EA978 – Lista 9 – Amostragem e Quantização

Data de Entrega: 07/05/2009

1. Em que diferenciam os seguintes conceitos em um espaço discreto em relação aos definidos para o espaço contínuo?

   (a) distância

   (b) vizinhança

   (c) conectividade

   (d) conexidade

   (e) adjacência

2. Seja uma área de desenho de 64×64 *pixels*, preenchida com a cor preta, e um segmento P_0P_1, onde $P_0 = (10, 50)$ e $P_1 = (45, 4)$. Considere ainda que a cor atribuída a P_0 seja $RGB_0 = (1, 0, 1)$ e a cor a P_1, $RGB_1 = (1, 1, 0)$. Rasterize o segmento com as cores interpoladas linearmente, com uso do

   (a) algoritmo DDA

   (b) algoritmo de ponto médio

   Compare os dois algoritmos, quanto aos resultados, tipo de operações e quantidade de operações envolvidas,

3. Dados 2 triângulos: $((1, 50, 10, 1), (12, 1, 4, 1), (57, 49, 10, 1))$ e $((15, 4, 5, 1), (63, 2, 15, 1), (11, 58, 12, 1))$. O primeiro é azul e o segundo é vermelho. Considere ainda que o observador esteja olhando na direção do semi-eixo z positivo.

   (a) Rasterize os 2 triângulos, com uso do algoritmo de *scan-line* com *z-buffer*. Mostre explicitamente os passos do procedimento.

   (b) O algoritmo de *scan-line* é eficiente porque explora alguns tipos de coerência para reduzir o número de operações. Indique dois tipos de coerência explorados no seu procedimento.

   (c) Para tirar melhor proveito das coerências, é necessário pré-processar os dados, estruturando-os de forma mais apropriada. Como é esta estruturação no algoritmo de *scan-line*?

4. Dados os valores, entre 0 e 255, das amostras de uma imagem em tons de cinza gerada pelo procedimento

```
   int i, j, c;

   for (i = 0; i < 64; i++) {
      for (j = 0; j < 64; j++) {
c = 155 * sin((i*M_PI)/10) * cos ((j*M_PI)/5) + 150;
        if (c < 0) c = 0;
        else if (c > 255) c = 255;
        imagem[i][j] = (GLubyte) c;
      }
   }
```

- Represente o histograma desta imagem. É uma imagem clara ou escura? Justifique.
- Determine os 16 níveis de quantização, utilizando
  - quantização uniforme
  - algoritmo de populosidade
  - algoritmo de corte mediano
- Particione os valores de tons de cinza em 16 células de quantização, com base nos 16 níveis de quantização obtidos com o algoritmo de populosidade.
- Com somente dois valores, 0 e 255, quantize esta imagem com aparência de 16 níveis de cinza com uso da
  - técnica de *dither* de Bayer
  - técnica de difusão de erro Floyd-Steinberg

5. OpenGL: Através do comando glDrawPixels consegue-se carregar um arranjo retangular de *pixels* armazenado em uma memória de processador para o *framebuffer* e exibí-lo.

   - Implemente o algoritmo de ponto médio e rasterize o segmento da questão 2 e os seguintes segmentos, em branco, num reticulado de 64×64:

     (a) $P_0 = (32, 4)$ e $P_1 = (32, 60)$
     (b) $P_0 = (4, 32)$ e $P_1 = (60, 32)$
     (c) $P_0 = (4, 4)$ e $P_1 = (60, 60)$
     (d) $P_0 = (2, 2)$ e $P_1 = (55, 30)$

     Visualize o resultado na tela de exibição.
   - Implemente o algoritmo de *scanline* com *z-buffer* para rasterizar os triângulos da questão 3. Visualize o resultado na tela de exibição.
   - Visualize a imagem em níveis de cinza da questão 4 na tela de exibição.
   - Dois comandos glHistogram e glGetHistogram permitem você obter os valores dos *pixels* de uma imagem, quando devidamente habilitadas. Há ainda os comandos glMinmax e glGetMinmax que fornecem os valores máximo e mínimo atribuídos aos *pixels* de uma imagem. Utilize estes comandos para implementar o algoritmo de populosidade. Em seguida, implemente a técnica de Bayer e técnica de Floyd-Steinber para visualizar a imagem quantizada em 2 níveis de cinza da questão 4. Compare visualmente as duas imagens quantizadas e calcule o erro médio quadrático da quantização.