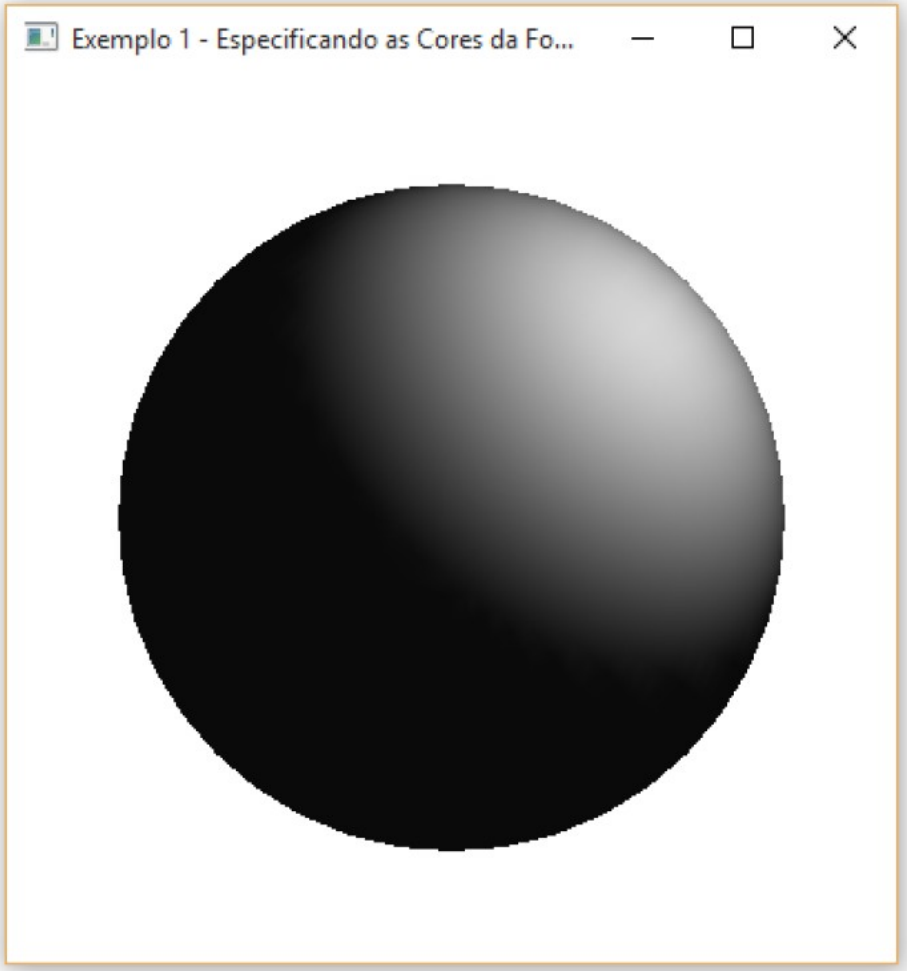
```
float global_ambient[4] = {0.9, 0.9, 0.9, 1.0};

glLightModelfv(GL_LIGHT_MODEL_AMBIENT, global_ambient);
```

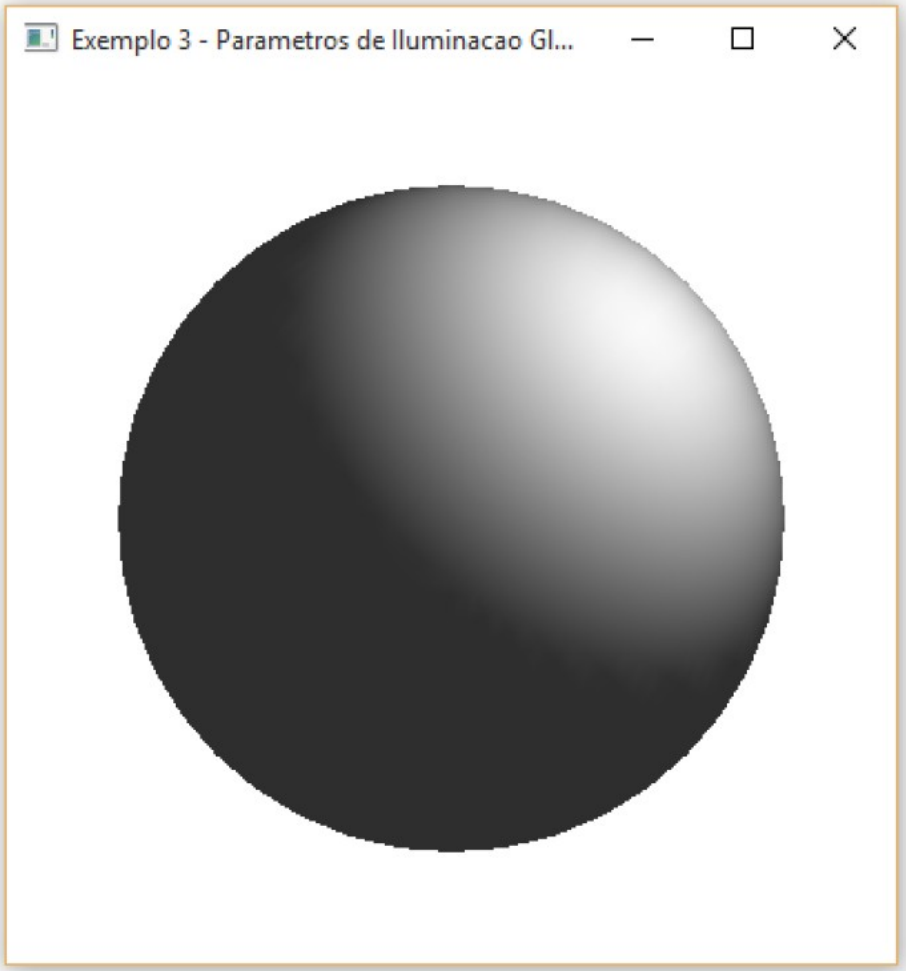
- Por padrão essa cor é **branca** de baixa intensidade (0.2, 0.2, 0.2, 1.0)

Exemplo 3 – Parâmetros de Iluminação Global

```
void lighting(){
    //uma fonte de luz local
    float position[4] = {2.0f, 2.0f, 2.0f, 1.0f};
    float white[4] = {1.0f, 1.0f, 1.0f, 1.0f};
    float black[4] = {0.0f, 0.0f, 0.0f, 1.0f};

    glLightfv(GL_LIGHT0, GL_POSITION, position);
    glLightfv(GL_LIGHT0, GL_AMBIENT, black);
    glLightfv(GL_LIGHT0, GL_DIFFUSE, white);
    glLightfv(GL_LIGHT0, GL_SPECULAR, white);

    //ativa luz ambiente global
    // LA independente de fonte de luz existentes
    float global_ambient[4] = {0.9, 0.9, 0.9, 1.0};
    glLightModelfv(GL_LIGHT_MODEL_AMBIENT, global_ambient);

    //ativa a iluminação
    glEnable(GL_LIGHTING);
    glEnable(GL_LIGHT0);
}
```

Exemplo 3 – Parâmetros de Iluminação Global





Parâmetros de Iluminação Global

- Para o cálculo da reflexão especular é necessário **determinar o vetor V** (da superfície para a posição de visão)
  - Podemos acelerar o processamento fazendo V constante independente da posição da superfície
- O valor padrão para V é a direção de z, (0.0, 0.0, 1.0), mas podemos usar a posição de visão corrente fazendo:

```
glLightModeli(GL_LIGHT_MODEL_LOCAL_VIEWER, GL_TRUE);
```

- Isso tornará o processo computacionalmente mais caro, mas o resultado será mais realístico
  - Para desabilitar o cálculo de V, fazer essa função igual a `GL_FALSE`

Propriedades da Superfície

- As propriedades óticas das superfícies são definidas usando:

```
glMaterial*(surface_face, surface_property,
                property_value);
```

- O sufixo pode ser `i`, `iv`, `f` ou `fv` dependendo do tipo de parâmetro
- O parâmetro `surface_face` indica a qual face o material se designa e pode ser:
  - `GL_FRONT`, `GL_BACK` e `GL_FRONT_AND_BACK`
- O parâmetro `surface_property` identifica o parâmetro da superfície e pode ser:
  - k_a, k_d, k_s ou n_s

Propriedades da Superfície

- Os flags `GL_AMBIENT, GL_DIFFUSE` e `GL_SPECULAR` são usados para definir os coeficientes de reflexão da superfície
  - Normalmente os valores dos coeficientes difuso e ambiente são os mesmos, para isso usamos `GL_AMBIENT_AND_DIFFUSE`
- Os valores padrão para os coeficientes são:
  - Luz ambiente (k_a): (0.2, 0.2, 0.2, 1.0)
  - Luz difusa (k_d): (0.8, 0.8, 0.8, 1.0)
  - Luz especular (k_s): (1.0, 1.0, 1.0, 1.0)
- Para definir o expoente de reflexão especular (n_s) usamos `GL_SHININESS` entre 0 e 128
  - O valor padrão é 0

Propriedades da Superfície

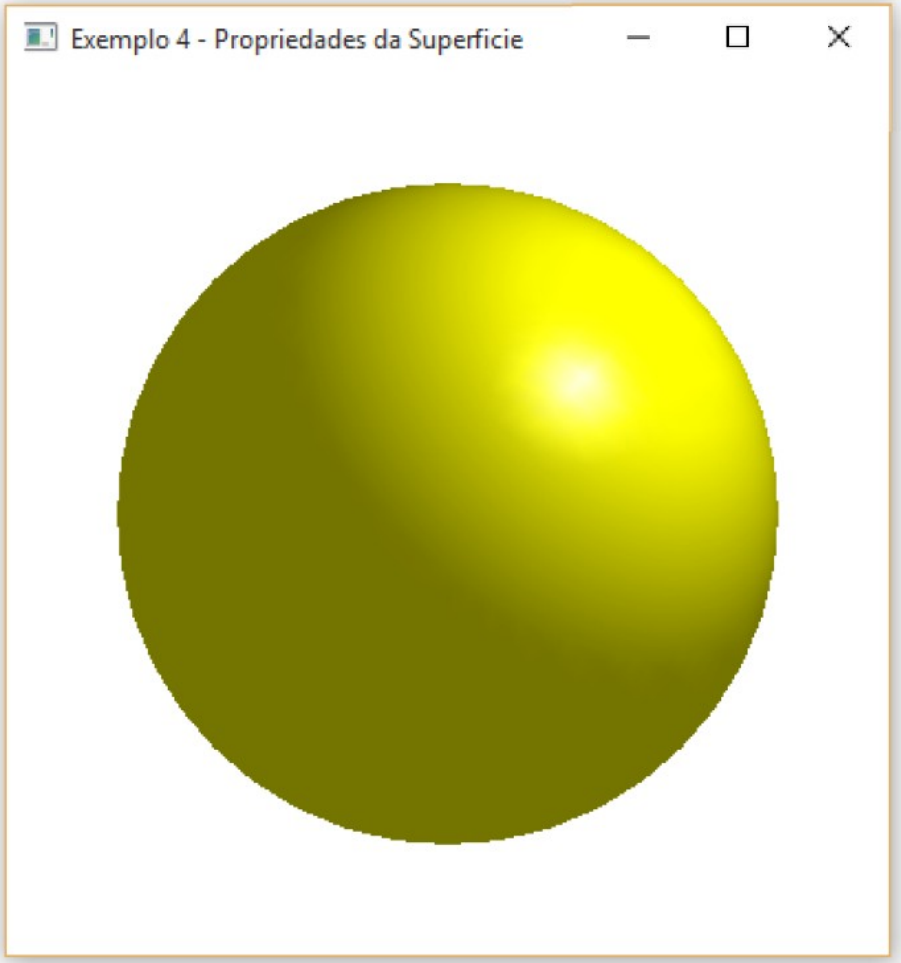
- Por exemplo, para definir os coeficientes de reflexão ambiente, difusa e especular podemos fazer:

```
float diffuse[4] = {0.65, 0.65, 0.0, 1.0};
float specular[4] = {0.9, 0.9, 0.9, 1.0};
float shininess = 65.0;

glMaterialfv(GL_FRONT, GL_AMBIENT_AND_DIFFUSE, diffuse);
glMaterialfv(GL_FRONT, GL_SPECULAR, specular);
glMaterialf(GL_FRONT, GL_SHININESS, shininess);
```

Exemplo 4 – Propriedades da Superfície

```
void display(){
    //limpa o buffer
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);

    //define material da superfície
    float kd[4] = {0.65f, 0.65f, 0.0f, 1.0f};
    float ks[4] = {0.9f, 0.9f, 0.9f, 1.0f};
    float ns = 65.0f;

    glMaterialfv(GL_FRONT, GL_AMBIENT_AND_DIFFUSE, kd);
    glMaterialfv(GL_FRONT, GL_SPECULAR, ks);
    glMaterialf(GL_FRONT, GL_SHININESS, ns);

    //define que a matrix é a a model view
    glMatrixMode(GL_MODELVIEW);
    glutSolidSphere(1.5, 40, 40);

    //força o desenho das primitivas
    glFlush();
}
```

Exemplo 4 – Propriedades da Superfície

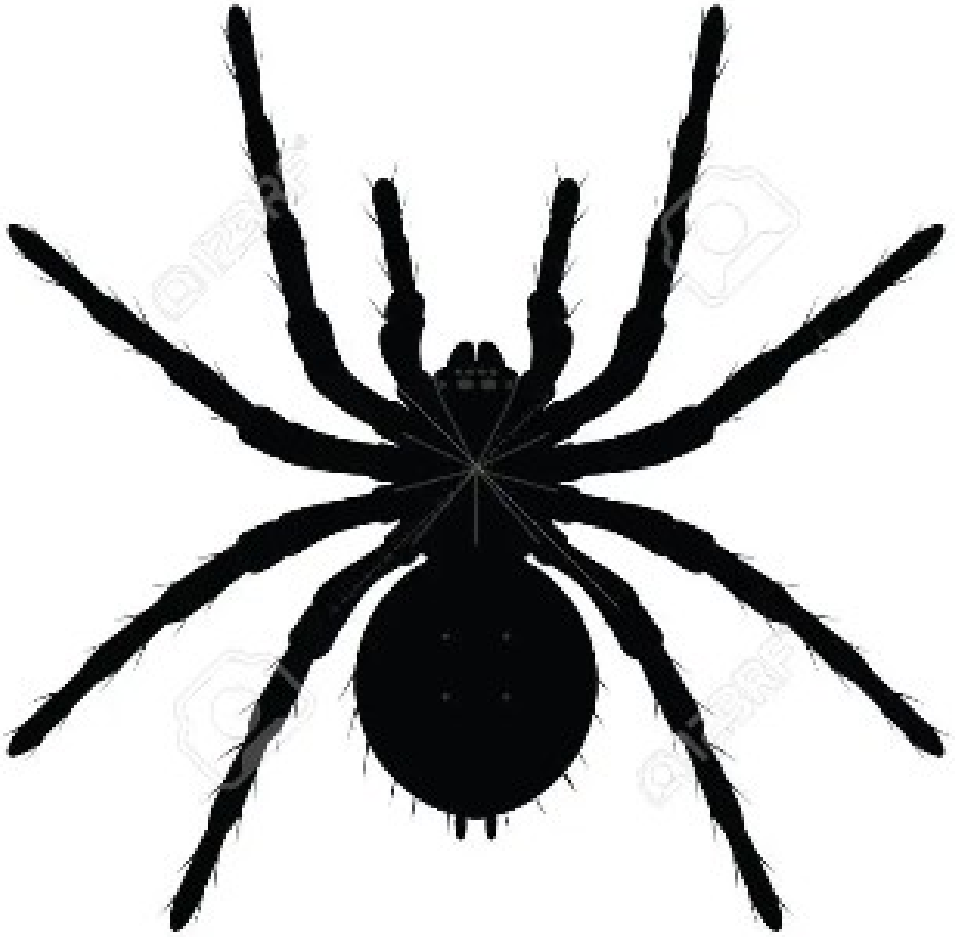


Exercício 1

- Faça um programa que simule o brilho do Sol refletindo no objeto da tela. À medida que o ângulo do Sol for se aproximando de 180, a intensidade de sua luz deve ir reduzindo. Ao passar de 0 graus, deve voltar a aumentar a intensidade.
  - **Dica:** Para determinar a posição X e Y do Sol, utilize as coordenadas polares:
    - X = raio * cos(angulo)
    - Y = raio * sen(angulo)

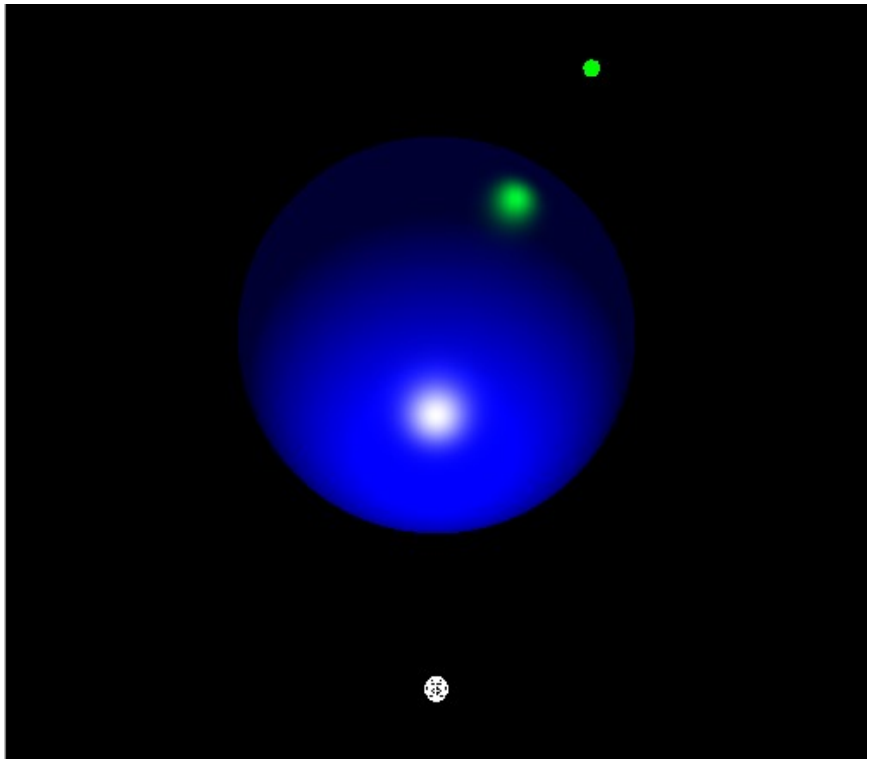
Exercício 2

- Faça um programa em OpenGL que desenhe uma aranha na tela. Esse ator deve ser constituído de cefalotórax, abdômen, dois olhos, pedipalpo e oito pernas. Esses elementos da aranha podem ser aproximados utilizando formas geométricas como esferas, linhas, paralelepípedos, cilindros, etc.
  - Crie 2 viewports, sendo uma com a visão superior da aranha e outra com uma visão lateral
  - Permita ao usuário rotacionar a aranha por meio do mouse
  - Aplique efeitos de iluminação ambiente, especular e difusa



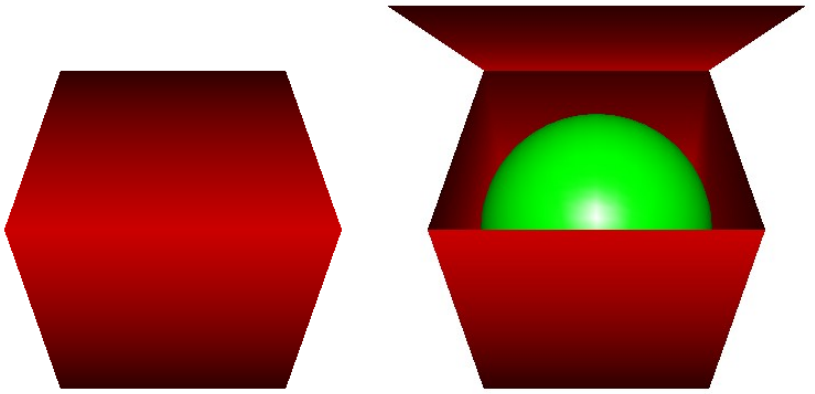
Exercício 3

- Faça um programa que exiba na tela uma esfera e duas fontes de luz (uma branca e uma verde). O usuário deve poder fazer as seguintes manipulações:
  - Ativar/desativar a luz branca
  - Ativar/desativar a luz verde
  - Aumentar/diminuir a intensidade especular e difusa da luz branca
  - Aumentar/diminuir a intensidade da luz branca ambiente
  - Alternar a posição da luz branca entre luz posicional e luz direcional
  - Alternar a posição do ponto de vista (local ou infinito)
  - Rotacionar a luz branca nos eixos X e Y
  - Mover a esfera no eixo Z



Exercício 4 – Ponto extra

- Faça uma animação que contenha uma esfera verde dentro de uma caixa vermelha. O usuário deve poder acionar as teclas direcionais (Up e Down) para abrir a tampa da caixa. A câmera deve estar posicionada em um local mais alto, de forma que seja visualizada conforme as imagens abaixo. Aplique efeitos de iluminação e de reflexão dos materiais.
  - **Dica:** para que se alcance esse efeito de transição suave da luz é preciso calcular e definir um valor médio das normais dos vértices que compõem a caixa (veja os slides 80 e 81).



Reflexão Especular e Modelo de Phong

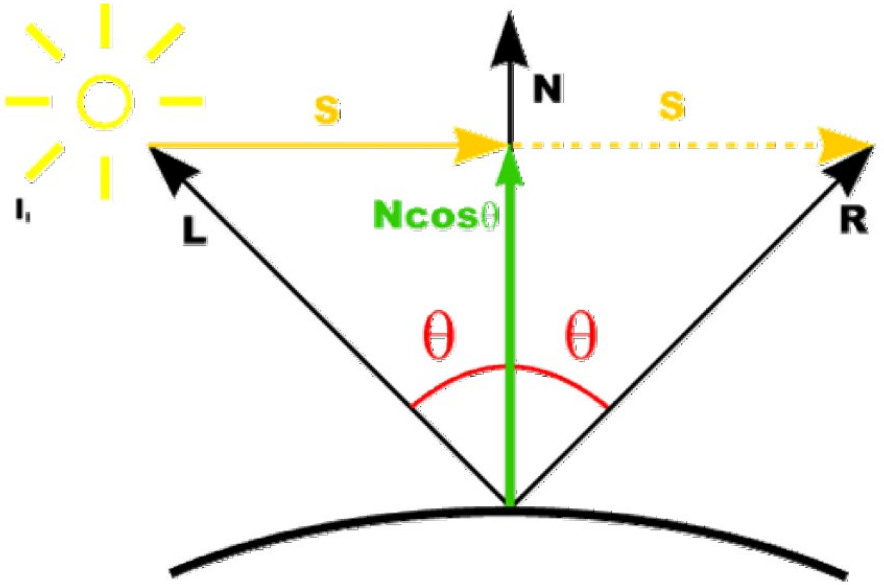
- A projeção de L na direção de N é ($N\ cos\ \theta$) (projeção escalar), então:

$$R - S = N\cos\theta$$

$$R = N\cos\theta + S$$

$$L + S = N\cos\theta$$

$$S = N\cos\theta - L$$



- De forma que o vetor de reflexão especular R é obtido fazendo:

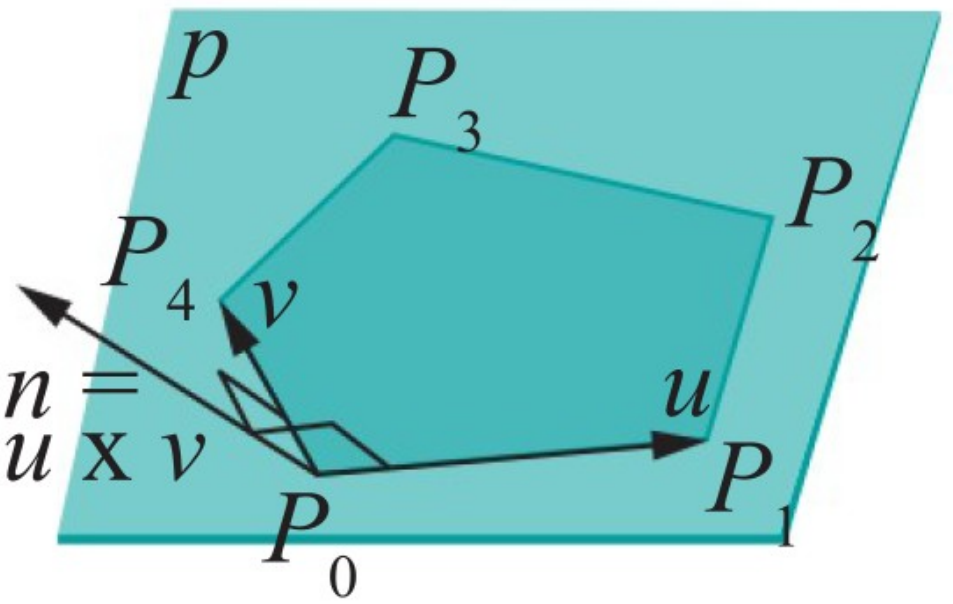
$$R = N\cos\theta + N\cos\theta - L$$

$$R = N2\cos\theta - L$$

$$R = N(2N \cdot L) - L$$

Calculando normais de um plano

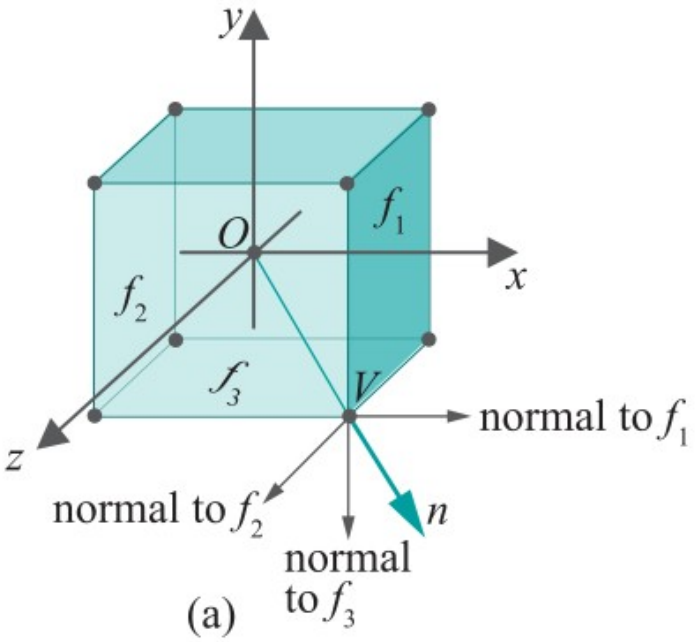
- Dado um plano *p* e dois vetores *u* e *v* não-colineares, a sua normal é dada pelo produto escalar *u* x *v*
- Na figura ao lado:
  - $u = P_1 - P_0$
  - $v = P_4 - P_0$
  - $n = u \times v$



$$P_0 = [0\ 3\ 5]^T, \quad P_1 = [1\ -2\ 0]^T, \quad P_2 = [3\ 3\ 3]^T$$

Calculando normais de um vértice

- Dado um vértice *V*, a sua normal é a média dos vetores unitários de cada face adjacente a esse vértice
- Na figura abaixo, o vértice V possui 3 faces adjacentes (f_1, f_2 e f_3), cujas normais são:
  - $f_1 = (1, 0, 0)$ $f_2 = (0, 0, 1)$ $f_3 = (0, -1, 0)$



Calculando normais de um vértice

- Calculando a média das normais, obtém-se:
  - n = [1/3, -1/3, 1/3]
- Normalizando:
  - Comprimento do vetor: $\hat{v} = \sqrt{(1/3)^2 + (-1/3)^2 + (1/3)^2} = \sqrt{3}/3$
  - $n = [\frac{u_x}{\hat{v}}, \frac{u_y}{\hat{v}}, \frac{u_z}{\hat{v}}] = [\frac{1/3}{\sqrt{3}/3}, \frac{-1/3}{\sqrt{3}/3}, \frac{1/3}{\sqrt{3}/3}] = [1/\sqrt{3}, 1/\sqrt{3}, 1/\sqrt{3}]$