EA978 – Lista 12 – Imagens Digitais e Texturização

Data de Entrega: 02/06/2009

1. (adaptada da questão 2.4 de Gonzalez) A unidade de medida comumente utilizada em transmissão de dados digitais é a "taxa *baud*", definida como o número de *bits* transmitidos por segundo. Considerando que a transmissão seja desempenhada em pacotes consistindo de um *bit* de início, um *byte* (8 bits de informação), um *bit* de paridade e um *bit* de parada, responda:

   (a) quantos minutos levaria para transmitir uma imagem de 512×512 *pixels* com 256 níveis de cinza à taxa de 600 *baud*?

   (b) qual seria o tempo à taxa de 9600 *baud*?

   (c) repita (a) e (b) para uma imagem de 1240×1240 com 256 níveis?

2. (adaptada da questão 2.5 de Gonzalez) Dada uma imagem binária

0	1	1	1	1	0	0	1	1	0
1	0	0	1	0	0	1	0	0	1
1	0	0	1	0	1	1	0	0	0
0	0	1	1	1	0	0	0	0	0
0	0	1	1	1	0	0	1	1	1

   (a) Considere que as coordenadas do *pixel* no canto superior esquerdo sejam $(0, 0)$. Qual é a vizinhaça N_4 e N_8 do *pixel* $(3, 2)$?

   (b) Quantos componentes conexos com valor 1 há na imagem, considerando conectados-de-4; conectados-de-8; e conectados-de-m.

   (c) Quantos componentes segmentados no item anterior são adjacentes.

3. (adaptada da questão 2.10 de Gonzalez) Considere uma sub-imagem mostrada abaixo

3	1	2	1 (q)
2	2	3	0
1	2	1	1
1	0	0	2
(p)1	1	1	2

   (a) Compute a distância D_4, D_8 entre p e q, supondo conectividade em *pixels* de valores 0 e 1.

   (b) Compute a distância D_4, D_8 entre p e q, supondo conectividade em *pixels* de valores 1 e 2.

4. Considere duas imagens em escala de cinza, de 0 a 255, uma gerada pelo procedimento

```
int i, j, c;

for (i = 0; i < LARGURA; i++) {
   for (j = 0; j < ALTURA; j++) {
      c = (((i&0x16)==0)^((j&0x16)==0))*255;
      imagem1[i][j] = (GLubyte) c;
   }
}
```

e outra pelo procedimento

```
for (i = 0; i < LARGURA; i++) {
   for (j = 0; j < ALTURA; j++)
    imagem2[i][j] = 0;
}
for (i = LARGURA/4; i < (3*LARGURA)/4; i++) {
  for (j = ALTURA/4; j < (3*ALTURA)/4; j++)
    imagem2[i][j] = (GLubyte) 255;
}
```

(a) Como são as duas imagens geradas proceduralmente?

(b) Como é o resultado da operação lógica AND, *bit* a *bit*, entre as duas imagens? E da operação aritmética divisão?

(c) Cite uma aplicação para cada operação lógico-aritmética: AND, OR, NOT, soma, subtração, multiplicação e divisão.

(d) Amplie a imagem pelo fator 2 na direção x e na direção y. O que vai acontecer com os *pixels* em "0"?

(e) Rode a imagem por 45^0 em torno do centro da imagem no sentido anti-horário. O que vai acontecer com os *pixels* em "0"?

(f) Aplique a matriz de transformação

$$\begin{bmatrix} 1 & 0.6 & 0 \\ 0.4 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

sobre todas as amostras da imagem. O que vai acontecer com os *pixels* em "0" da imagem?

(g) Em quais imagens geradas no item anterior pode ocorrer sobreposição das amostras? E em quais imagens as amostras podem ficar espaçadas deixando lacunas entre elas? Justifique.

(h) Quais são as técnicas que se podem utilizar para corrigir as "falhas" que aparecem após algumas transformações?

5. Dada uma imagem definida no espaço de textura $[0,1] \times [0,1]$

Como se pode aplicá-la sobre uma superfície paramétrica, como a superfície de Bézier $P(u,v) = \sum_{i=0}^{m}(\sum_{j=0}^{n} P_{ij}B_{n,j}(u))B_{m,i}(v)$?

6. Dada uma textura de reflectância definida no espaço de textura $[0,1] \times [0,1]$

Determine a correspondência entre as coordenadas de textura (s, t) e as coordenadas (x, y, z) de uma esfera. Justifique.

7. Sejam $b(s, t) = \cos(5s)/5$ um mapa de textura de perturbação das normais e $\mathbf{r}(u, v) = u(1, 0, 0, 0) + v(1, 1, 1, 0)$ um plano. Determine a posição do ponto $\mathbf{r}(1, 1)$ e a normal da superfície neste ponto após a texturização, supondo que o ponto $\mathbf{r}(1, 1)$ seja mapeado ao ponto $(2, 1)$ do mapa de textura.

8. OpenGL

   (a) OpenGL provê operações lógico-aritméticas entre um fragmento de valores já existente em *frame buffer* e um outro novo fragmento de valores. Elas são habilitadas através do comando glEnable(GL_COLOR_LOGIC_OP) e define-se o tipo de operação desejada com glLogicOp() antes de pintar o novo fragmento. Verifique a sua resposta nos itens (a) e (b) da questão 4 com uso destas funções.

   (b) OpenGL provê uma grande variedade de funções relacionadas com a textura. Uma delas é a "filtragem" que "procura compatibilizar" a resolução de uma imagem discreta com a resolução da textura aplicada nela. Para habilitar a texturização, deve-se utilizar o comando glEnable(GL_TEXTURE_2D) e para selecionar o tipo de filtro, glTexParameteri(). Utilizando estas funções, podemos "preencher" as lacunas ao "esticarmos" uma textura sobre uma superfície. Podemos, portanto, contornar o problema de "lacunas" que apareceram na questão 4 através da técnica de texturização sobre um plano com as imagens dadas. Verifique isso com uma implementação.

   (c) É possível texturizar uma superfície de Bézier através de OpenGL. Além de habilitar as funções de textura, deve-se habilitar o mapeamento 2D por meio de glEnable(GL_MAP2_TEXTURE_COORD_2) e definir o mapeamento da textura por meio de glMap2d(). Escolha uma imagem e utilize-a para texturizar uma superfície de Bézier bicúbica.