UNIVERSIDADE FEDERAL DA PARAÍBA
CENTRO DE INFORMÁTICA

Disciplina: [**GDSCO0051] Introdução à Computação Gráfica - Turma 01**.
Semestre: **2019.2**.
Professor: **Christian Azambuja Pagot (email: christian@ci.ufpb.br)**.
Data: **16/12/2019**.
**Nome: ___________________________________ Matrícula: ______________**

**Prova I**

# Questão 1 – 2.5 pts)

O livro do Foley *et al*. apresenta uma versão do algoritmo de Bresenham, vista em aula, que funciona para linhas com coeficiente angular $m$ tal que $0 \leq m \leq 1$. Abaixo, segue o algoritmo:

```
void MidPointLine(int x0, int y0,      // ponto inicial da linha
                  int x1, int y1,      // ponto final da linha
                  int R, int G, int B) // cor da linha {
    int dx = x1 - x0;
    int dy = y1 - y0;
    int d = 2 * dy - dx;
    int incrE = 2 * dy;
    int incrNE = 2 * (dy - dx);
    int x = x0;
    int y = y0;
    PutPixel(x, y, R, G, B); // primeiro pixel da linha
    while (x<x1) {
        if (d<=0) {
            d += incrE;
            x++;
        } else {
            d += incrNE;
            x++;
            y++;
        }
        PutPixel(x, y, R, G, B);
    }
}
```

Altere o algoritmo **`MidPointLine()`** apresentado acima de forma que ele detecte se a linha a ser rasterizada é horizontal e utilize, neste caso, um algoritmo mais eficiente que o MidPoint para rasterizá-la.

Resposta:

```
void MidPointLine(int x0, int y0,      // ponto inicial da linha
                  int x1, int y1,      // ponto final da linha
                  int R, int G, int B) // cor da linha {
    int dy = y1 - y0;

>>> if (dy == 0) {
>>>     for (int x=x0; x<=x1; ++x)
>>>         PutPixel(x, y0, R, G, B);
>>> } else {
        int dx = x1 - x0;
        int d = 2 * dy - dx;
        int incrE = 2 * dy;
        int incrNE = 2 * (dy - dx);
        int x = x0;
        int y = y0;
        PutPixel(x, y, R, G, B); // primeiro pixel da linha
        while (x<x1) {
            if (d<=0) {
                d += incrE;
                x++;
            } else {
                d += incrNE;
                x++;
                y++;
            }
            PutPixel(x, y, R, G, B);
        }
>>> }
}
```

## Questão 2 – 2.5 pts)

Transformações geométricas são utilizadas quando se deseja alterar a forma ou a posição de um determinado objeto. Estas transformações são geralmente descritas na forma de matrizes, o que permite a combinação de uma sequência de transformações em uma única matriz. A partir da matriz **M=ABC**, onde

$$\mathbf{A}=\begin{bmatrix}1&0&0&2\\0&1&0&2\\0&0&1&2\\0&0&0&1\end{bmatrix}\qquad \mathbf{B}=\begin{bmatrix}3&0&0&0\\0&3&0&0\\0&0&3&0\\0&0&0&1\end{bmatrix}\qquad \mathbf{C}=\begin{bmatrix}1&0&0&5\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{bmatrix},$$

construa a matriz $\mathbf{M}^{-1}$ como um produto de matrizes.

Resposta

$\mathbf{M}^{-1}= \mathbf{C}^{-1}\mathbf{B}^{-1}\mathbf{A}^{-1}$

onde

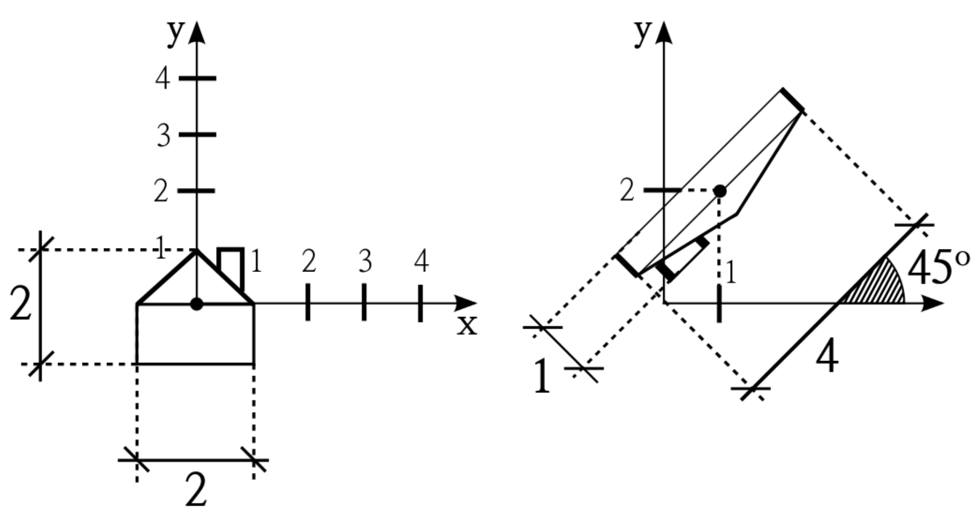
$$\mathbf{A}^{-1}=\begin{bmatrix}1&0&0&-2\\0&1&0&-2\\0&0&1&-2\\0&0&0&1\end{bmatrix}\qquad \mathbf{B}^{-1}=\begin{bmatrix}1/3&0&0&0\\0&1/3&0&0\\0&0&1/3&0\\0&0&0&1\end{bmatrix}\qquad \mathbf{C}^{-1}=\begin{bmatrix}1&0&0&-5\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{bmatrix},$$

Abaixo a solução numérica em Octave que **não foi solicitado na prova**, mas que coloquei aqui apenas para voces verem como implementar em Octave.

```
>> a = [1 0 0 02; 0 1 0 2; 0 0 1 2; 0 0 0 1]
a =

   1   0   0   2
   0   1   0   2
   0   0   1   2
   0   0   0   1

>> b = [3 0 0 0; 0 3 0 0; 0 0 3 0; 0 0 0 1]
b =

   3   0   0   0
   0   3   0   0
   0   0   3   0
   0   0   0   1

>> c = [1 0 0 5; 0 1 0 0; 0 0 1 0; 0 0 0 1]
c =

   1   0   0   5
   0   1   0   0
   0   0   1   0
   0   0   0   1

>> inv(a * b * c)
ans =

   0.33333  -0.00000  -0.00000  -5.66667
   0.00000   0.33333  -0.00000  -0.66667
   0.00000   0.00000   0.33333  -0.66667
   0.00000   0.00000   0.00000   1.00000
```

## Questão 3 – 2.5 pts)

Suponha que cada transformação geométrica 2D vista em aula (rotação, escala e translação) seja considerada uma transformação geométrica fundamental, ou seja, não pode ser representada pela combinação das outras transformações. Com base no desenho abaixo, descreva a matriz **M**, responsável por transformar a figura da esquerda na figura da direita, como o produto de n matrizes, tal que $\mathbf{M} = \mathbf{M}_1 \mathbf{M}_2 \ldots \mathbf{M}_n$, onde $\mathbf{M}_i$ (para $1 \le i \le n$), é uma transformação fundamental.



Resposta

**M** = **M**$_1$ **M**$_2$ **M**$_3$

onde

$$\mathbf{M_1}=\begin{bmatrix}1 & 0 & 1\\ 0 & 1 & 2\\ 0 & 0 & 1\end{bmatrix} \qquad \mathbf{M_2}=\begin{bmatrix}\cos(\theta) & -\sin(\theta) & 0\\ \sin(\theta) & \cos(\theta) & 0\\ 0 & 0 & 1\end{bmatrix} \qquad \mathbf{M_3}=\begin{bmatrix}2 & 0 & 0\\ 0 & 1/2 & 0\\ 0 & 0 & 1\end{bmatrix},$$

onde theta = 135o ou -225o.

## Questão 4 – 2.5 pts)

Com base na matrix de projeção abaixo

$$\mathbf{M_{projection}}=\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & d\\ 0 & 0 & -\frac{1}{d} & 0\end{bmatrix}$$

onde $d$=1, e com base no ponto P=(4, 6, -2, 1), definido no espaço da câmera, responda:

1. Quais as coordenadas do ponto P após sua transformação para o espaço de recorte?
2. Quais as coordenadas do ponto P após sua transformação para o espaço canônico?

Resposta

$$\mathbf{M_{projection}}=\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 1\\ 0 & 0 & -1 & 0\end{bmatrix}$$

1) P=(4, 6, -1, 2)
2) P=(2, 3, -1/2, 1)