

N.º _____ Nome _____

Duração da prova: 45 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

10%

- a. **[3.3]** Num sistema gráfico dotado de um *frame buffer* RGBA de 1024 x 1024 x 32 bits
- i. É possível a reprodução de imagens com $2^{24} \approx 16$ milhões de cores
 - ii. Cada píxel é descrito por 11 bits para a componente vermelha, 11 bits para a verde e 10 bits para a azul, num total de 32 bits
 - iii. É possível a reprodução de imagens com 1024 níveis de transparência
 - iv. Nenhuma das anteriores
- b. **[3.3]** Uma projecção perspectiva constitui um exemplo de
- i. Uma transformação linear afim
 - ii. Uma transformação identidade
 - iii. Uma transformação rígida
 - iv. Nenhuma das anteriores
- c. **[3.3]** Qual das seguintes transformações usaria para transformar o objecto representado à esquerda na Figura 1 no objecto da direita?
- i. Rotação
 - ii. Escalamento
 - iii. *Shearing*
 - iv. Nenhuma das anteriores

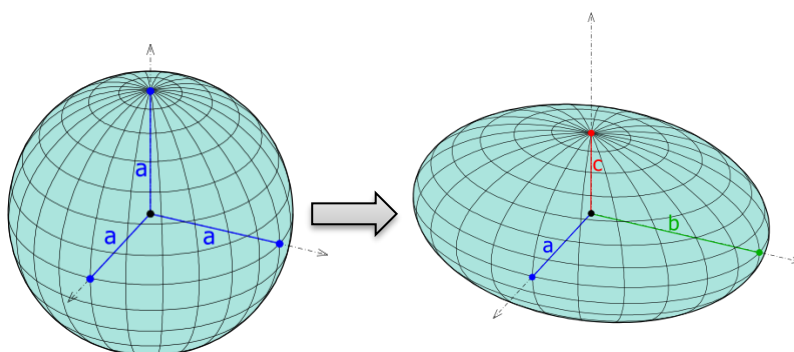


Figura 1

- d. **[3.3]** Considere o objecto delimitado pela superfície descrita pela seguinte equação:

$$(x - 1)^2 + (y - 2)^2 + (z - 3)^2 - 1 = 0$$

O ponto de coordenadas (1.0, 2.0, 3.0) encontra-se

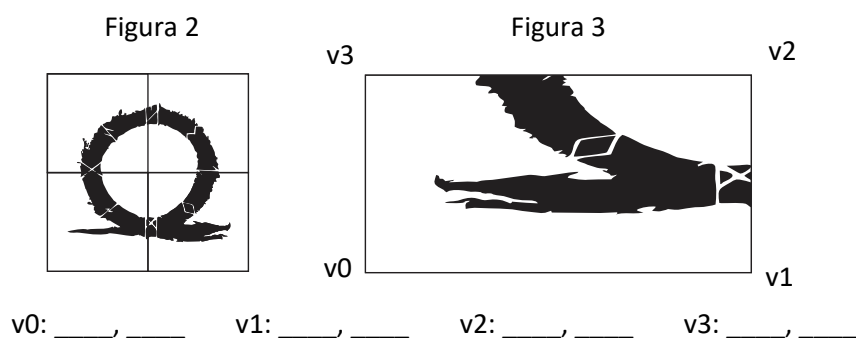
- i. No interior do objecto
 - ii. Na fronteira do objecto
 - iii. No exterior do objecto
 - iv. Nenhuma das anteriores
- e. **[3.3]** Quais os valores dos factores de atenuação que permitem simular uma situação em que a intensidade da luz reflectida por um objecto se reduz para metade quando a distância entre a fonte de luz e o objecto iluminado aumenta para o dobro?
- i. Constante = 1.0; linear = 0.0; quadrático = 0.0
 - ii. Constante = 0.0; linear = 1.0; quadrático = 0.0
 - iii. Constante = 0.0; linear = 0.0; quadrático = 1.0
 - iv. Nenhuma das anteriores
- f. **[3.3]** A técnica de *mipmapping* de mapeamento de texturas
- i. Não é suportada pela maioria das API gráficas
 - ii. É aplicável aos contextos de magnificação
 - iii. É incompatível com as parametrizações esféricas
 - iv. Nenhuma das anteriores

N.º _____ Nome _____

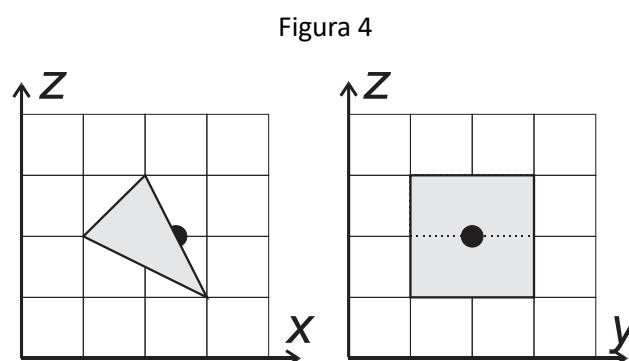
Parte Teórico-Prática

20%

- a. **[4.0]** Pretende-se mapear a textura representada na Figura 2 num rectângulo, de modo que este fique com o aspecto ilustrado na Figura 3. Indique as coordenadas de textura correspondentes a cada um dos vértices do polígono.



- b. **[2.0]** No objecto 3D da Figura 4, qual a normal no ponto assinalado pela semiesfera preta?



Normal (pode ser não unitária): _____, _____, _____

- c. **[1.4]** A diferença entre um *Mesh* e um *Object3D* é que o *Mesh* tem
- i. Materiais e Geometria
 - ii. Possibilidade de ter subobjectos
 - iii. Posição no espaço 3D
 - iv. Suporte de Sombras
- d. **[1.4]** Numa *PerspectiveCamera*, o factor de *zoom* pode ser ajustado com
- i. Ângulo
 - ii. Largura
 - iii. *Aspect Ratio*
 - iv. *zFar*

- e. **[1.4]** Uma vantagem de usar índices para definir uma geometria é ter-se menos
- Vértices
 - Arestas
 - Materiais
 - Faces
- f. **[1.4]** Ao usar sombras é muito simples definir
- Que objectos específicos originam sombras
 - Que objectos específicos recebem sombras
 - Que luzes vão ter sombras
 - Todas as anteriores
- g. **[1.4]** Ao usar uma *SpotLight*, o cálculo das sombras é efectuado usando
- Uma câmara *Orthographic*
 - Uma câmara *Perspective*
 - Seis câmaras *Orthographic*
 - Seis câmaras *Perspective*
- h. **[1.4]** Para rodar o objecto `obj` em torno do eixo dos Z, no ângulo `ang`, pode-se usar
- `obj.rotate("Z", ang);`
 - `threeJS.rotate(obj, 0.0, 0.0, ang);`
 - `obj.rotateZ(ang);`
 - `axisZ.rotate(obj, ang);`
- i. **[1.4]** Para obter uma renderização mais perfeita de texturas deve-se usar
- LinearFilter*
 - NearestFilter*
 - LambertFilter*
 - RadiosityFilter*
- j. **[1.4]** Ao usar *picking* com *RayCaster* obtém-se como resultado:
- O objecto mais próximo
 - Um *array* de objectos em que o mais próximo é o primeiro elemento do *array*
 - Um *array* de objectos em que o mais próximo é o último elemento do *array*
 - Um *array* de objectos não ordenados, mas com informação de profundidade
- k. **[1.4]** A propriedade *fog* dos materiais permite indicar
- O tipo de nevoeiro a aplicar ao objecto (*Linear*, *Exponential*)
 - A ordem pela qual o efeito de nevoeiro é aplicado
 - Se o objecto vai ser afectado pelo nevoeiro
 - A cor do nevoeiro a aplicar no objecto
- l. **[1.4]** Para implementar reflexões melhoradas em tempo real podemos usar
- CubeCamera*
 - ReflectEngine*
 - Ammo.js*
 - DynamicReflection*