Universidade Federal
de Ouro Preto

Trabalho de Implementação

Aluno em Graduação da Universidade Federal de Ouro Preto do curso Ciência da Computação:

Halliday Gauss Costa dos Santos.

Matrícula: 18.1.4093.

Área: Processamento de Imagens.

Questão 1)

Através dos seguintes comandos:

```
>> img = imread("C:\Users\halli\Desktop\7Periodo\PDI\Listas\Lista6\lenna.png");
>> imshow(img);
```

É mostrado na tela a imagem que será utilizada para testes de detecção de borda:

Utilizando a máscara de Sobel sobre a imagem da Lenna através dos seguintes comandos:

```
>> sobel = edge(img, "sobel");
>> imshow(sobel);
```

Obtêm-se o seguinte resultado:

Utilizando a máscara de Prewitt sobre a imagem da Lenna através dos seguintes comandos:

```
>> prewitt = edge(img, "prewitt");
>> imshow(prewitt);
```

Obtêm-se o seguinte resultado:

Utilizando a máscara de Roberts sobre a imagem da Lenna através dos seguintes comandos:

```
>> roberts = edge(img, "roberts");
>> imshow(roberts);
```

Obtêm-se o seguinte resultado:

Utilizando a máscara de Canny sobre a imagem da Lenna através dos seguintes comandos:

```
>> canny = edge(img, "canny");
>> imshow(canny);
```

Obtêm-se o seguinte resultado:

Utilizando a máscara Laplaciano sobre a imagem da Lenna através dos seguintes comandos:

```
>> laplaciano = edge(img, "log");
>> imshow(laplaciano);
```

Obtêm-se o seguinte resultado:

Utilizando a máscara Zero Crossing sobre a imagem da Lenna através dos seguintes comandos:

```
>> zerocrossing = edge(img, "zerocross");
>> imshow(zerocrossing);
```

Obtêm-se o seguinte resultado:

Conclui-se que para a imagem da Lenna usada para teste a máscara de Canny obteve o melhor resultado.

Questão 2)

Função threshold global implementada:

thresholdingGlobal.m

```
function T = thresholdingGlobal(img, precisao)

    [lin, col] = size(img);  % pega as dimensoes da imagem

    T = img(randi(lin), randi(col));  % estimativa inicial de T aleatória

    T_old = 0;

    while(T-T_old) > precisao
        G1 = img(img > T);

        G2 = img(img <= T);

        mu1 = mean(G1);

        mu2 = mean(G2);
        T_old = T;
        T = (mu1 + mu2)/2;

    end
end
```

Questão 3)

A partir da seguinte implementação:

Editor - C:\Users\halli\Desktop\7Periodo\PDI\Listas\Lista6\segmentacao.m*

thresholdingGlobal.m | segmentacao.m*

```
function nimg = segmentacao()

    t = imread ("C:\Users\halli\Desktop\7Periodo\PDI\Listas\Lista6\coins.png");
    t = rgb2gray(t) ; %transforma para escala de cinza

    [row , col] = size (t);
    [x , y]= meshgrid ( 1 : row , 1 : col ) ;

    t2 = double(t) .* ((x+y)/2+64)+x+y ;
    t3 = uint8 (255 * mat2gray (t2)) ;

    figure; imshow(t3);

    T = thresholdingGlobal(t3, 10);

    t3(t3> T) = 255;
    t3(t3 <= T) = 0;

    nimg = t3;


    figure;imshow(nimg);

end
```

Gerará as seguintes imagens:

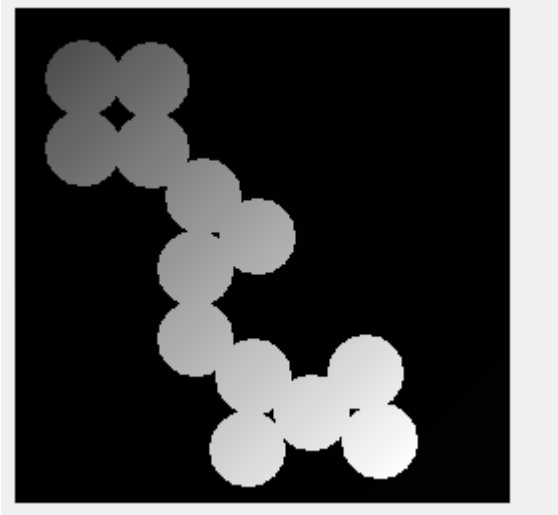

Imagem em escala de cinza com degradê.

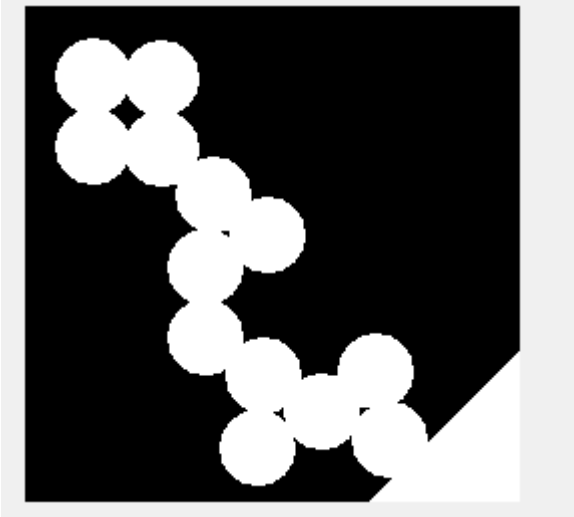

Imagem binarizada.

É possível perceber que a terceira partição não se adequou e uma parte do fundo da imagem ficou branco. Esse resultado foi atingindo utilizando a função thresholdingGlobal com uma precisão em 10.

Questão 4)

Através dos seguintes comandos:

```
>> img = imread("C:\Users\halli\Desktop\7Periodo\PDI\Listas\Lista6\lenna.png");
>> imshow(img);
```

É mostrado na tela a imagem que será utilizada para testes de binarização:

E através dos seguintes comandos:

```
>> T = graythresh(img);
>> imgB = im2bw(img, T);
>> imshow(imgB, []);
```

Utiliza-se o método de Otsu para determinar o limiar que irá separar o preto do branco na imagem original destacando o ponto de interesse, e a imagem binarizada resultante será: