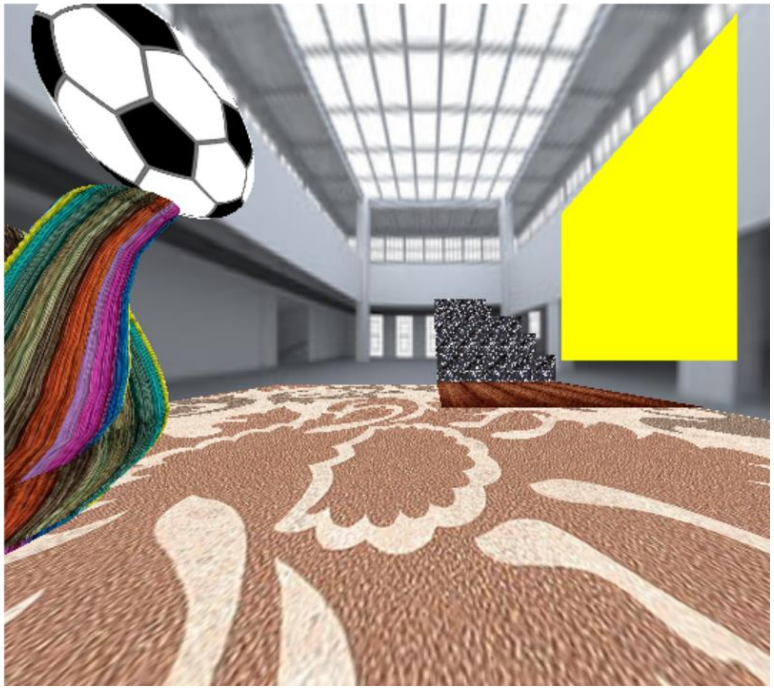
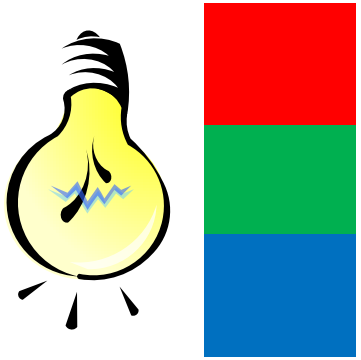
TP6 : Cor & Iluminação

Computação Gráfica



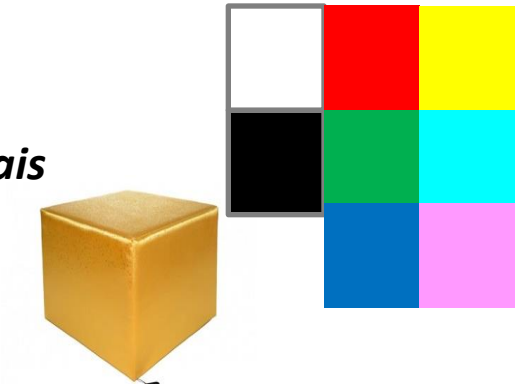
Objectivo : cor & iluminação

- **Definir Luzes e Materiais**



- Luz
  - Tipo (pontual, foco, direccional) e cor
  - **Permitir ligar cada uma das componentes de cor: R, G B**

- Material
  - Polígono: normal é fundamental !
  - ***Definir "cores"=propriedades de reflexão dos materiais***



Interacção luz+material ???

~~glColor~~

OPENGL – cor & iluminação

Assuntos a tratar neste trabalho

- ***Parte 1 - Básico***
  - Modelo interacção: luz & material

- ***Parte 2 - Detalhes***
  - 1. Color Material
  - 2. Luz dos dois lados
  - 3. Combinar iluminação com textura
  - 4. Nevoeiro (uma forma de atenuação da intensidade …)
  - 5. Malha de polígonos
  - 6. Transparências

OPENGL – cor & iluminação

- **Modelos interacção : *luz/materiais***
  - Fontes de Luz
  - Interacção luz / materiais
  - Cor

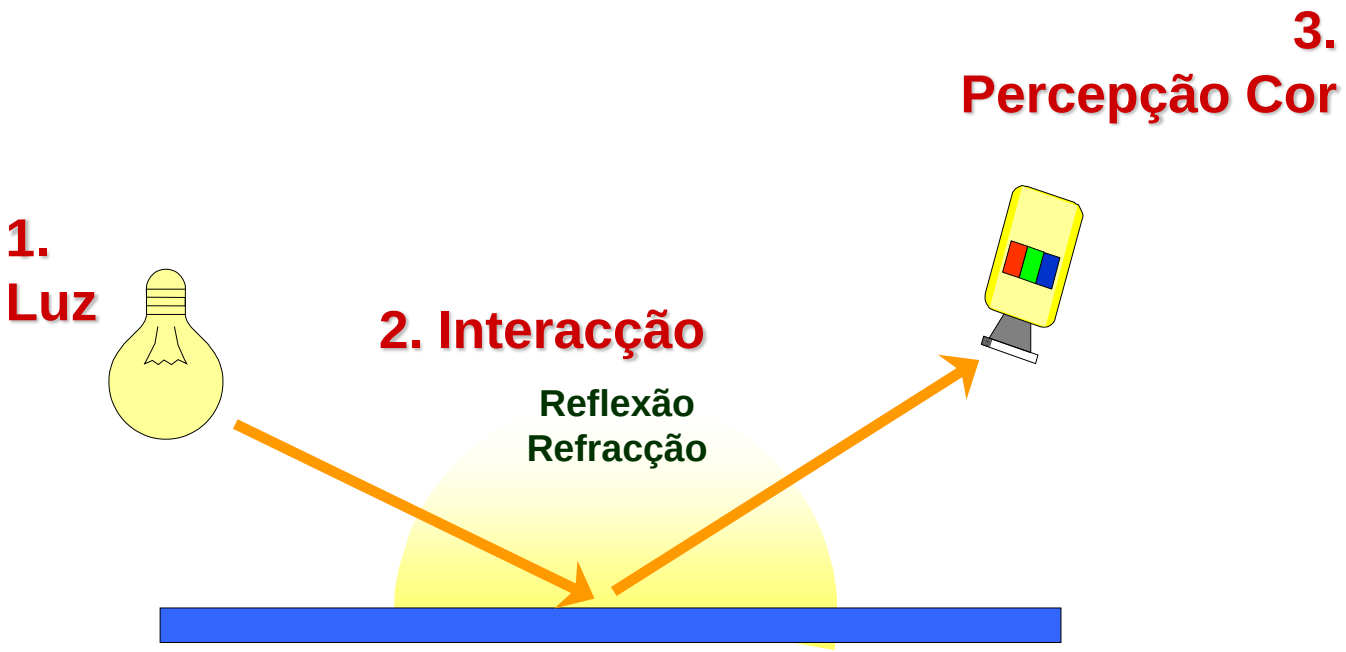
- Fontes de Luz
- Materiais
- Normais
- Ordem operações

Modelos interacção

Como calcular a cor / intensidade num vértice ?

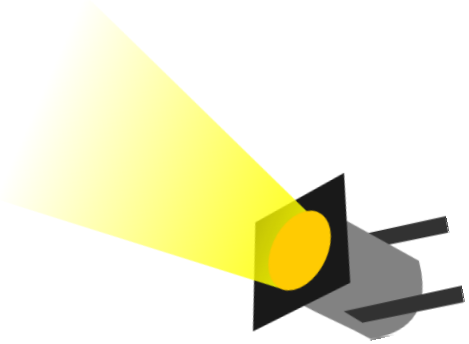
1. *Luz – tipos de luzes*
2. *Interacção luz/material*

   Modelo de Phong



OpenGL: 1. Definições

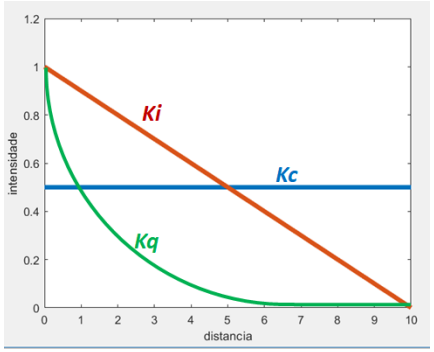
- **1. Fontes de iluminação:**
  - Pontuais (ex. *lâmpada*)
  - Direcionais (ex. *sol*)
  - Foco (ex. *holofote*)



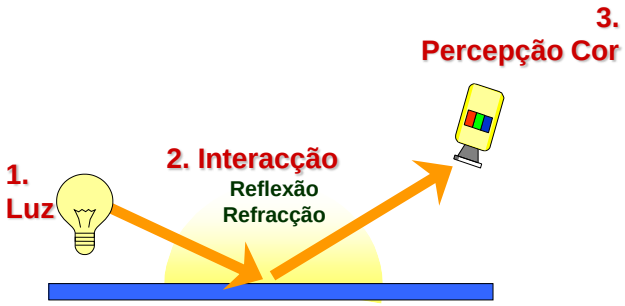
OpenGL: 1. Definições

- **2. Interacção luz/materiais**

- **Intensidade**
  - Modelo de *Phong: ambiente +difusa +especular*
  - *Atenuação: constante, linear, quadrática*



- **Percepção da cor**
  - Modelos tricromátco : **RGB**
  - Modelo multiplicatico
    - corPercepcionada = corLuz x corReflexãoMaterial

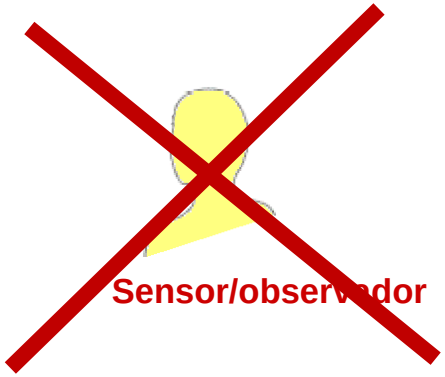


Phong : intensidade ambiente

Constante

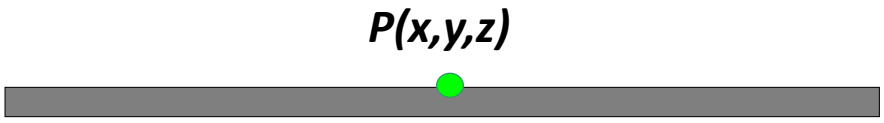
- Não depende da localização
- Não depende da orientação do objecto
- Não depende do observador

Ambiente



Intensidade em P = ?

$$I_P = K_A I_{amb}$$

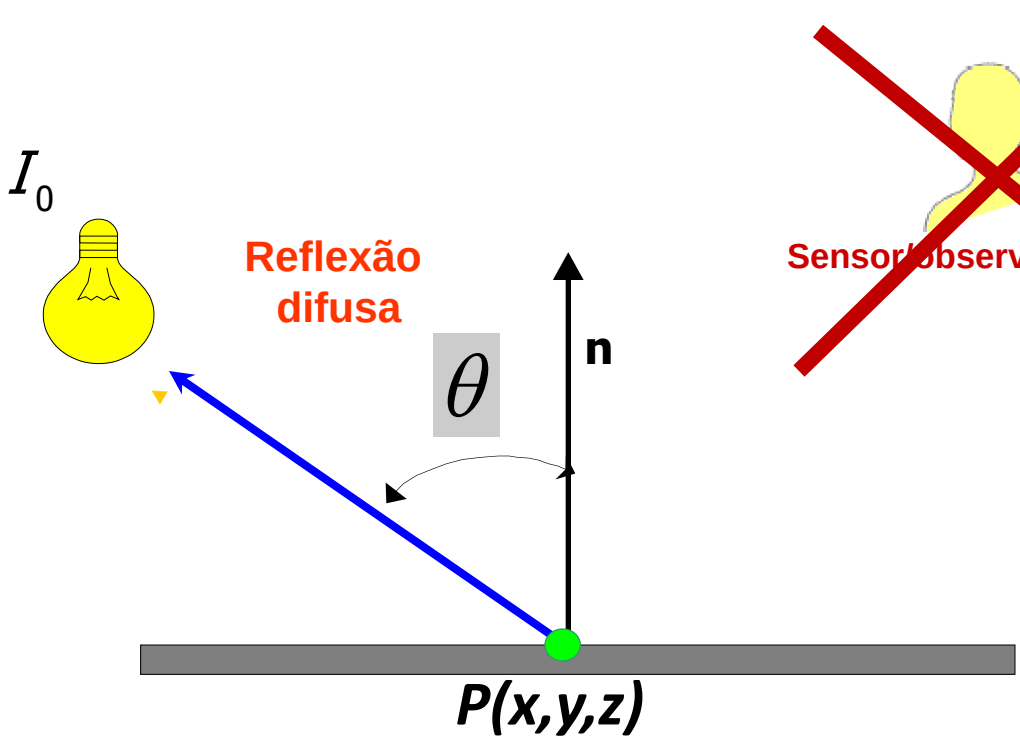


Phong : intensidade difusa

- Depende da posição da luz
- **Normal** ao objecto
- Características do material
- Independente da posição do observador

Intensidade em P = ?

$$I_P = K_B I_0 \cos\theta$$

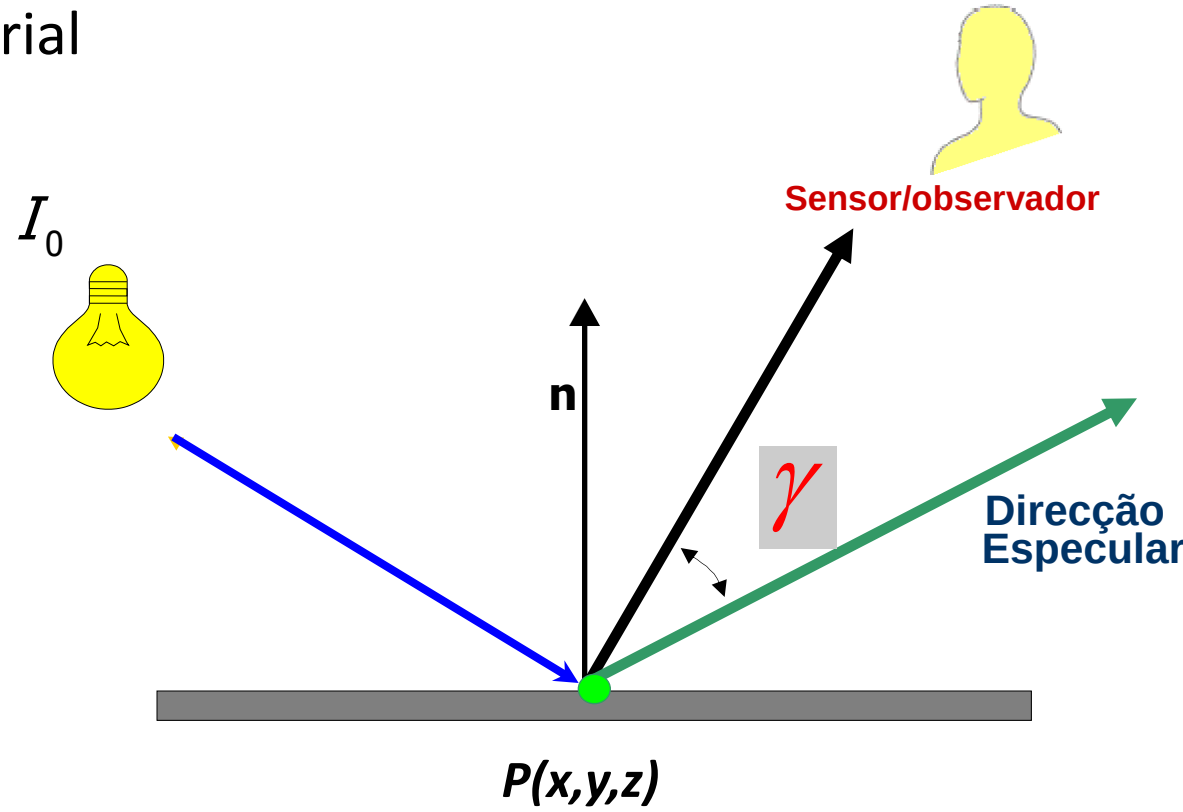


Phong : Intensidade especular

- Depende da posição da luz
- **Normal** ao objecto
- **Posição do observador**
- Características do material

Intensidade em P = ?

$$I_P = K_S I_0 \cos^{ns} \gamma$$



2. Cor ? – Modelo multiplicativo

- **Luz** caracterizada pelas componentes (R,G,B)
  - (0,0,0) – Luz preta (não emite nada)
  - (1,1,1) – Luz branca
  - (1,0,0) – Luz vermelha

- **Material** caracterizado pelas componentes que reflecte (R,G,B)
  - (0,0,0) – Não reflete nada (material é preto)
  - (1,1,1) – Reflete tudo (material é branco)
  - (0,1,0) – Reflete verde (material verde)

- **Exemplo: cor percepcionada pelo observador ?**
  - Luz: (1.0, 0.0 ,0.5)
  - Material (0.5, 0.5, 0.5)
  - **Cor = (0.5, 0.0, 0.25)**

Modelo **multiplicativo**
corLuz x corReflexãoMAterial

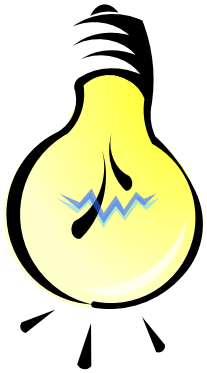
OPENGL – cor & iluminação

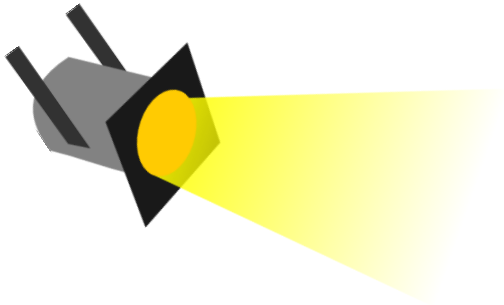
- Modelos interacção : *luz/materiais*
- **Fontes de Luz – *como definir em openGL ?***
- Materiais
- Normais
- Ordem operações

Modelos de Luz

OpenGL : como definir

- Tipos de fontes de luz ?
- Atenuação ?





$$I(x,y,z) = \frac{I_0}{k_c + k_l d + k_q d^2}$$

OpenGL: 1. Fontes de Luz

- Activação global

```
glEnable (GL_LIGHTING);
```

- **8 fontes** de Luz individuais parametrizáveis
  - pontuais,
  - direcionais,
  - Foco

- Activação
  - **`glEnable (source);`**
    - **`source`** é uma constante cujo nome é **$\mathbf{GL\_LIGHT_i}$**,
    - Começando com **`GL_LIGHT0`**

OpengL: 1. Fontes de Luz

- Configuração:

```
glLightfv(source, property, value);
```

  - **Property**, constante, permitindo definir:
    - ***Coeficientes*** de cor usados no modelo de iluminação:
      - **GL_AMBIENT, GL_DIFFUSE, GL_SPECULAR**
    - ***Tipo*** (geometria) da fonte
      - **GL_POSITION,**
      - **GL_SPOT_DIRECTION, GL_SPOT_CUTOFF, GL_SPOT_EXPONENT**
    - Coeficientes de ***atenuação***
      - **GL_CONSTANT_ATTENUATION, GL_LINEAR_ATTENUATION,**
      - **GL_QUADRATIC_ATTENUATION**

OpenGL: 1. Fontes de Luz

- Adicionalmente: componente ambiente global
  - É possível usar luminosidade ambiente não relacionada com fontes luminosas individuais
    - Intensidade para toda a cena (valores RGB)

```
• glLightModelfv(GL_LIGHT_MODEL_AMBIENT, luzGlobalCor);

    luZGlobalCor ={0.5, 0.8, 0.3}
```

Exemplo: definir luz pontual

- Posição: posição
- Emite apenas ambiente + difusa e é amarela “clara”
- Atenuação constante, linear e quadrática = {2.0, 0.2, 0.1}

- Definir propriedades
  - glLightf**v** (GL_LIGHT0, GL_POSITION, **Posicao**);
  - glLightf**v** (GL_LIGHT0, GL_AMBIENT, **CorAmbiente**);
  - glLightf**v** (GL_LIGHT0, GL_DIFFUSE, **CorDifusa**);
  - glLightf (GL_LIGHT0, GL_CONSTANT_ATTENUATION, **atConst**);
  - glLightf (GL_LIGHT0, GL_LINEAR_ATTENUATION, **0.2**) ;
  - glLightf (GL_LIGHT0, GL_QUADRATIC_ATTENUATION, **0.1**);

- **Ligar/desligar luz**
  - glEnable(GL_LIGHT0)
  - glDisable(GL_LIGHT0);

```
float    localizacao [ ] ={1,2,3, 1};
float  corAmbiente [ ] ={0.8,0.8,0.0,1.0};
float       corDifusa [ ] =0.8,0.8,0.0,1.0};
float   atConst = 2.0;
```

Luzes pontuais / direcionais: 1/0

- Definição de uma ***luz pontual*** na ***posição*** **(x,y,z)**
  - Pos[4] = **{ x, y, z , 1.0 };**
  - glLightfv(GL_LIGHT1, GL_POSITION, ***Pos***);

- Definição de uma ***luz directional*** de ***direcção*** **(x,y,z)**
  - Dir[4] = **{ x, y, z , 0.0 };**
  - glLightfv(GL_LIGHT1, GL_POSITION, ***Dir***);

1. Definições: modelo interacção

- OpenGL
  - Modelos interacção : *luz/materiais*
  - Fontes de Luz
  - **Materiais** – ***definir propriedades dos Materiais em OpenGL ?***
  - Normais
  - Ordem operações

Materiais

- ***Phong***: considera ambiente + reflexão especular + reflexão difusa

- ***Material*: como definir as suas propriedades ?**
  - Características de reflexão da luz/componente **ambiente**
  - Características de reflexão da luz/componente **difusa**
  - Características de reflexão da luz/componente **especular**

- Além disso
  - Coeficientes de reflexão ambiente Ka
  - Coeficientes de reflexão difusa Kb
  - Coeficientes de reflexão especular Ks
    - Coeficiente de especularidade **Shininess - Ce**

OpengL: 2. Materiais

- Configuração:

```
glMaterialfv (face, property, value)
```

- **Face**: quais lados da superfície se quer configurar:
  - **GL_FRONT, GL_BACK, GL_FRONT_AND_BACK**

- **Property**: propriedade do material (características reflexão):
  - **GL_AMBIENT, GL_EMISSION, GL_DIFFUSE, GL_SPECULAR, GL_SHININESS**

value={RGB}={r, g, b} ∈[0..1]

Exemplo: definir material verde

- Exemplo material “VERDE” (nas três componentes)
- **Capacidade de reflectir VERDE**
  - corAmb [ ] = { 0, 1, 0 };
  - corDif [ ] = { 0, 1, 0 };
  - corSpec [ ] = { 0, 1, 0 };
  - Coef = 2;

- Definir propriedades
  - glMaterialf**v** (GL_FRONT, **GL_AMBIENT**, **corAmb**);
  - glMaterialf**v** (GL_FRONT, **GL_DIFFUSE**, **corDf**);
  - glMaterialf**v** (GL_FRONT, **GL_SPECULAR**, **corSpec**);
  - glMaterialf (GL_FRONT, **GL_SHININESS**, **Coef**);

Materiais: propriedades

Name	Ambient			Diffuse			Specular			Shininess
Values	**R**	**G**	**B**	**R**	**G**	**B**	**R**	**G**	**B**	**value**
Ruby	0.1745	0.01175	0.01175	0.61424	0.04136	0.04136	0.727811	0.626959	0.626959	0.6
Bronze	0.2125	0.1275	0.054	0.714	0.4284	0.18144	0.393548	0.271906	0.166721	0.2
Gold	0.24725	0.1995	0.0745	0.75164	0.60648	0.22648	0.628281	0.555802	0.366065	0.4
Silver	0.19225	0.19225	0.19225	0.50754	0.50754	0.50754	0.508273	0.508273	0.508273	0.4
red plastic	0.0	0.0	0.0	0.5	0.0	0.0	0.7	0.6	0.6	0.25
black rubber	0.02	0.02	0.02	0.01	0.01	0.01	0.4	0.4	0.4	0.078125

Exemplo: definir material dourado

- Exemplo material “OURO”
  - goldAmb [ ]= { 0.2472, 0.1995, 0.0745 };
  - goldDif [ ]= { 0.7516, 0.6064, 0.2264 };
  - goldSpec [ ]= { 0.6282, 0.5558, 0.3660 };
  - goldCoef = 0.4 *128;

Gold	0.24725	0.1995	0.0745	0.75164	0.60648	0.22648	0.628281	0.555802	0.366065	0.4

- Definir propriedades
  - glMaterialf**v** (GL_FRONT, **GL_AMBIENT**, **goldAmb**);
  - glMaterialf**v** (GL_FRONT, **GL_DIFFUSE**, **goldDif**);
  - glMaterialf**v** (GL_FRONT, **GL_SPECULAR**, **goldSpec**);
  - glMaterialf (GL_FRONT, **GL_SHININESS**, **goldCoef**);

OPENGL – cor & iluminação

- **Modelos interacção : *luz/materiais***
- Fontes de Luz
- Materiais

- **Normais – *já falámos sobre a importância das normais !***

- Ordem operações

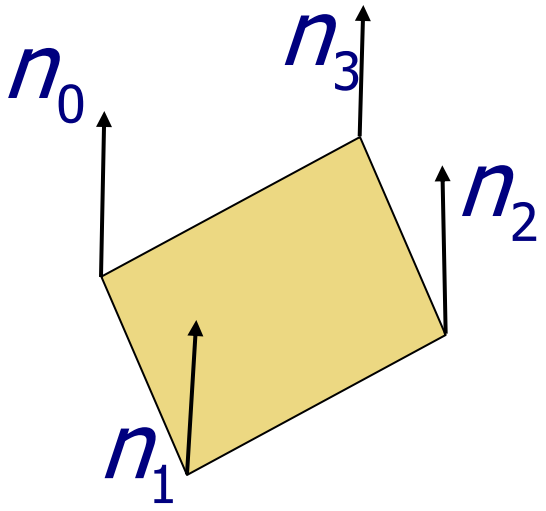
Normais

- Além das propriedades da luz e materiais, a geometria do objecto é também importante:
  - A posição dos vértices em relação ao observador e à fonte de iluminação
  - O vector *normal* é fundamental
    - **NÃO É** calculado automaticamente
    - Necessita de ser especificada com `glNormal(.)`
    - Por omissão "aponta" para fora (*regra da mão direita*)

Normais

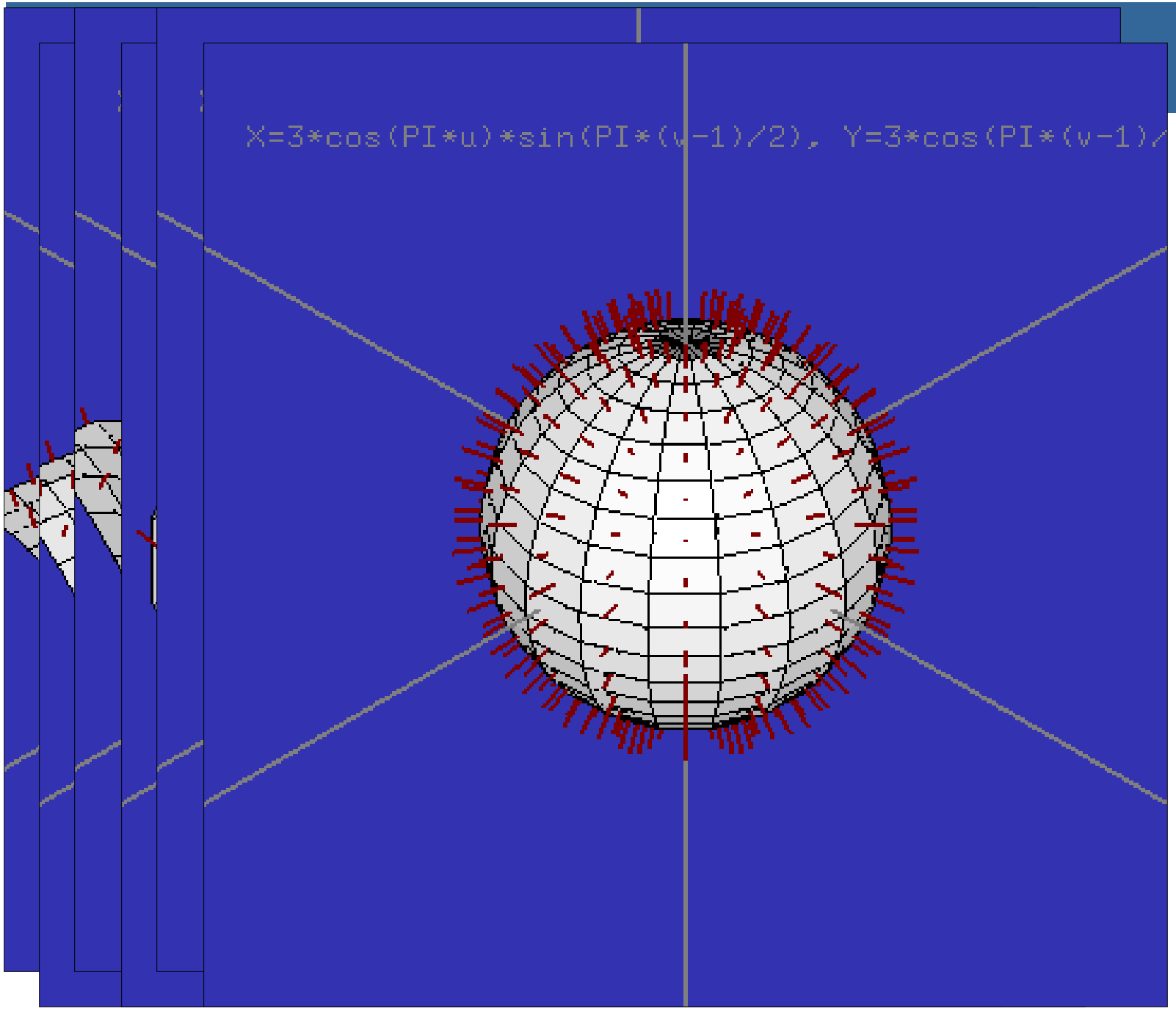
- Definição vértice a vértice:

```
glNormal3fv(n0);
glBegin (GL_POLYGON);
    glVertex3fv(v0);
    glVertex3fv(v1);
    glVertex3fv(v2);
    glVertex3fv(v3);
glEnd();
```



Normais

- Normalização:
  - Por omissão normais não são normalizadas
  - Para o fazer
    - glEnable(GL_NORMALIZE)
  - Implica cálculos adicionais, …
  - Problema caso não esteja normalizado !!!

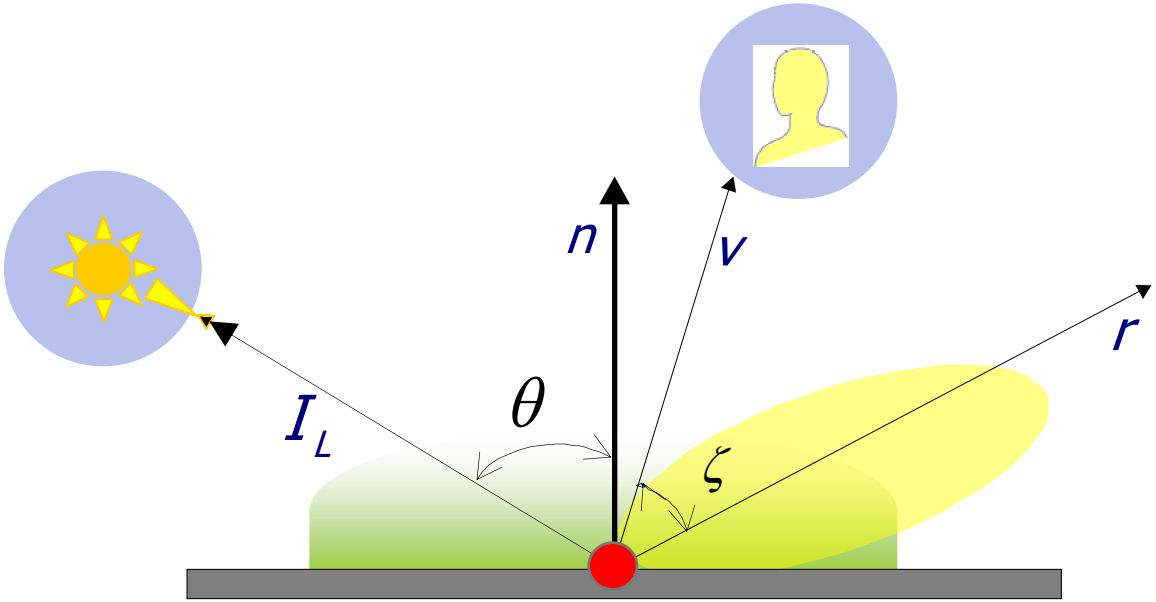

X=3*cos(PI*u)*sin(PI*(v-1)/2), Y=3*cos(PI*(v-1)/

OPENGL – cor & iluminação

- Modelos interacção : *luz/materiais*
- Fontes de Luz
- Materiais
- Normais

- **Ordem operações –** ***qual a ordem da iluminação ?***

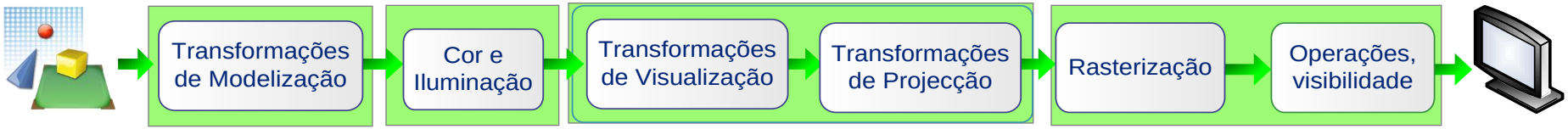
Ordem das operações



Cor = ?

Ordem das operações

- Ordem correcta de operações



5. Window

4. Projecção

3. Observador

2. Iluminação

1. Vértices/objectos

Ordem sem iluminação

```
//------------------------------------------------ Janela
glViewport (0, 0, w, h);                                    //… 5. window

//------------------------------------------------ Projecção
glMatrixMode (GL_PROJECTION);
glLoadIdentity( );
glOrtho (-1.0, 1.0, -1.0, 1.0, -10.0, 10.0);                //… 4. Projecção+volume

//------------------------------------------------ Observador+objectos
glMatrixMode (GL_MODELVIEW);
glLoadIdentity( );
gluLookAt( Ox, Oy, Oz, Dx, Dy, Dz, UPx, UPy, UPz );         //… 3. observador
glutSolidTeapot( );                                         //… 1. objectos
```

Ordem com Iluminação

```
//---------------------------------------- Janela
glViewport (0, 0, w, h);                                  //… 5. window

//---------------------------------------- Projecção
glMatrixMode (GL_PROJECTION);
glLoadIdentity( );
glOrtho (-1.0, 1.0, -1.0, 1.0, -10.0, 10.0);              //… 4. projecção+volume

//---------------------------------------- Observador+Luz+objectos
glMatrixMode (GL_MODELVIEW);
glLoadIdentity( );
gluLookAt( Ox, Oy, Oz, Dx, Dy, Dz, UPx, UPy, UPz );       //… 3. observador
Light position                                            //… 2. iluminação
glutSolidTeapot( );                                       //… 1.objectos
```

IMP: Uma fonte de Luz é tratado como um Vértice/objecto
É afectada pela matriz MODELVIEW

Regras : IMPORTANTE !!

***GL_Lighting** enabled ?*

glEnable(GL_LIGHTING);
glDisable(GL_LIGHTING);

No

Yes

Calculada com

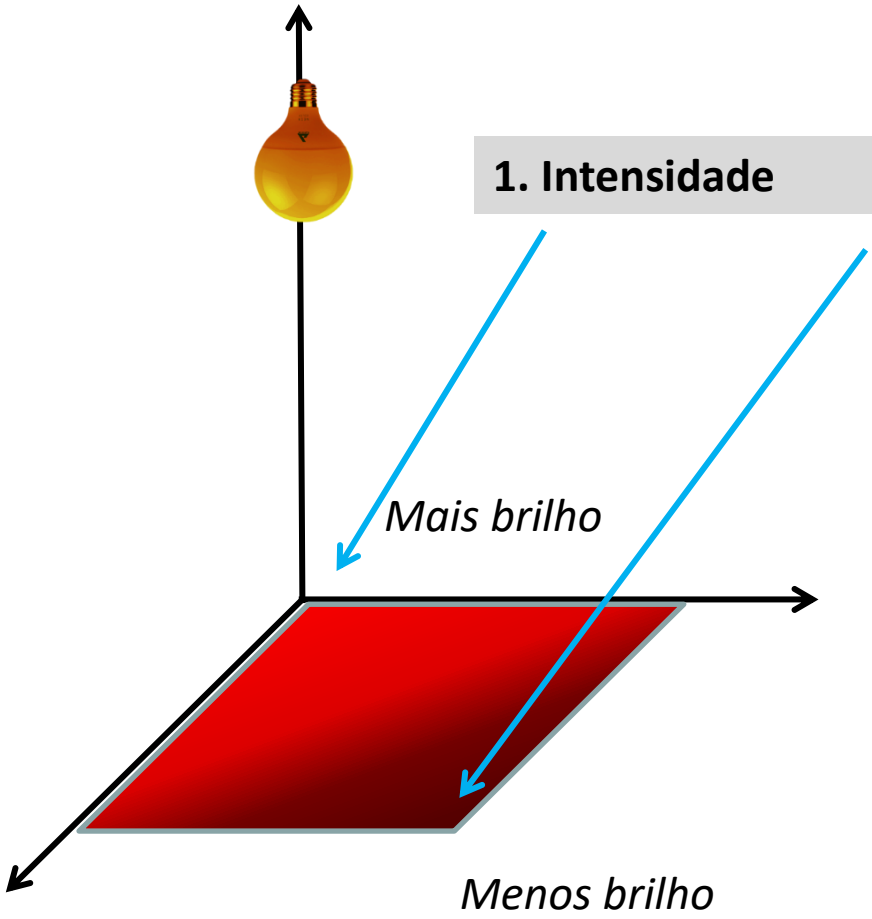
glColor(...)

Calculada com

1. **Proriedades da Luz**
2. **Propriedades reflexão Material**

glMaterial(...)

Ou seja, glColor não funciona com a iluminação !!!



1. Intensidade

2. Cor

Cor Luz x ReflexãoMaterial

Ex.
Luz Amarela =110
Material = Roxo = 011

Cor final = verde = 010

Trabalho

Primeira parte 1: Aprender a definir Luzes e materiais

Modelo de cor (multiplicativo)

//---------- Luz no tecto (pontual)

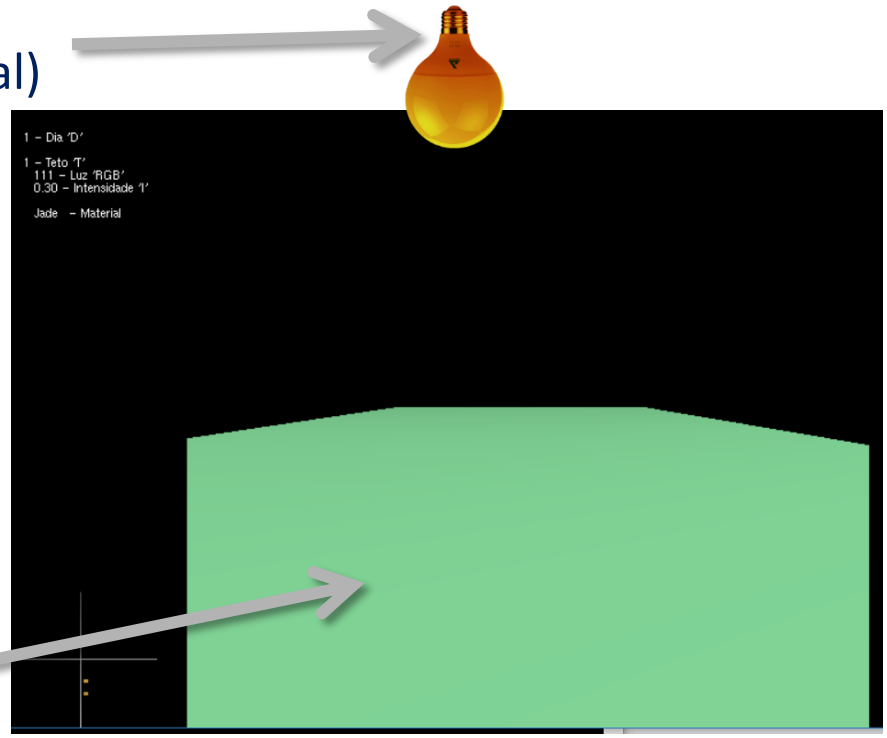
Ligar/desligar: : **'L' {0,1}**

Red: : **'R' {0,1}**

Green: : **'G' {0,1}**

Blue : **'B' {0,1}**

Variação intensidade : **'I' [0..1]**

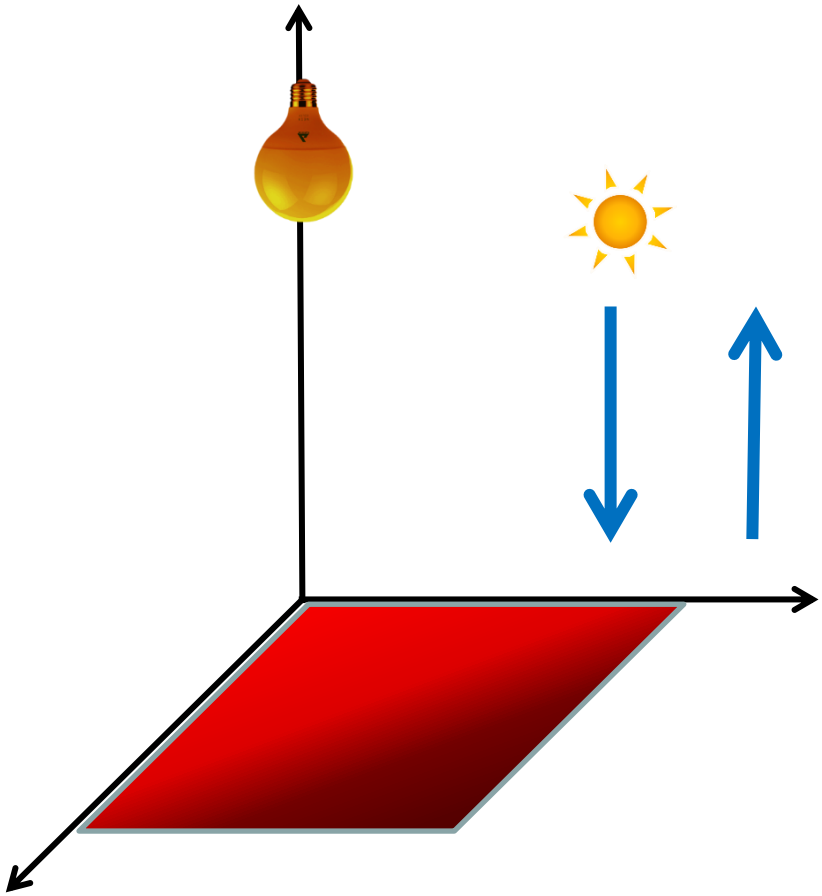


//---------- Material do chão

Cor=Propriedades : **'M'**

// W, B, R, G, B,

// Y, C, M



Sugestão

Definir uma luz direccional
Pode comutar o sentido
- cima para baixo
- baixo para cima

OPENGL – cor & iluminação

Assuntos a tratar neste trabalho

- ***Parte 1 - Básico***
  - Cor: interacção luz + material

- ***Parte 2 - Detalhes***
  - 1. Color Materi
  - 2. Luz d
  - 3
  - com textura
  - forma de atenuação da intensidade …)
  - de polígonos
  - Transparências

Continua ….