

## Sistemas Gráficos e Interação

Época de Recurso

2017-02-08

N.º \_\_\_\_\_ Nome \_\_\_\_\_

**Duração da prova:** 75 minutos

**Cotação de cada pergunta:** assinalada com parêntesis rectos

**Perguntas de escolha múltipla:** cada resposta incorrecta desconta 1/3 do valor da pergunta

### Parte Teórica

30%

a. [2.5] Qual a dimensão em bytes de um *frame buffer* RGB de 1024 x 512 x 8 bits?

- i. 512 byte
- ii. 512 Kilobyte
- iii. 512 Megabyte
- iv. Nenhuma das anteriores

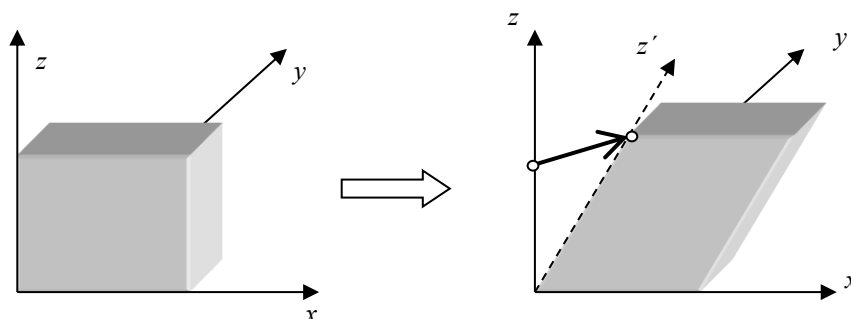
b. [2.5] Qual das seguintes matrizes representa o ponto de coordenadas (3, -1, 8)?

- i.  $[3.0, -1.0, 8.0, 0.0]^T$
- ii.  $[3.0, -1.0, 8.0, 1.0]^T$
- iii.  $[3.0, -1.0, 8.0, 2.0]^T$
- iv. Nenhuma das anteriores

c. [2.5] Qual das seguintes transformações é homotética?

- i. `glScaled(1.0, 2.0, 3.0);`
- ii. `glScaled(2.0, 2.0, 2.0);`
- iii. `glScaled(3.0, 2.0, 1.0);`
- iv. Nenhuma das anteriores

d. [2.5] Qual das seguintes transformações permite deformar o objecto da maneira ilustrada na figura?



- i. Translação
- ii. Rotação
- iii. Escalamento
- iv. Nenhuma das anteriores

e. [2.5] Qual das seguintes equações descreve implicitamente a superfície de uma esfera unitária centrada na origem?

i.  $x^2 + y^2 - z^2 = 0$

ii.  $x^2 + y^2 + z^2 - 1 = 0$

iii.  $\frac{x^2}{4} + \frac{y^2}{9} + \frac{z^2}{16} = 1$

iv. Nenhuma das anteriores

f. [2.5] Uma forma de determinar o vector normal a um polígono planar consiste em

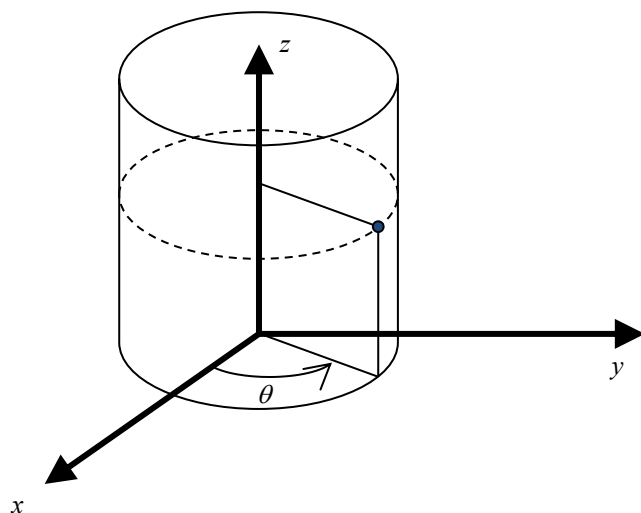
i. Calcular o produto vectorial dos vectores definidos por duas arestas do polígono e dividir as componentes do vector resultante pelo comprimento deste último

ii. Projectar o polígono nos planos OYZ, OZX e OXY e calcular as áreas dos polígonos resultantes; as componentes da normal serão proporcionais a estes valores

iii. As respostas i. e ii.

iv. Nenhuma das anteriores

g. [2.5] A função de mapeamento de texturas que a seguir se discrimina baseia-se numa parametrização



$$\begin{aligned} x &= \cos\theta \\ y &= \sin\theta \\ z &= z \end{aligned} \quad \begin{aligned} \theta &= 2\pi \cdot s \\ z &= t \end{aligned}$$

i. Esférica

ii. Cilíndrica

iii. Cúbica

iv. Nenhuma das anteriores

h. [2.5] O mecanismo de mapeamento de texturas do OpenGL permite

i. A geração automática de coordenadas de textura

ii. Diversos modos de filtragem

iii. Que texturas de diferentes níveis de resolução sejam aplicadas de forma adaptativa

iv. Todas as anteriores

## Sistemas Gráficos e Interação

Época de Recurso

2017-02-08

Parte Teórico-Prática

40%

**Resolução:** Em folhas próprias e separadas (A4). Não se esqueça de indicar, em cada folha, o número de estudante e o nome completo

**Perguntas de escolha múltipla:** cada resposta incorrecta desconta 1/3 do valor da pergunta

**Nota:** Em todas as perguntas, a menos que algo seja dito em contrário, assuma a posição da câmara por omissão

- a. **[3.0]** Considere o modelo de um cilindro centrado no eixo dos Z. Complete o seguinte excerto de código em C, na parte assinalada com `/* CÓDIGO */`, para desenhar as paredes laterais do referido modelo. Defina normais unitárias de forma a que a superfície não exiba arestas significativas. Use apenas as seguintes funções do OpenGL: `glBegin()`, `glVertex3f()`, `glNormal3f()`, `glEnd()`.

```
z1 = 0.0; z2 = 10.0;
alpha = 0.0; alpha_inc = (2.0 * M_PI) / slices;
x1 = radius * cos(alpha); y1 = radius * sin(alpha);

for (i = 0; i < slices; i++) {
    alpha += alpha_inc;
    x2 = radius * cos(alpha); y2 = radius * sin(alpha);

    glBegin(GL_QUADS);
        glNormal3f(x1 / radius, y1 / radius, 0.0);
        glVertex3f(x1, y1, z2);
        glVertex3f(x1, y1, z1);
        glNormal3f(x2 / radius, y2 / radius, 0.0);
        glVertex3f(x2, y2, z1);
        glVertex3f(x2, y2, z2);
    glEnd();
}
```

ou

```
glBegin(GL_QUADS);
    glNormal3f(cos(alpha - alpha_inc), sin(alpha - alpha_inc),
0.0);
    glVertex3f(x1, y1, z2);
    glVertex3f(x1, y1, z1);
    glNormal3f(cos(alpha), sin(alpha), 0.0);
    glVertex3f(x2, y2, z1);
    glVertex3f(x2, y2, z2);
glEnd();
```

ou se se colocar a seguinte normal antes do ciclo for()

```
glNormal3f(cos(alpha), sin(alpha), 0.0);
```

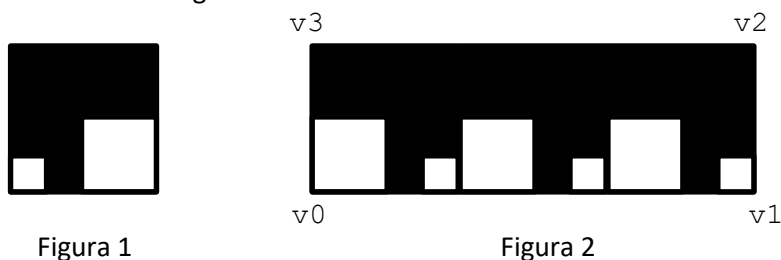
podemos usar só uma normal dentro do ciclo for()

```
glBegin(GL_QUADS);
    glVertex3f(x1, y1, z2);
    glVertex3f(x1, y1, z1);
```

```
glNormal3f(cos(alpha), sin(alpha), 0.0);
glVertex3f(x2, y2, z1);
glVertex3f(x2, y2, z2);
glEnd();

x1 = x2; y1 = y2;
}
```

- b. **[3.0]** Aplique a textura apresentada na Figura 1 a um rectângulo, de modo a ficar com o aspecto apresentado na Figura 2.



```
glTexCoord2f(3.0, 0.0);
glVertex3fv(v0);
glTexCoord2f(0.0, 0.0);
glVertex3fv(v1);
glTexCoord2f(0.0, 1.0);
glVertex3fv(v2);
glTexCoord2f(3.0, 1.0);
glVertex3fv(v3);
```

ou qualquer outra solução idêntica, desfasada em valores inteiros em s ou t

- c. **[2.0]** Indique os parâmetros de configuração de texturas que é necessário realizar para o código acima funcionar correctamente.

Activar o modo de repetição do padrão de textura no eixo s (ou seja, configurar `GL_TEXTURE_WRAP_S` como `GL_REPEAT`).

**Nota:** mude para uma nova folha de respostas

- d. **[3.0]** Observe o seguinte objecto 3D articulado.

As coordenadas do ponto central do corpo são indicadas por:

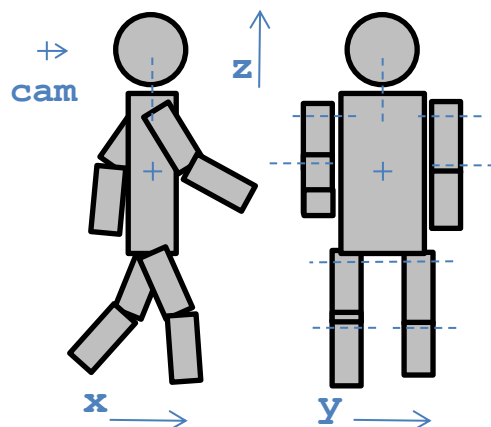
```
(modelo.corpo.x, modelo.corpo.y,
modelo.corpo.z);
```

o corpo roda apenas em torno do eixo dos Z:

```
modelo.corpo.dir (em radianos);
```

a cabeça roda apenas em torno do eixo dos Z  
**relativamente** ao corpo:

```
modelo.corpo.head_dir (em radianos);
```



## Sistemas Gráficos e Interacção

Época de Recurso

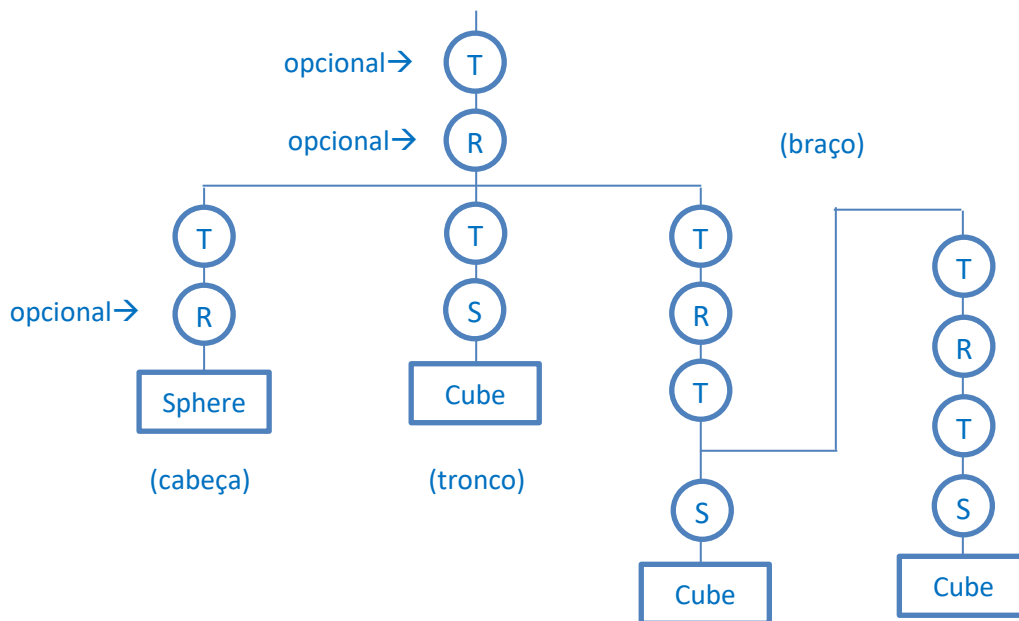
2017-02-08

os braços e as pernas são articulados apenas na direcção do eixo dos Y (linhas tracejadas na imagem).

Imagine que quer colocar uma câmara por trás da cabeça do modelo (ver figura) a olhar em frente. Indique os parâmetros a utilizar em `gluLookAt()`.

```
gluLookAt (  
    modelo.corpo.x - DIST_CAM * cos(modelo.corpo.dir +  
    modelo.cabeça.dir),  
    modelo.corpo.y - DIST_CAM * sin(modelo.corpo.dir +  
    modelo.cabeça.dir),  
    modelo.corpo.z + ALTURA_CAM,  
  
    modelo.corpo.x,  
    modelo.corpo.y,  
    modelo.corpo.z + ALTURA_CAM,  
  
    0,  
    0,  
    1);
```

- e. **[3.0]** Pretende-se implementar a árvore de desenho do modelo anterior, para o corpo, cabeça e **apenas um dos braços**. Use as funções `glutSolidCube(1.0)` para desenhar cada um dos paralelepípedos e `glutSolidSphere(modelo.corpo.raio_cabeca, 10, 10)` para desenhar a cabeça. Ambas as instruções desenhavam objectos centrados na origem actual. Construa a árvore de cena recorrendo a um ou mais dos blocos a seguir indicados:



## Sistemas Gráficos e Interacção

Época de Recurso

2017-02-08

**Nota:** mude para uma nova folha de respostas

- f. **[2.0]** Considere um cubo com material laranja (1.0, 0.5, 0.25) iluminado por uma única fonte de luz verde claro (0.5, 1.0, 0.0). Qual as componentes primárias da cor resultante?

$$R = 0.5 \quad G = 0.5 \quad B = 0.0$$

- g. **[2.0]** Imagine dois objectos: uma parede branca mate e uma esfera de plástico brilhante verde. Indique as instruções do OpenGL e os parâmetros respectivos que usaria para configurar as características diferentes entre eles.

```
GLfloat[] mat_black = {0.0f, 0.0f, 0.0f, 1.0f};  
GLfloat[] mat_white = {1.0f, 1.0f, 1.0f, 1.0f};  
GLfloat[] mat_green = {0.0f, 1.0f, 0.0f, 1.0f};
```

```
/* branco mate */  
  
glMaterialfv(GL_FRONT, GL_DIFUSE, mat_white);  
glMaterialfv(GL_FRONT, GL_SPECULAR, mat_black);  
glMaterialf(GL_FRONT, GL_SHININESS, 0.0);
```

```
/* verde brilhante */  
  
glMaterialfv(GL_FRONT, GL_DIFUSE, mat_green);  
glMaterialfv(GL_FRONT, GL_SPECULAR, mat_green);  
glMaterialf(GL_FRONT, GL_SHININESS, 100.0);
```

- h. **[2.0]** Suponha que pretende animar um objecto ao longo do eixo dos X. Indique qual o **callback GLUT** que deve utilizar para conseguir realizar a animação e escreva o código correspondente.

*callback:*

```
glutTimerFunc()
```

*código:*

```
void Timer(int value) {  
    modelo.x++;  
    glutPostRedisplay();  
    glutTimerFunc(delay, Timer, 1);  
}
```