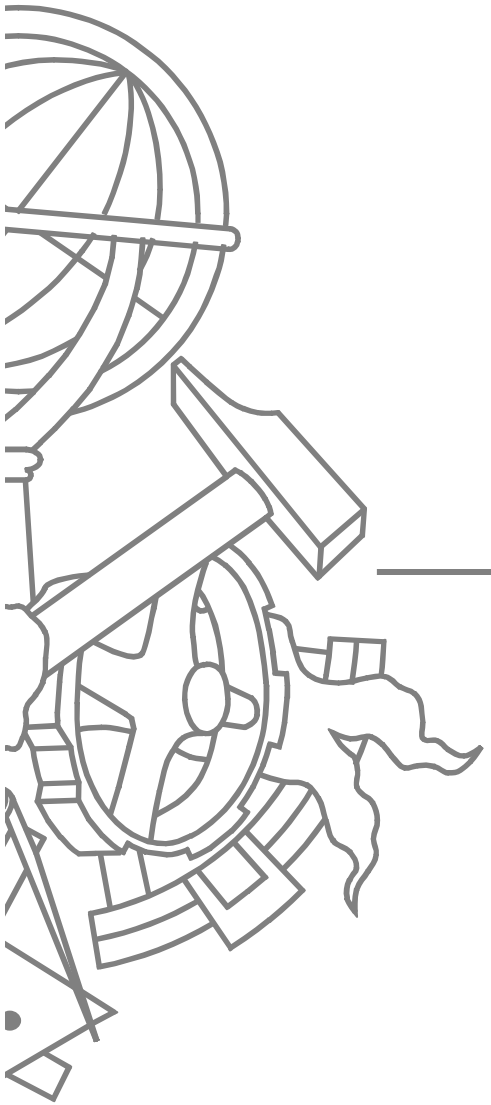


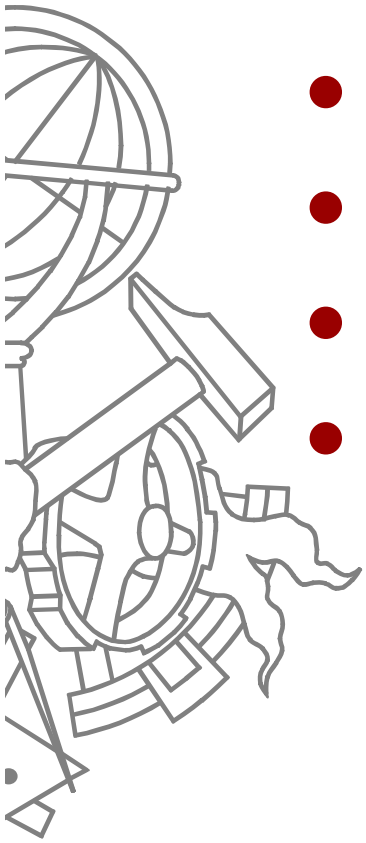
Iluminação

Aula 6

Sistemas Gráficos e Interactivos
Instituto Superior de Engenharia do Porto

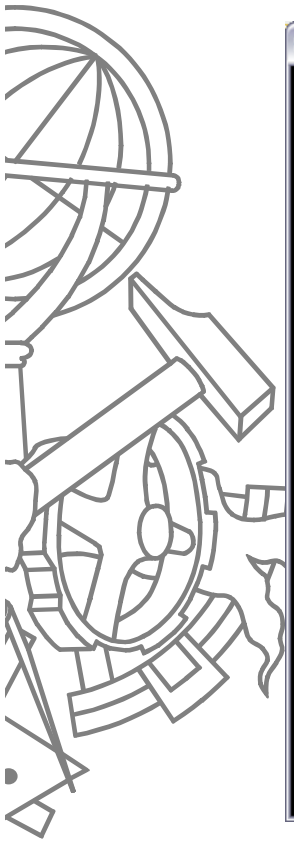
Paulo Gandra de Sousa
psousa@dei.isep.ipp.pt

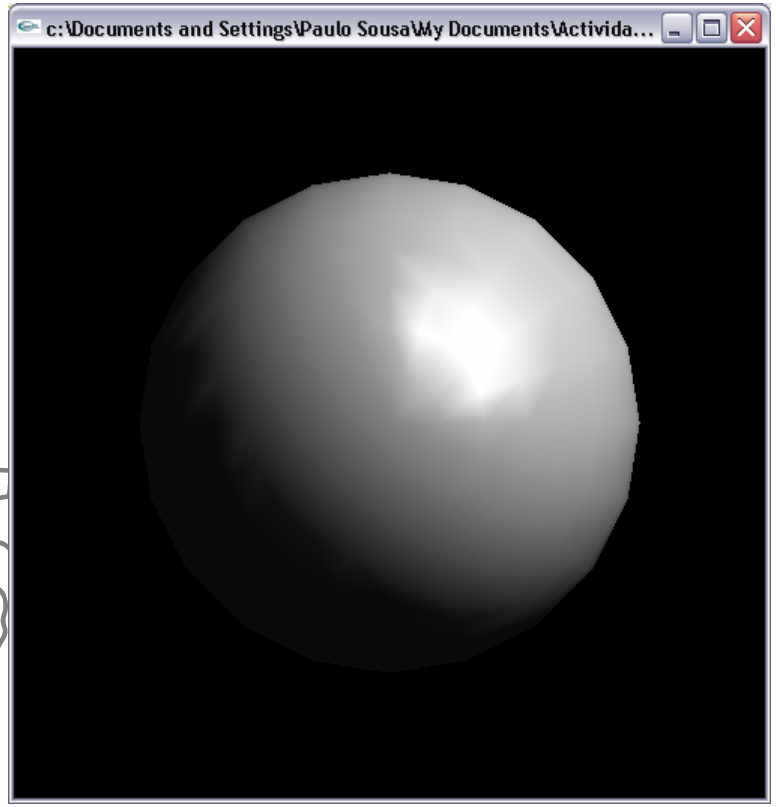
Conteúdo

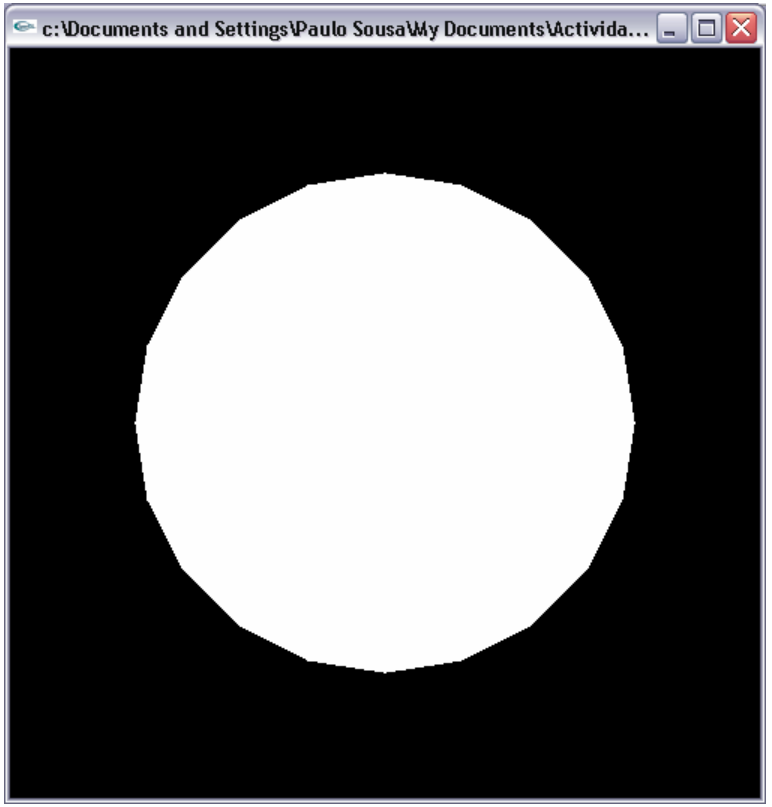


- Tipos de iluminação
- Materiais
- Luzes
- Modelos de iluminação

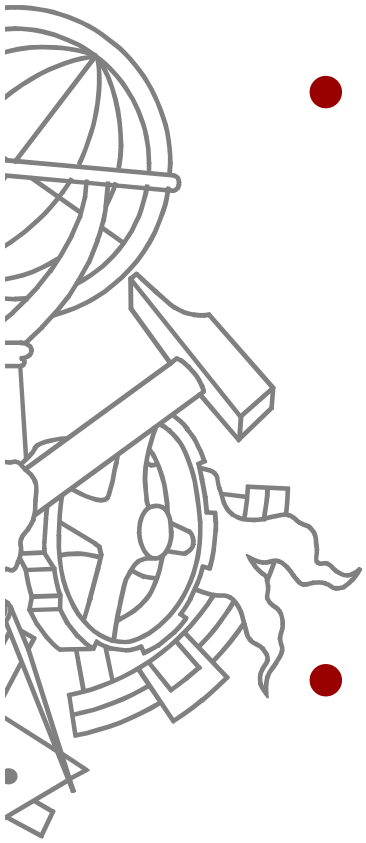
Esfera iluminada ou não





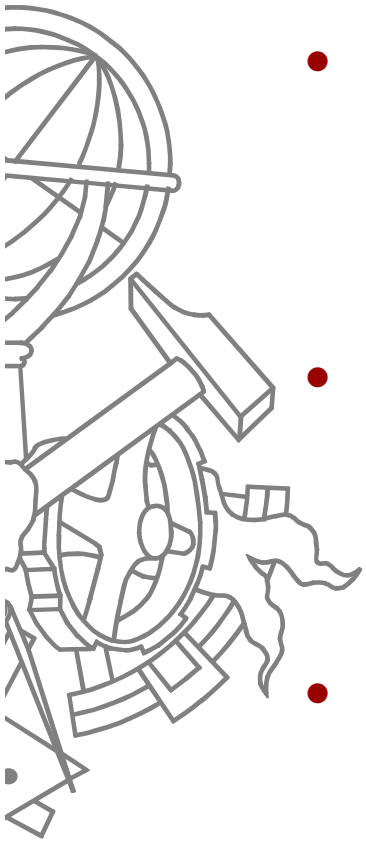


Iluminação

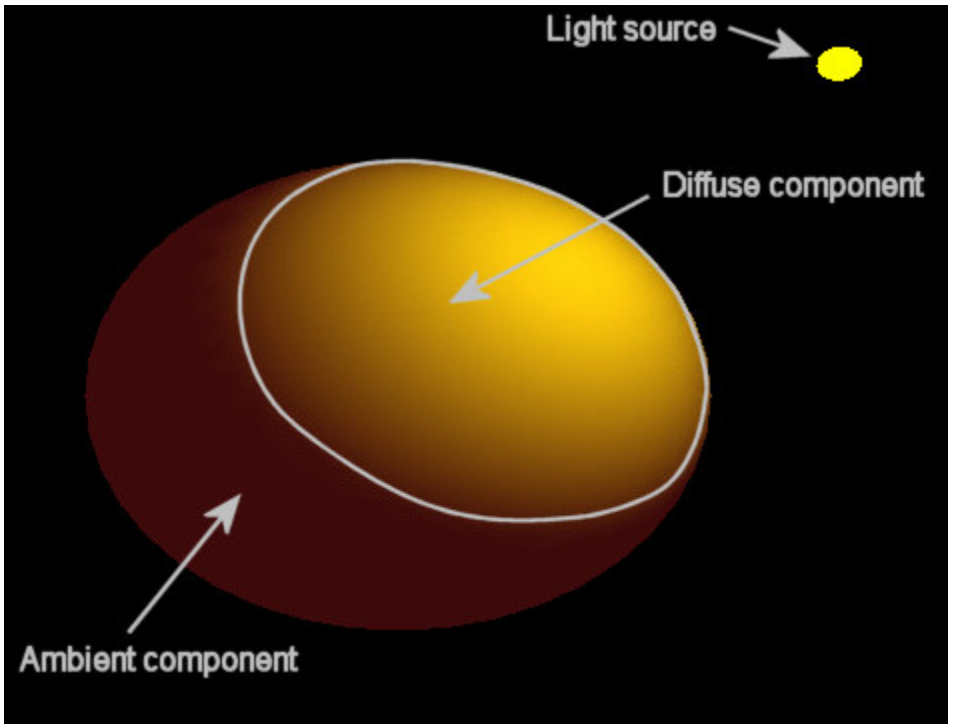


- OpenGL aproxima luz do mundo real em componentes RGB
  - Fontes de luz, emitem luz
  - Objectos (materiais) reflectem luz
    - Espalhando-a genericamente
    - Numa direcção preferencial
- Luz de uma cena é proveniente de várias fontes de luz
  - Posicionais ou ambiente

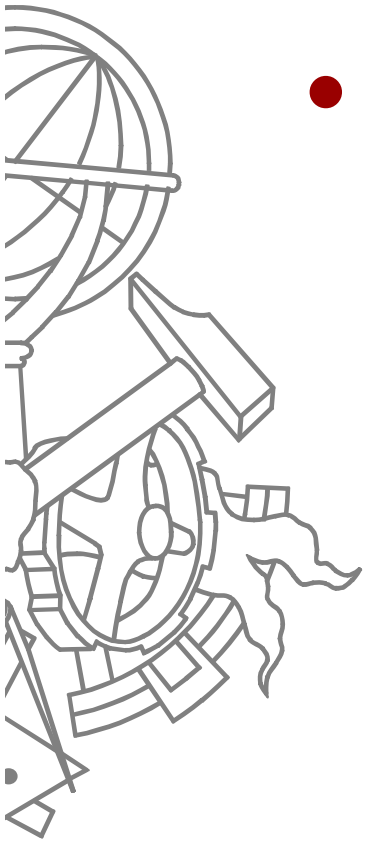
Tipos de iluminação



- **Ambiente**: Luz espalhada uniformemente em todas as direcções; resulta da luz a bater e ser reflectida em superfícies
- **Difusa**: Luz vinda de uma determinada direcção; ao bater numa superfície a luz é espalhada uniformemente
- **Especular**: Luz vinda de uma determinada direcção; ao bater numa superfície a luz é reflectida numa direcção especifica

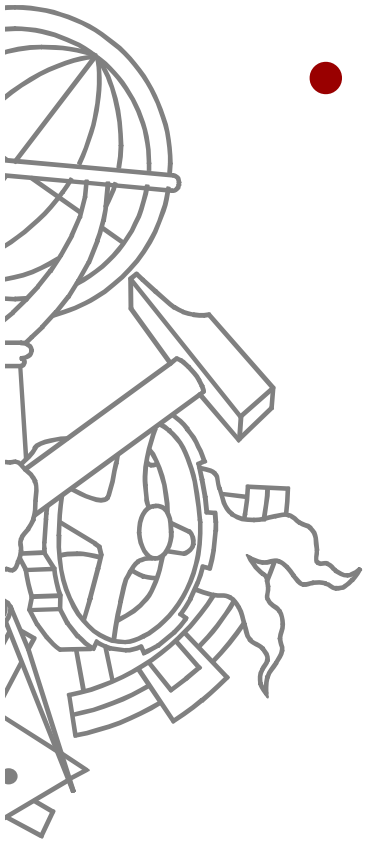


Iluminação em OpenGL



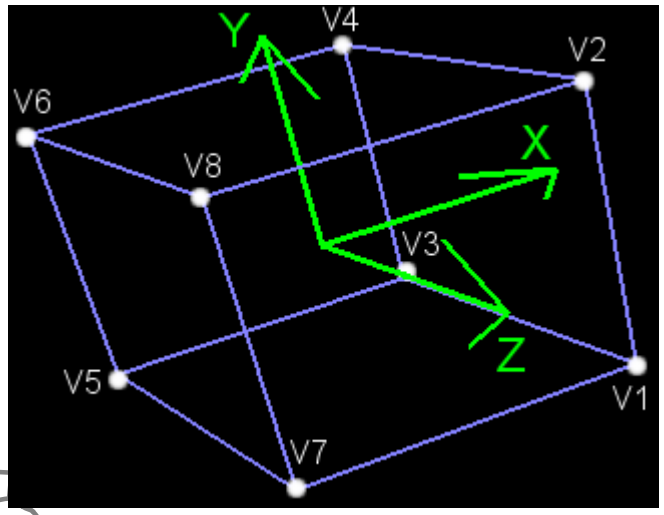
- Passos necessários:
  1. Definir normais para cada vértice (irão determinar a orientação do objecto em relação às fontes de luz)
  2. Configurar e posicionar uma ou mais fontes de luz
  3. Configurar e escolher um modelo de iluminação (nível de luz ambiente e posicionamento do ponto de vista).
  4. Definir propriedades dos materiais que compõem os objectos da cena

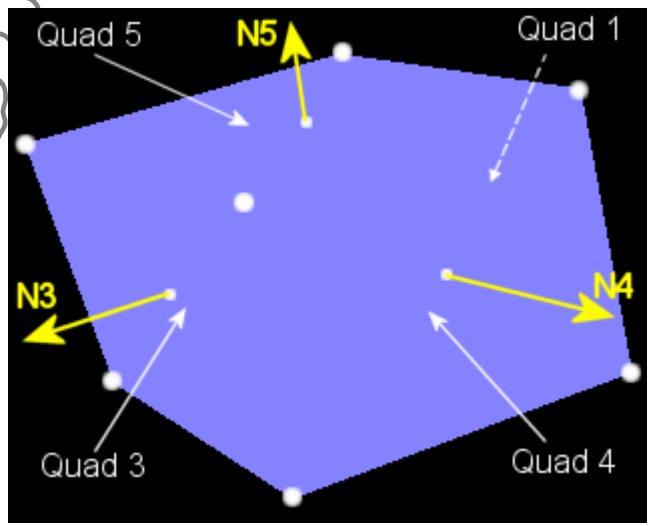
Vector normal

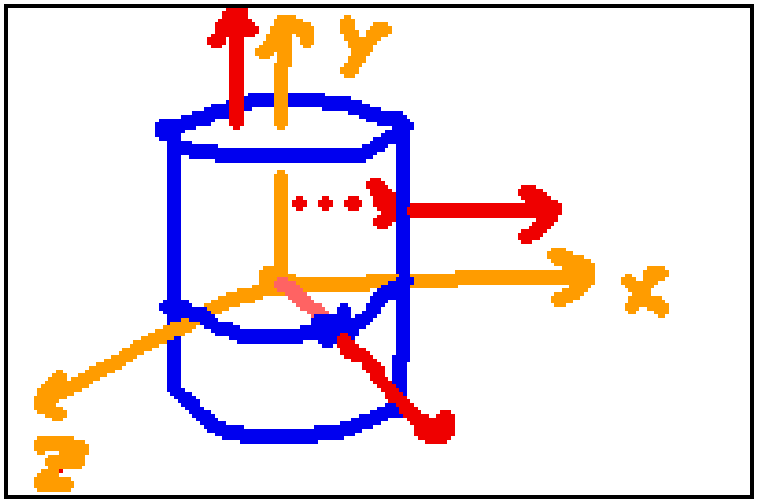


- **`glNormal(x, y, z)`**
  - Utilizado para calcular a maneira como a luz incide na superfície do objecto
  - Define um vector perpendicular à superfície/vértice
    - Invocado dentro de glBegin/glEnd antes de glVertex
  - Deve ter comprimento unitário
    - Dividir cada componente x, y, z pelo comprimento da normal $\sqrt{x^2 + y^2 + z^2}$
    - **`glEnable(GL_AUTONORMALIZE)`**

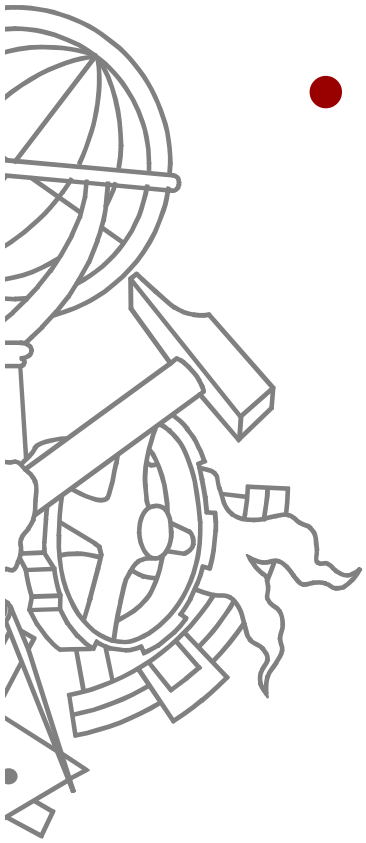
Normais





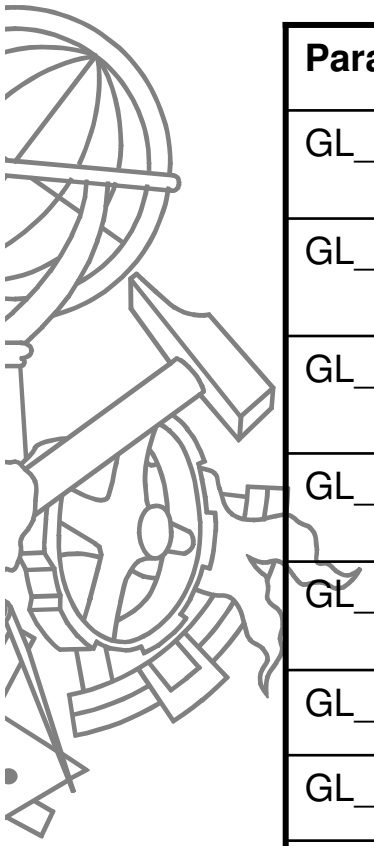


Luzes



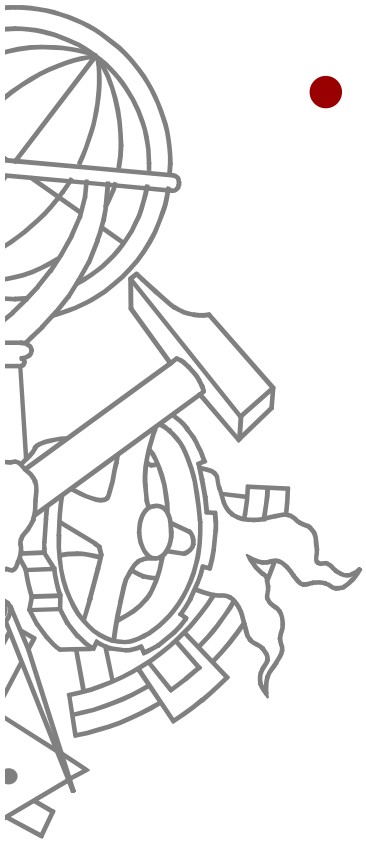
- glLightfv(luz, parâmetro, valor)
  - Luz
    - GL_LIGHT0 .. GL_LIGHT7
  - Parâmetro
    - GL_AMBIENT
    - GL_DIFFUSE
    - GL_SPECULAR
    - GL_POSITION
    - ...

Parâmetro de glLight

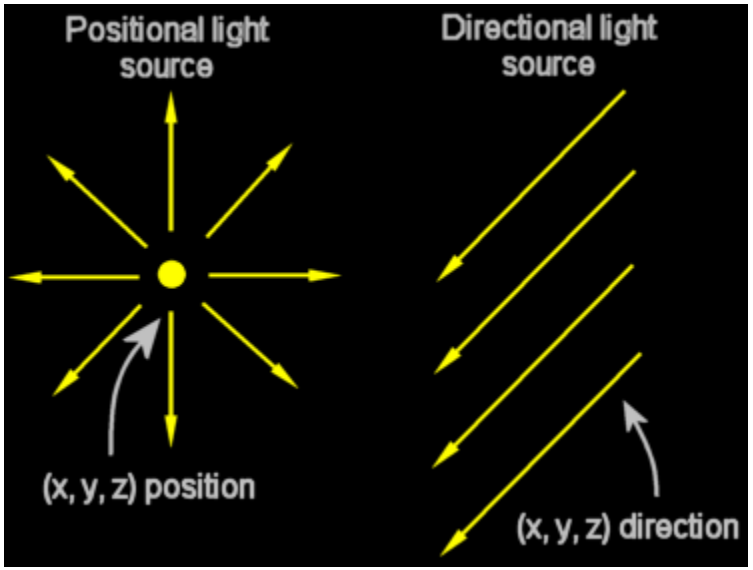


Parameter Name	Default Value	Meaning
GL_AMBIENT	(0.0, 0.0, 0.0, 1.0)	ambient RGBA intensity of light
GL_DIFFUSE	(1.0, 1.0, 1.0, 1.0) – Luz 0 (0.0, 0.0, 0.0, 1.0) – Luz 1 .. 7	diffuse RGBA intensity of light
GL_SPECULAR	(1.0, 1.0, 1.0, 1.0) – Luz 0 (0.0, 0.0, 0.0, 1.0) – Luz 1 .. 7	specular RGBA intensity of light
GL_POSITION	(0.0, 0.0, 1.0, 0.0)	(x, y, z, w) position of light
GL_SPOT_DIRECTION	(0.0, 0.0, -1.0)	(x, y, z) direction of spotlight
GL_SPOT_EXPONENT	0.0	spotlight exponent
GL_SPOT_CUTOFF	180.0	spotlight cutoff angle
GL_LINEAR_ATTENUATION	0.0	linear attenuation factor
GL_QUADRATIC_ATTENUATION	0.0	quadratic attenuation factor

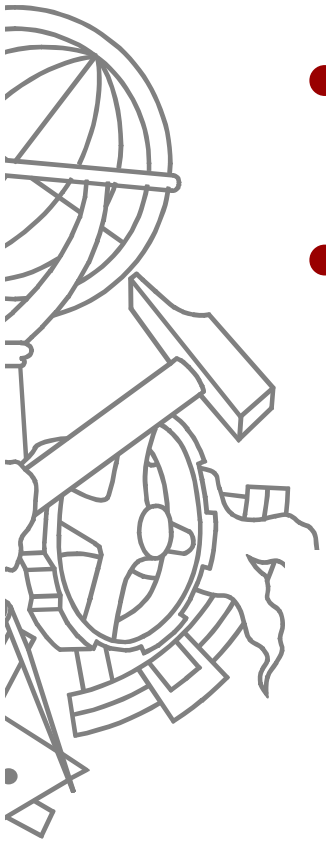
GL_POSITION



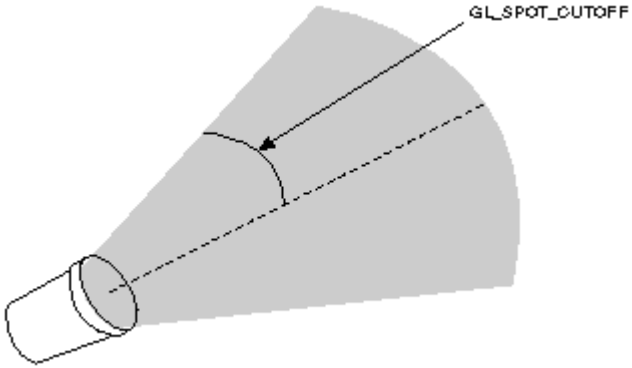
- Luz direccional ou posicional
  - (x, y, z, w)
  - w = 0 ➔ direccional
  - w ≠ 0 ➔ posicional
    - *Eye coordinates*



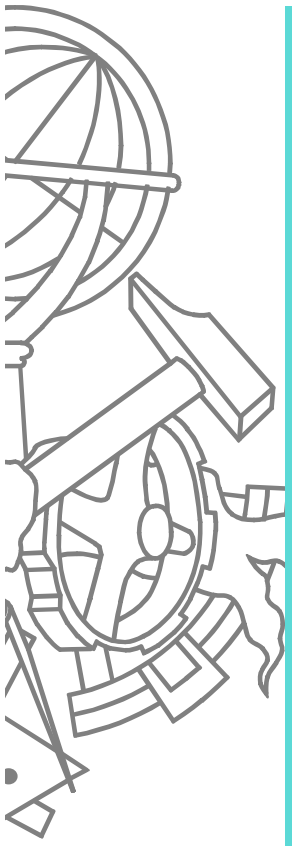
Focos



- Por omissão uma luz emite em todas as direcções
- É possível definir um foco indicando a direcção e a abertura
  - É possível definir a "concentração" de luz no cone GL_SPOT_EXPONENT

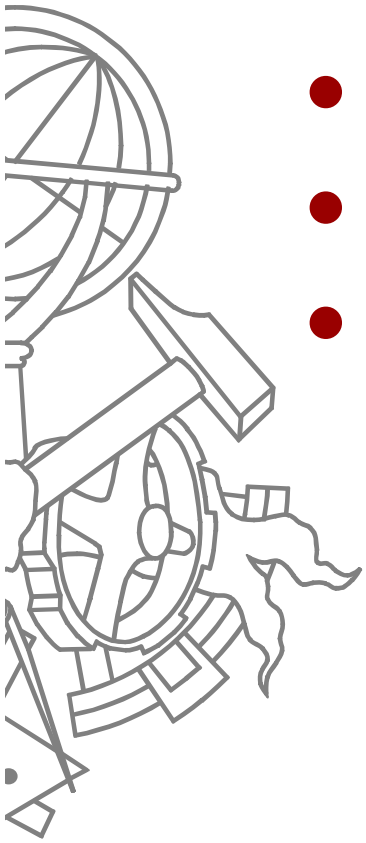


Exemplo



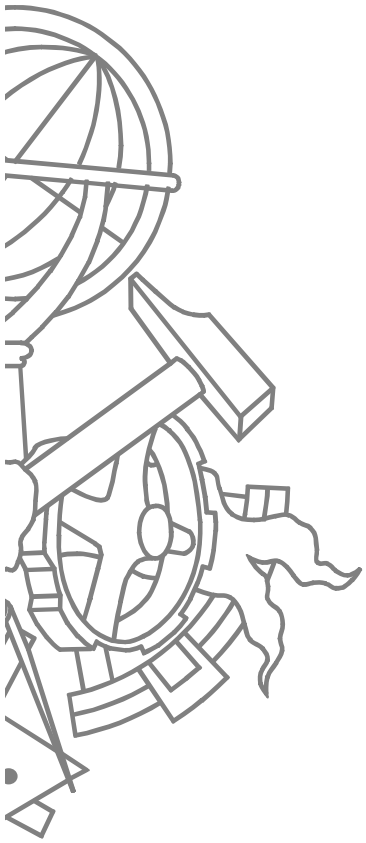
```
void init()
{
   GLfloat light_ambient[] = { 0.0, 0.0, 0.0, 1.0 };
   GLfloat light_diffuse[] = { 1.0, 1.0, 1.0, 1.0 };
   GLfloat light_specular[] = { 1.0, 1.0, 1.0, 1.0 };
   GLfloat light_position[] = { 1.0, 1.0, 1.0, 0.0 };

   ...
   glLightfv(GL_LIGHT0, GL_AMBIENT, light_ambient);
   glLightfv(GL_LIGHT0, GL_DIFFUSE, light_diffuse);
   glLightfv(GL_LIGHT0, GL_SPECULAR, light_specular);
   glLightfv(GL_LIGHT0, GL_POSITION, light_position);
   ...
}
```

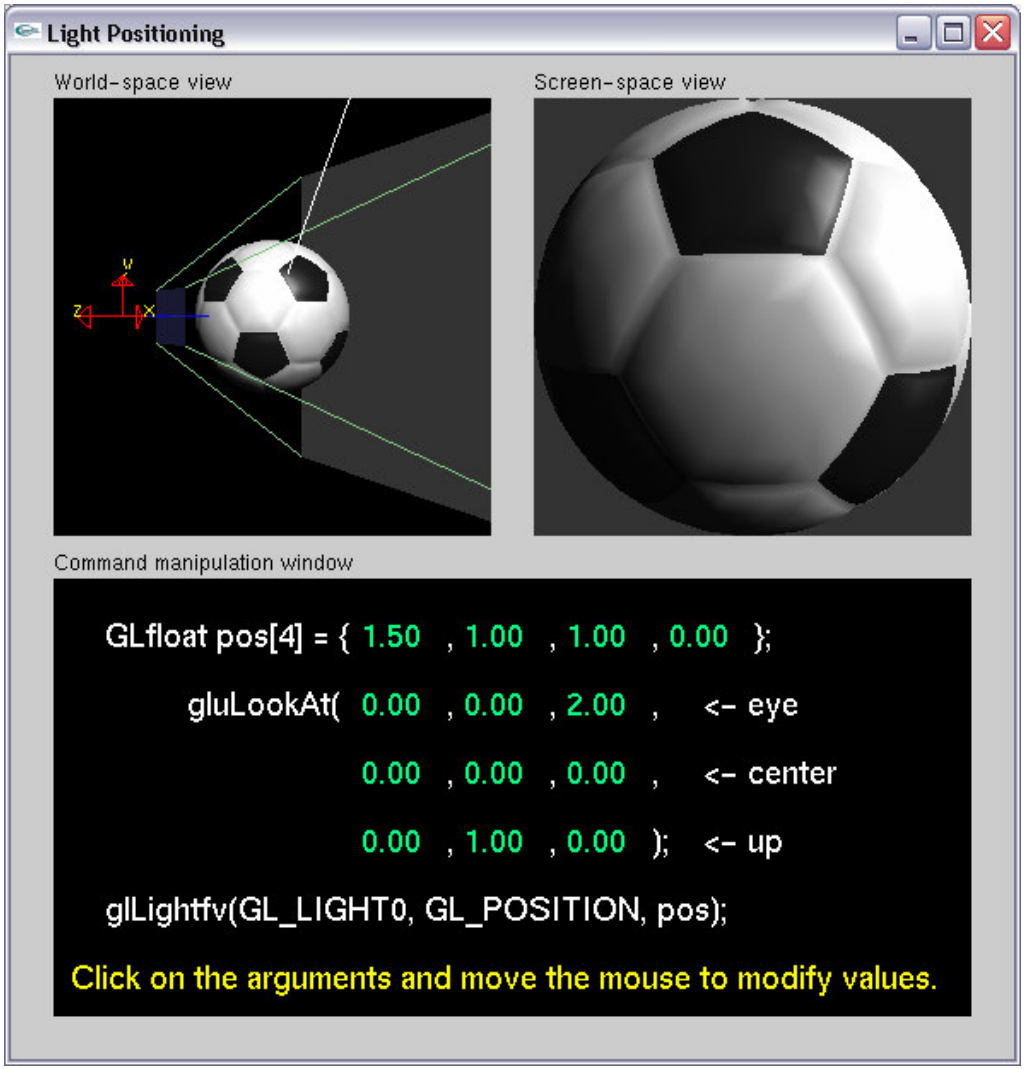
Ligar/desligar iluminação



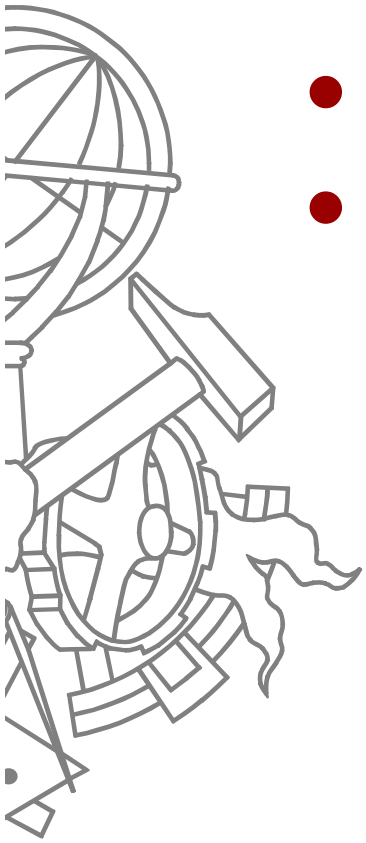
- glEnable(GL_LIGHTING)
- glEnable(GL_LIGHT*n*)
- glDisable(GL_LIGHT*n*)

Demo



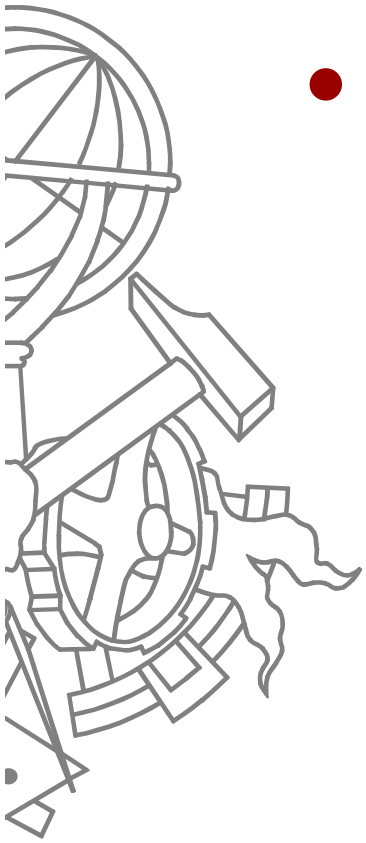

Light Positioning
World-space view
Screen-space view
Command manipulation window
GLfloat pos[4] = { 1.50 , 1.00 , 1.00 , 0.00 };
gluLookAt(0.00 , 0.00 , 2.00 , <- eye
0.00 , 0.00 , 0.00 , <- center
0.00 , 1.00 , 0.00); <- up
glLightfv(GL_LIGHT0, GL_POSITION, pos);
Click on the arguments and move the mouse to modify values.

Modelo de iluminação



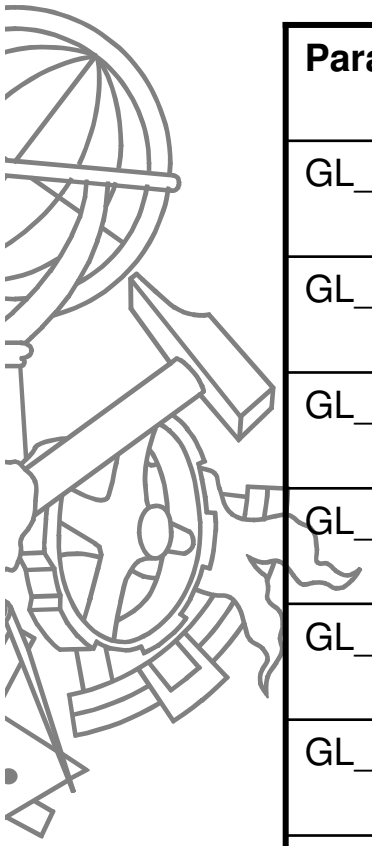
- glLightModel(parâmetro, valor)
- Parâmetro
  - GL_LIGHT_MODEL_AMBIENT
    - Default: (0.2, 0.2, 0.2, 1.0)
  - GL_LIGHT_MODEL_LOCAL_VIEWER
    - Default: GL_FALSE
  - GL_LIGHT_MODEL_TWO_SIDE
    - Default: GL_FALSE

Materiais



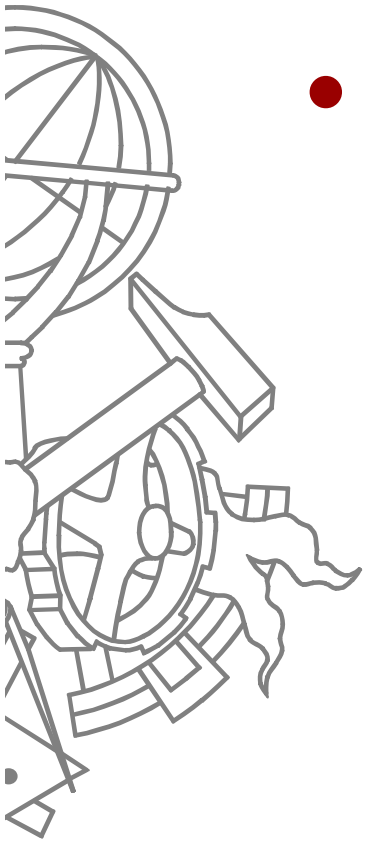
- glMaterial(face, parâmetro, valor)
  - Face
    - GL_FRONT
    - GL_BACK
    - GL_FRONT_AND_BACK
  - Parâmetro
    - GL_AMBIENT
    - GL_DIFFUSE
    - GL_SPECULAR
    - ...

Parâmetro de glMaterial



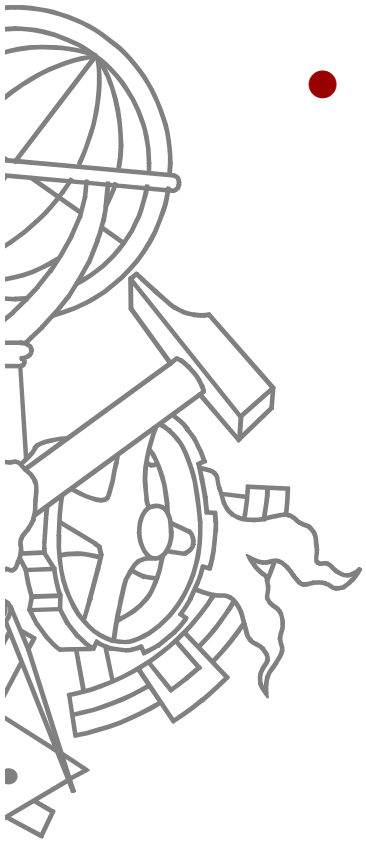
Parameter Name	Default Value	Meaning
GL_AMBIENT	(0.2, 0.2, 0.2, 1.0)	ambient color of material
GL_DIFFUSE	(0.8, 0.8, 0.8, 1.0)	diffuse color of material
GL_AMBIENT_AND_DIFFUSE		ambient and diffuse color of material
GL_SPECULAR	(0.0, 0.0, 0.0, 1.0)	specular color of material
GL_SHININESS	0.0	specular exponent
GL_EMISSION	(0.0, 0.0, 0.0, 1.0)	emissive color of material
GL_COLOR_INDEXES	(0,1,1)	ambient, diffuse, and specular color indices

Materiais



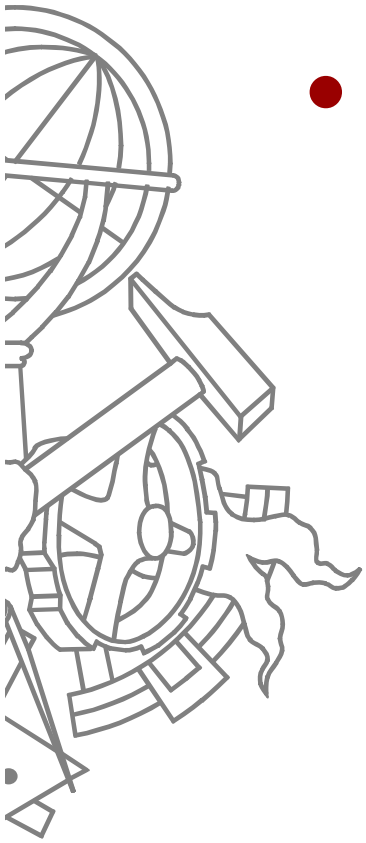
- Possuem componente ambiente, difusa, especular e emissora
  - Ambiente e difusa definem a cor do objecto e são normalmente iguais
  - Especular é normalmente branco para garantir a cor do foco de projecção
  - Emissora simula uma fonte de luz dentro do próprio objecto

Materiais



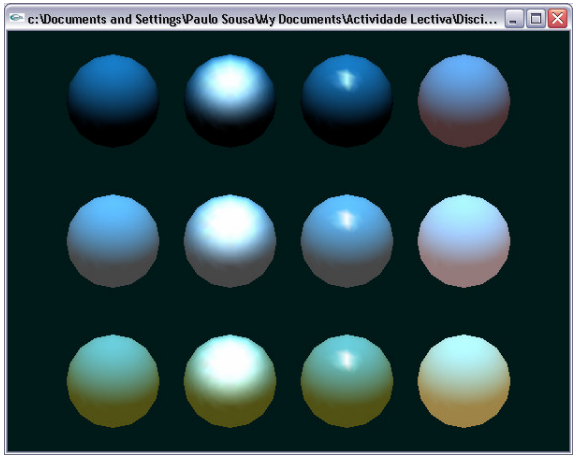
- Percentagem de reflexão de cada componente de cor RGB
  - Exemplo:
    - R = 1, G= 0.5, B = 0
    - Reflecte **todo** o vermelho
    - Reflecte 50% do verde
    - Absorve **todo** o azul
  - Ou seja:
    - Fonte de Luz (LR, LG, LB)
    - Material (MR, MG, MB)
    - “Cor” visível = (LR*MR, LG*MG, LB*MB)

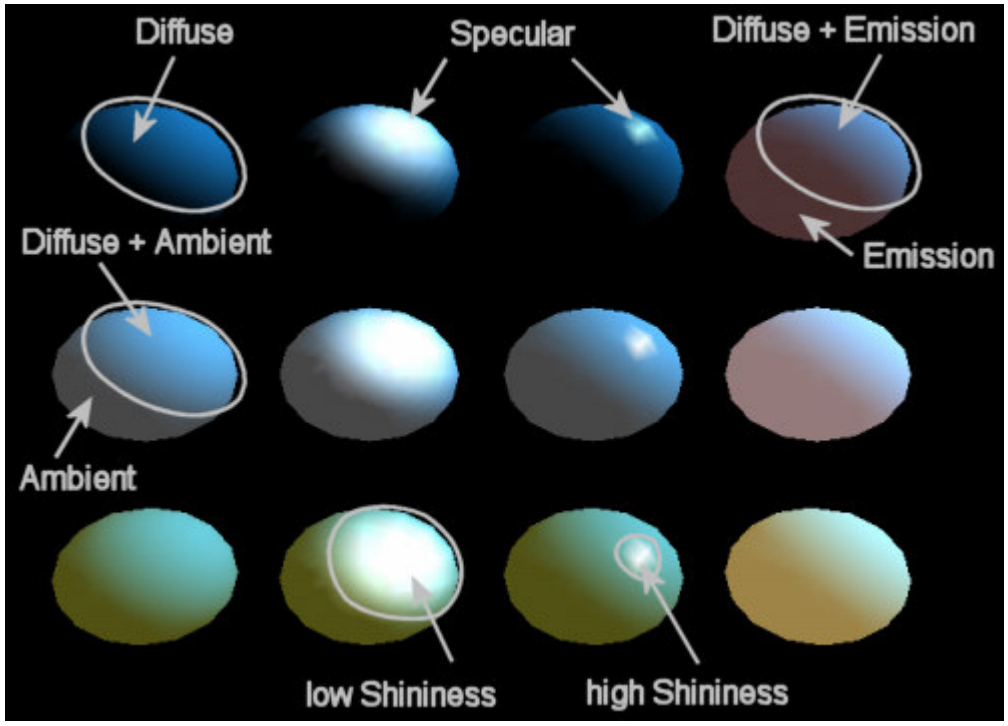
Materiais: exemplo



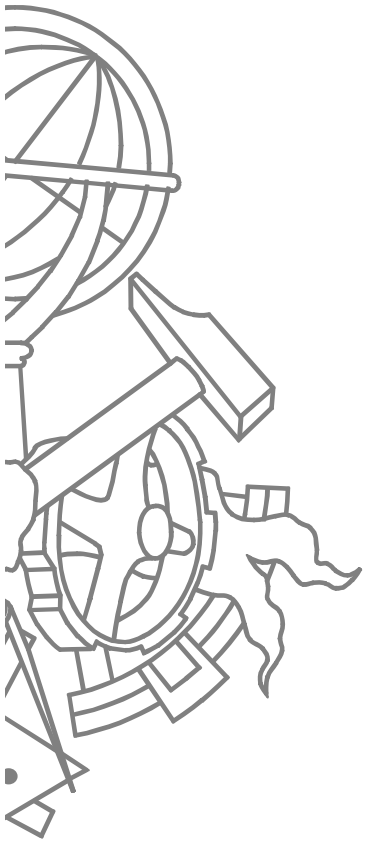
- Objecto vermelho (R=1, G=0, B=0)
  - Luz vermelha ➔ objecto vermelho
  - Luz branca ➔ objecto vermelho
    - Componente vermelha da luz branca é reflectida
  - Luz verde ➔ objecto preto
    - Todo o verde é absorvido

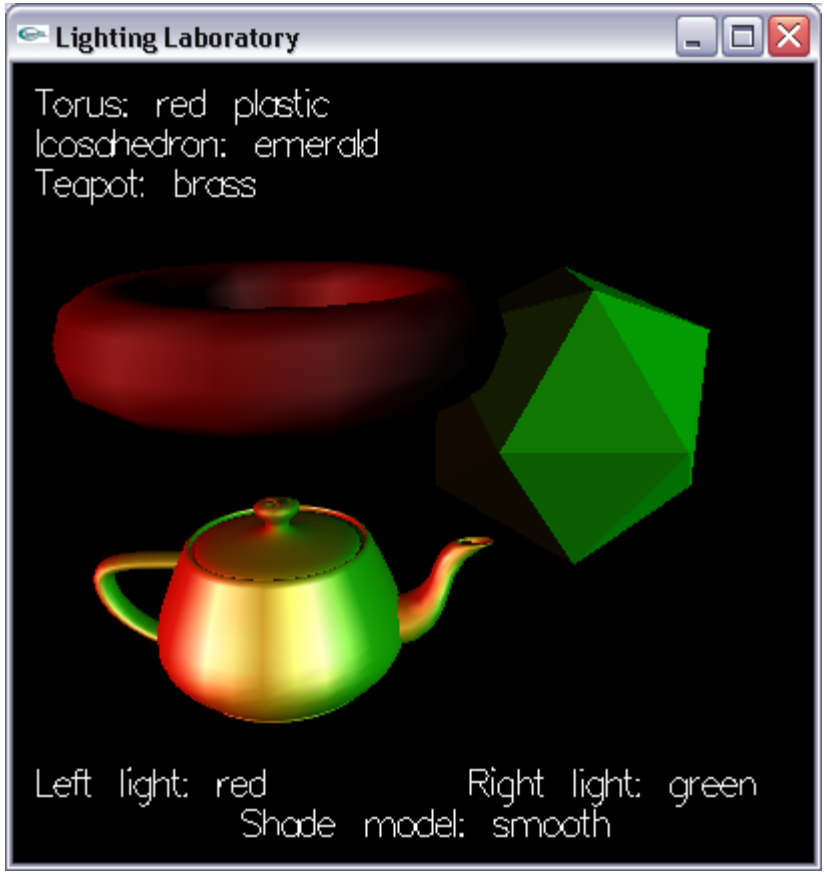
Demo


c:\Documents and Settings\Paulo Sousa\My Documents\Actividade Lectiva\Disci...

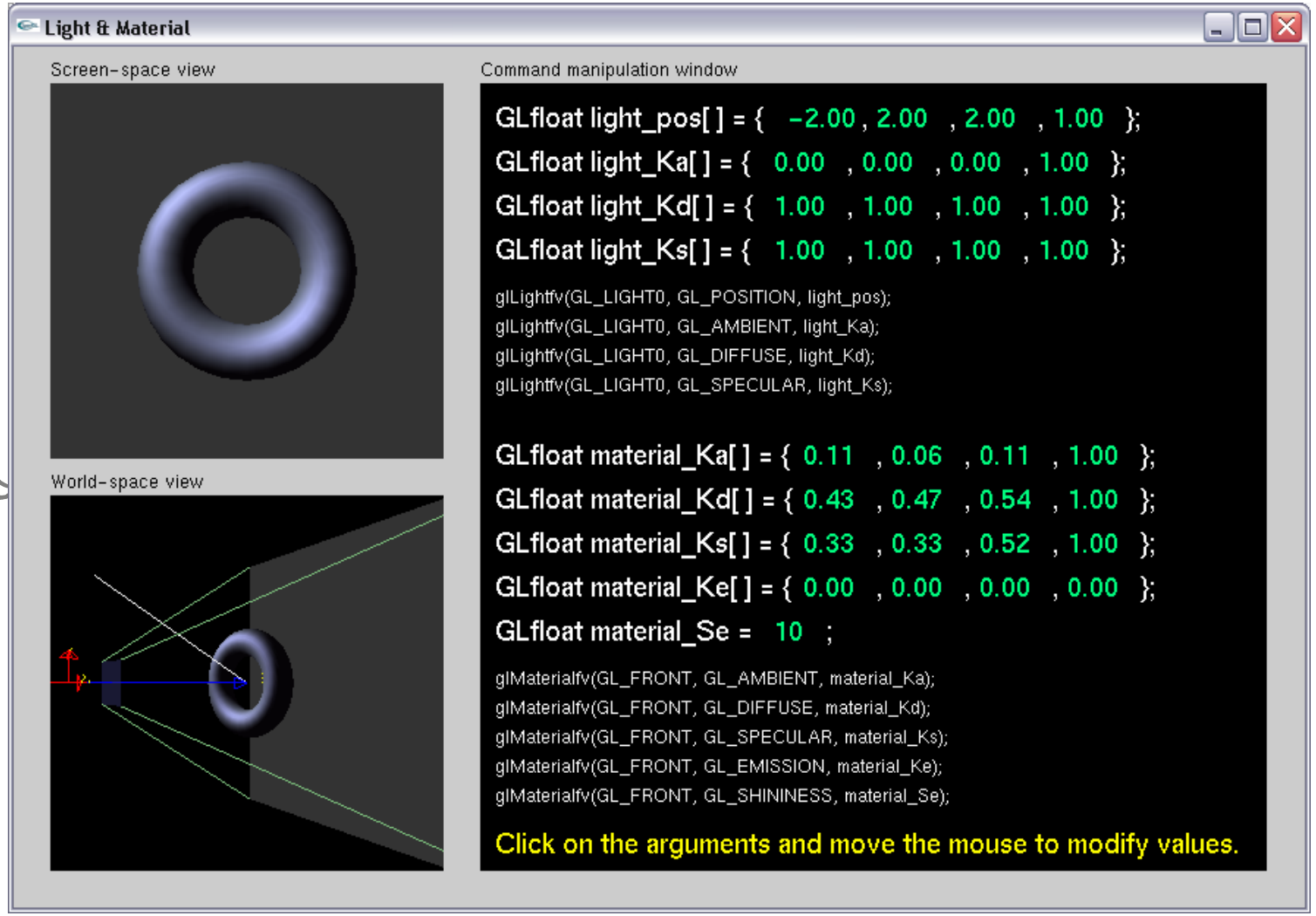

Diffuse
Specular
Diffuse + Emission
Diffuse + Ambient
Emission
Ambient
low Shininess
high Shininess

Light lab

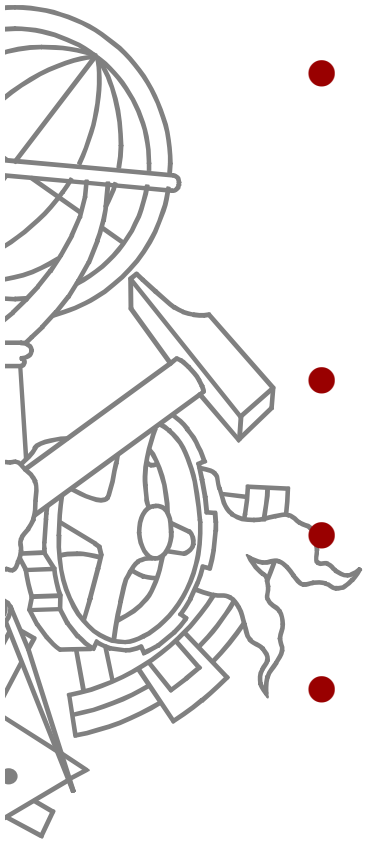



Lighting Laboratory
Torus: red plastic
Icosahedron: emerald
Teapot: brass
Left light: red
Right light: green
Shade model: smooth

Demo


Light & Material
Screen-space view
World-space view
Command manipulation window
GLfloat light_pos[ ] = { -2.00 , 2.00 , 2.00 , 1.00 };
GLfloat light_Ka[ ] = { 0.00 , 0.00 , 0.00 , 1.00 };
GLfloat light_Kd[ ] = { 1.00 , 1.00 , 1.00 , 1.00 };
GLfloat light_Ks[ ] = { 1.00 , 1.00 , 1.00 , 1.00 };
glLightfv(GL_LIGHT0, GL_POSITION, light_pos);
glLightfv(GL_LIGHT0, GL_AMBIENT, light_Ka);
glLightfv(GL_LIGHT0, GL_DIFFUSE, light_Kd);
glLightfv(GL_LIGHT0, GL_SPECULAR, light_Ks);
GLfloat material_Ka[ ] = { 0.11 , 0.06 , 0.11 , 1.00 };
GLfloat material_Kd[ ] = { 0.43 , 0.47 , 0.54 , 1.00 };
GLfloat material_Ks[ ] = { 0.33 , 0.33 , 0.52 , 1.00 };
GLfloat material_Ke[ ] = { 0.00 , 0.00 , 0.00 , 0.00 };
GLfloat material_Se = 10 ;
glMaterialfv(GL_FRONT, GL_AMBIENT, material_Ka);
glMaterialfv(GL_FRONT, GL_DIFFUSE, material_Kd);
glMaterialfv(GL_FRONT, GL_SPECULAR, material_Ks);
glMaterialfv(GL_FRONT, GL_EMISSION, material_Ke);
glMaterialfv(GL_FRONT, GL_SHININESS, material_Se);
Click on the arguments and move the mouse to modify values.

Posicionar luzes na cena



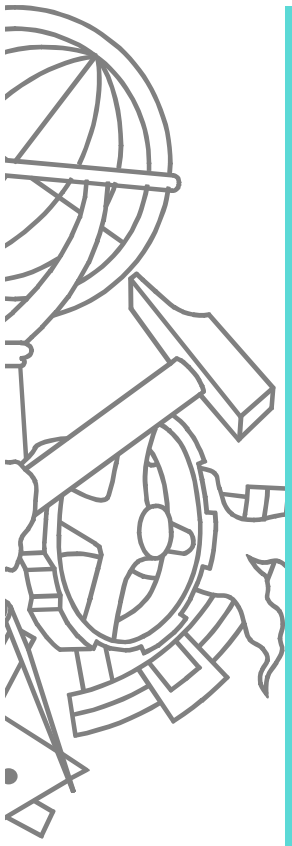
- Em OpenGL a posição de uma fonte de luz é tratada como uma primitiva, sendo por isso transformada pela matriz de modelo/vista

- Luz fixa
  - Definir posição da luz após as transformações de vista
- Luz móvel
  - Aplicar transformação antes de definir posição da luz
- Luz que acompanha o ponto de vista
  - Definir posição da luz **antes** de qualquer transformação (i.e., a seguir a glLoadIdentity)
  - Modificar o ponto de vista usando gluLookAt

Luz fixa

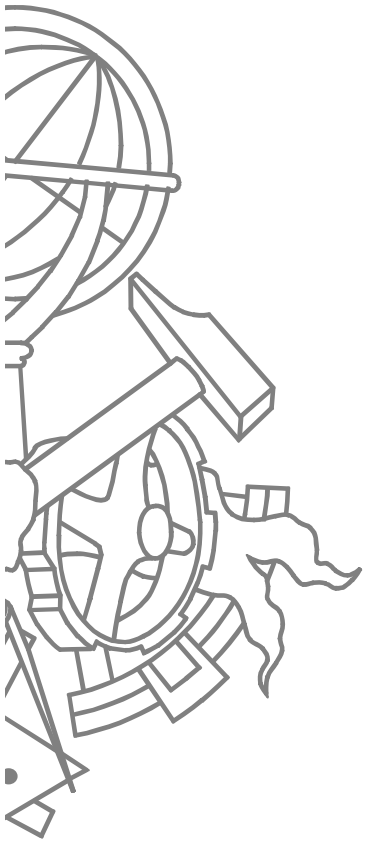
```
void reshape(int w, int h)
{
glViewport (0, 0, (GLsizei) w, (GLsizei) h);
glMatrixMode (GL_PROJECTION);
glLoadIdentity();
if (w <= h)
   glOrtho (-1.5, 1.5, -1.5*h/w, 1.5*h/w, -10.0, 10.0);
else
   glOrtho (-1.5*w/h, 1.5*w/h, -1.5, 1.5, -10.0, 10.0);
glMatrixMode (GL_MODELVIEW);
glLoadIdentity();

//viewing transformation
...

/* later in init() */
GLfloat light_position[] = { 1.0, 1.0, 1.0, 1.0 };
glLightfv(GL_LIGHT0, GL_POSITION, position);
}
```

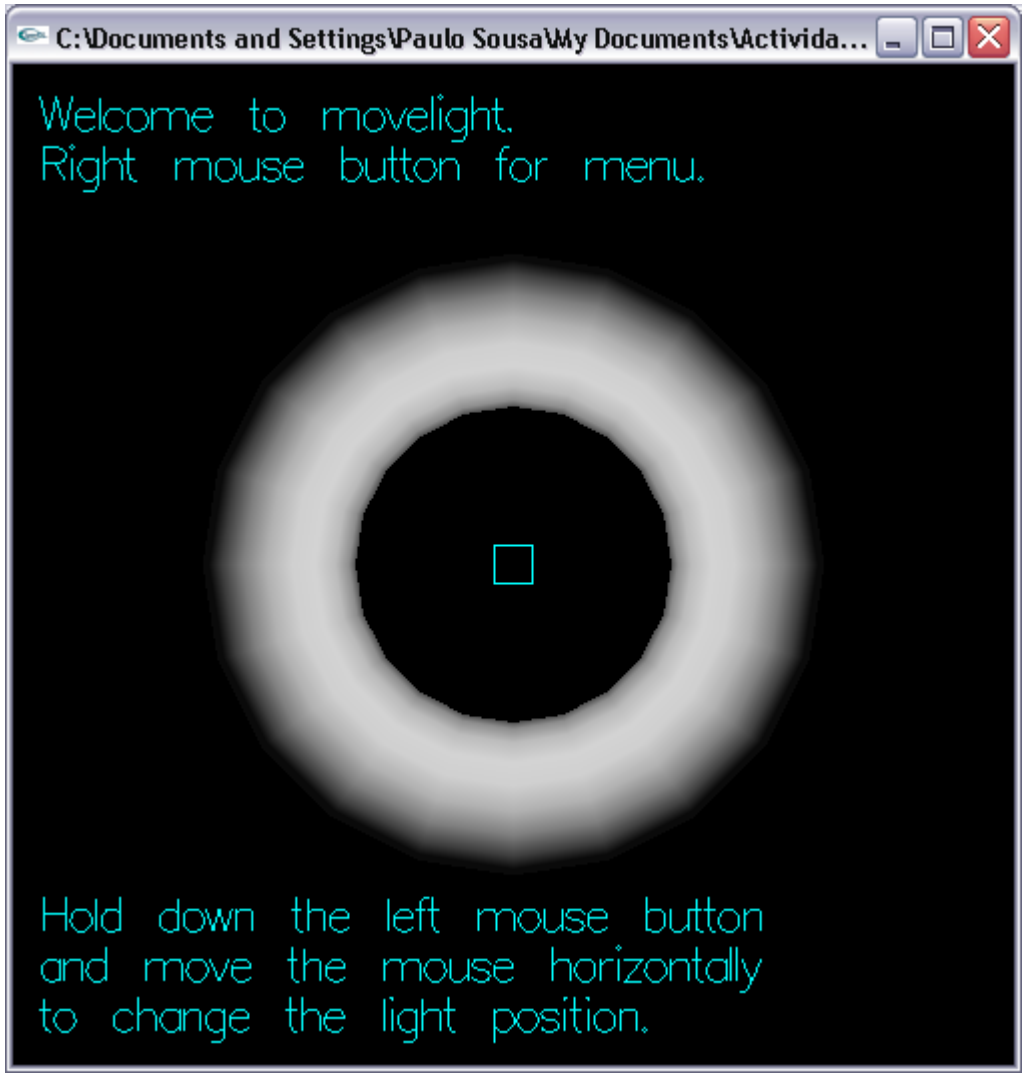
Luz móvel



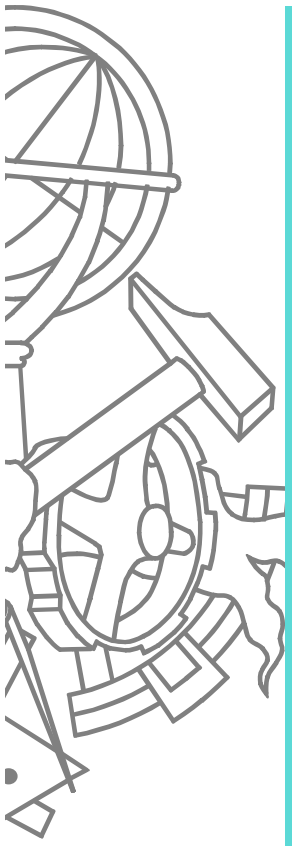
```
static GLdouble spin;

void display(void)
{
GLfloat light_position[] = { 0.0, 0.0, 1.5, 1.0 };
glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
glPushMatrix();
   // viewing transformation
   gluLookAt(0.0, 0.0, 5.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0);

   // light position
   glPushMatrix();
        glRotated(spin, 1.0, 0.0, 0.0);
        glLightfv(GL_LIGHT0, GL_POSITION, light_position);
   glPopMatrix();

// model
glutSolidTorus (0.275, 0.85, 8, 15);
glPopMatrix();
glFlush();
}
```

Demo



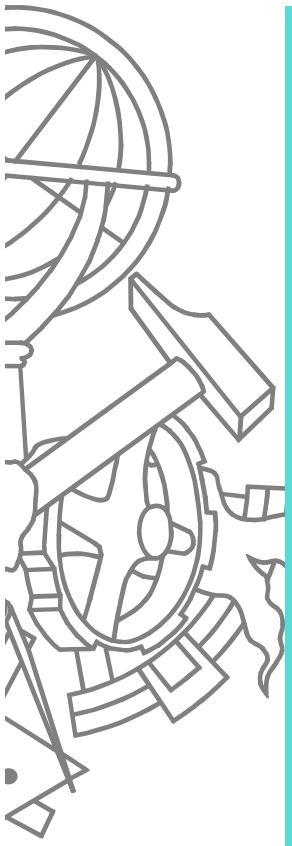

C:\Documents and Settings\Paulo Sousa\My Documents\Actividа...
Welcome to movelight.
Right mouse button for menu.
Hold down the left mouse button
and move the mouse horizontally
to change the light position.

Luz no ponto de vista



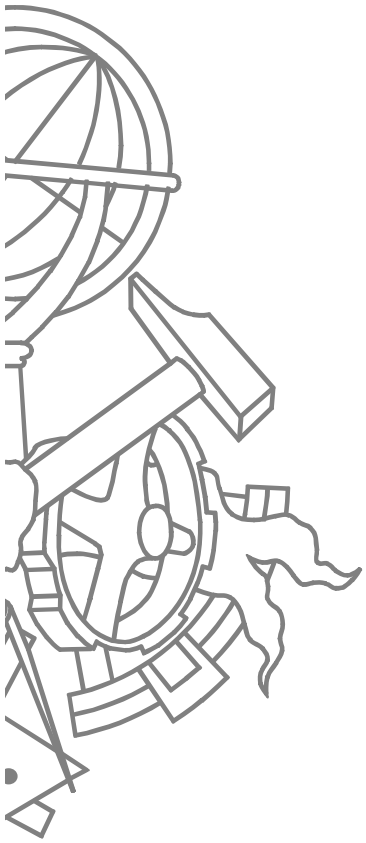
```
void reshape (int w, int h)
{
glViewport(0, 0, (GLint) w, (GLint) h);
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
gluPerspective(40.0, (GLfloat) w/(GLfloat) h, 1.0, 100.0);
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();

//light position before viewing transformation
GLfloat light_position() = {0.0, 0.0, 0.0, 1.0};
glLightfv(GL_LIGHT0, GL_POSITION, light_position);
}
```

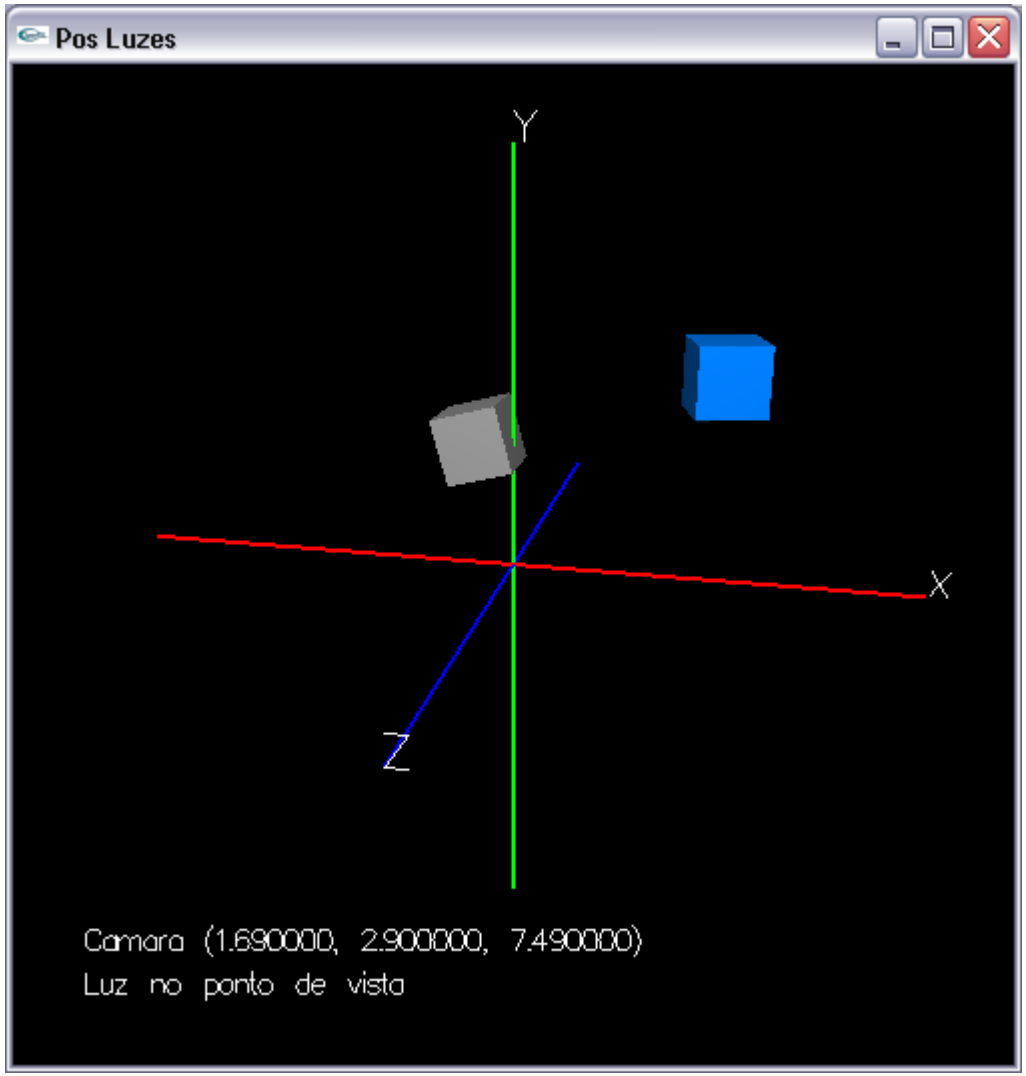
Luz no ponto de vista



```
static GLdouble ex, ey, ez, upx, upy, upz;

void display(void)
{
glClear(GL_COLOR_BUFFER_MASK | GL_DEPTH_BUFFER_MASK);
glPushMatrix();
   gluLookAt (ex, ey, ez, 0.0, 0.0, 0.0, upx, upy, upz);
   glutSolidTorus (0.275, 0.85, 8, 15);
glPopMatrix();
glFlush();
}
```

Demo




Pos Luzes
Y
X
Z
Camara (1.690000, 2.900000, 7.490000)
Luz no ponto de vista