Licenciatura em Engenharia Informática
Sistemas Gráficos e Interacção

Época Especial 10-09-2008

Parte Teórica **50% – 60 minutos**

N.º_____ Nome _____

1. Na representação de um vector 2D em coordenadas homogéneas são usadas

   a) Apenas duas componentes: *x* e *y*
   b) Três componentes: *x*, *y* e *w*, em que *w* = 0
   c) Quatro componentes: *x*, *y*, *z* e *w*, em que *w* = 1
   d) Nenhuma das anteriores

2. Se a um determinado objecto 3D aplicarmos uma transformação de escalamento com coeficientes (-1, 1, 1) obteremos

   a) Um objecto reflectido no plano OXY
   b) Um objecto rodado em torno do eixo dos X
   c) Um erro, pois nas transformações de escalamento não são permitidos coeficientes negativos
   d) Nenhuma das anteriores

3. Uma forma de determinar o vector normal a um triângulo genérico consiste em

   a) Calcular o produto vectorial dos vectores definidos pelos vértices do triângulo e dividir o vector resultante pelo seu comprimento
   b) Calcular o produto escalar dos vectores definidos pelos vértices do triângulo e multiplicar o valor assim obtido pelas coordenadas do ponto médio do triângulo
   c) Calcular o produto externo dos vectores definidos pelos vértices do triângulo e dividir o valor assim obtido pela área do triângulo
   d) Dividir o triângulo genérico em dois ou mais triângulos isósceles

4. As transformações projectivas servem, no modelo da câmara virtual do *OpenGL*, para

   a) Mover e/ou deformar os objectos abrangidos pelo volume de visualização da câmara
   b) Mover e/ou orientar a câmara
   c) Ajustar a lente/objectiva da câmara
   d) Ampliar/reduzir a fotografia tirada pela câmara

5. A codificação *Winged-Edge* de sólidos

   a) É usada na representação de sólidos por fronteira (B-Rep)
   b) Armazena informação numa estrutura associada às arestas
   c) Permite determinar em tempo constante os 9 tipos de adjacência de vértices, arestas e faces
   d) Todas as anteriores

6. No modelo de iluminação do *OpenGL*

   a) As fontes de luz omnidireccionais não têm componente especular
   b) As fontes de luz posicionais encontram-se localizadas a uma distância infinita da cena
   c) As fontes de luz direccionais são atenuadas
   d) Nenhuma das anteriores

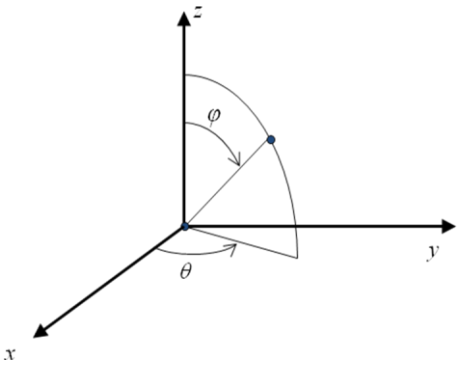
7. Uma função de mapeamento de textura

   a) Devolve, para cada ponto do espaço de textura, a cor do objecto ao qual a textura está a ser aplicada
   b) Devolve, para cada ponto do espaço de textura, o ponto correspondente do objecto ao qual a textura está a ser aplicada
   c) Não pode basear-se na descrição paramétrica da superfície do objecto ao qual a textura está a ser aplicada
   d) Nenhuma das anteriores

8. A função de mapeamento que a seguir se discrimina baseia-se numa parametrização

$$x(\varphi,\theta) = \sin\varphi\cos\theta$$
$$y(\varphi,\theta) = \sin\varphi\sin\theta$$
$$z(\varphi,\theta) = \cos\varphi$$

$$\varphi = \pi \cdot t$$
$$\theta = 2\pi \cdot s$$



   a) Cúbica
   b) Cilíndrica
   c) Esférica
   d) Nenhuma das anteriores

9. Num sistema interactivo, a divisão de tarefas grandes e/ou complexas em subtarefas mais pequenas e simples

   a) Não deve ser contemplada, pois diminui significativamente a performance do utilizador na realização da tarefa
   b) Não deve ser feita de forma hierárquica, pois aumenta significativamente o esforço de memorização imposto ao utilizador
   c) Só se aplica às interfaces de linha de comandos
   d) Nenhuma das anteriores

Parte Teorico-Prática

N.º_____ Nome _____

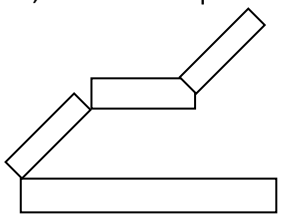
10. Para simular o efeito de iluminação de um poste público ao longo da noite numa cena de exterior (por exemplo, banco de jardim), devemos usar uma luz móvel ou fixa? Posicional ou direccional?

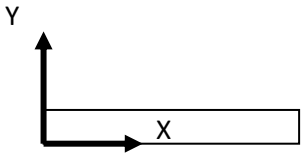
    Luz fixa posicional

11. Podemos desenhar várias primitivas sem ser necessário definir, para cada uma delas, qual a cor a utilizar. Qual a característica do *OpenGL* que permite isso?

    O *OpenGL* guarda um conjunto de variáveis de estado como, por exemplo, a cor actual/ corrente e a espessura das linhas.

12. Supondo que possui um método **desenhaParalelepipedo**(*L*, *A*, *P*) que desenha um paralelepípedo com largura *L*, altura *A* e profundidade *P*, com base na origem do sistema de eixos, escreva a sequência de instruções necessárias para desenhar o seguinte objecto:



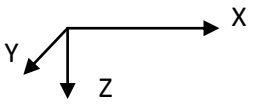



```
glPushMatrix();
        desenhaParalelepipedo(L1, A1, P1);
        glTranslatef(0, A1, 0);
        glRotatef(45, 0, 0, 1);
        desenhaParalelepipedo(L2, A2, P2);
        glTranslatef(L2, 0, 0);
        glRotatef(-45, 0, 0, 1);
        desenhaParalelepipedo(L3, A3, P3);
        glTranslatef(L3, A3/2.0, 0);
        glRotatef(45, 0, 0, 1);
        desenhaParalelepipedo(L4, A4, P4);
glPopMatrix();
```

13. O volume de projecção (por exemplo, glOrtho) define:

    a) Quais os objectos que ficarão visíveis
    b) O cubo ou pirâmide dentro do qual serão desenhados os objectos
    c) Tem por "origem" a posição da câmara e "cresce" no sentido oposto ao da câmara
    d) O espaço 3D cujos objectos no seu interior serão eventualmente desenhados

14. Caso queira que o seu sistema de eixos fique orientado como indicado na figura, deve usar:



    a) gluLookAt(0, 0, -1, 0, 0, 0, 0, -1, 0);
    b) gluLookAt(0, 0, 1, 0, 0, 0, 0, -1, 0);
    c) gluLookAt(0, 1, 0, 0, 0, 0, 0, 0, -1);
    d) gluLookAt(0, 1, 0, 0, 0, 0, 0, 0, 1);

15. Qual dos seguintes vectores define uma luz direccional perpendicular à linha de visão da câmara na sua configuração por omissão?

    a) (1, 1, 1, 0)
    b) (1, 0, 0, 1)
    c) (0, 1, 0, 0)
    d) (0, 0, 1, 1)

16. Um objecto que se mova no espaço bidimensional XY pode ser caracterizado pela posição (*x*, *y*) e direcção (ângulo em radianos). Se pretendermos que a câmara se encontre sempre atrás do objecto devemos definir a posição do objecto como o centro da câmara e utilizar para *eye* qual dos seguintes valores?

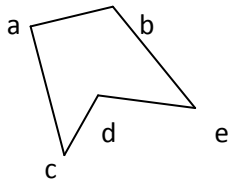
    a) eye.x = obj.x - r * cos(obj.dir);
       eye.y = obj.y - r * sin(obj.dir);

    b) eye.x = obj.x + r * cos(obj.dir);
       eye.y = obj.y - r * sin(obj.dir);

    c) eye.x = obj.x + r * cos(obj.dir);
       eye.y = obj.y + r * sin(obj.dir);

    d) eye.x = obj.x - r * cos(obj.dir);
       eye.y = obj.y + r * sin(obj.dir);

17. Indique qual a sequência de código que permitiria desenhar o perímetro do seguinte polígono:



a)
```
glBegin(GL_LINES);
    glVertex3fv(a);
    glVertex3fv(b);
    glVertex3fv(c);
    glVertex3fv(d);
    glVertex3fv(e);
glEnd();
```

b)
```
glBegin(GL_POLYGON);
    glVertex3fv(a);
    glVertex3fv(c);
    glVertex3fv(e);
    glVertex3fv(d);
    glVertex3fv(c);
glEnd();
```

c)
```
glBegin(GL_TRIANGLE_FAN);
    glVertex3fv(a);
    glVertex3fv(b);
    glVertex3fv(e);
    glVertex3fv(d);
    glVertex3fv(c);
glEnd();
```

d)
```
glBegin(GL_LINE_LOOP);
    glVertex3fv(a);
    glVertex3fv(b);
    glVertex3fv(e);
    glVertex3fv(d);
    glVertex3fv(c);
glEnd();
```

18. Qual a normal (não necessariamente unitária) do polígono definido pelos seguintes vértices: (2, 1, 0), (-2, 1, 0), (-2, 1, 1) e (2, 1, 1)?

a) (1, 0, 0)
b) (1, 1, 1)
c) (1, 1, 0)
d) (0, 1, 0)