Dados do relatório Lista 2

Matéria: Processamento Digital de Imagens (PDI) **código:** CMP1084 A01

Aluno: Vitor de Almeida Silva

Matrícula: 20161003305497

ENUNCIADO

1. Converta a sua imagem em uma imagem cinza e normalize a imagem para valores entre [0 e 1]. Posteriormente binarize a imagem com limite k = 0,68 e qualquer número desejado e, mostre as imagens resultantes.

2. Execute a correção gama para a imagem inicial em escala de cinza e normalize a imagem para valores entre [0 e 1]. A equação de transformação é $s=r*c\hat{}y$, onde r é a intensidade de entrada, s é a intensidade de saída, c é uma constante e geralmente é definida como 1, y é o parâmetro que você pode alterar. Defina y = 0.2, y = 0.8 e y = 2.5, exiba e comente os diferentes resultados.

3. Refaça a questão 2, mas utilize sua imagem colorida.

4. Crie seu algoritmo para calcular o histograma de uma imagem e utilize-a na imagem resultante da questão 2 e 3 para y=2.5.

5. Torne a imagem de entrada mais clara ou mais escura.

6. Escolha um limiar e aplique em todos os componentes da sua imagem.

7. Quantifique os planos de cores usando 2 bits e 4 bits visualize o efeito das operações.

A imagem 1 foi a utilizada no relatório.

Imagem 1: imagem utilizado ao longo da lista

Fonte: https://aninerd.com.br/noticia/2019/12/04/go-toubun-no-hanayome-chega-ao-fim-no-14o-volume/

DESENVOLVIMENTO

1. **Converta a sua imagem em uma imagem cinza e normalize a imagem para valores entre [0 e 1]. Posteriormente binarize a imagem com limite k = 0,68 e qualquer número desejado e, mostre as imagens resultantes.**

Resposta:

Para converter a imagem em escala de cinza, pode-se extrair os componentes R, G e B da imagem e utilizar a formula padrão NTSC para converter os pixels em intensidade de cinza. Para tanto, basta realizar um somatório dos pixels R, G e B para uma nova imagem utilizando os pesos padrões (TEAM, 2010).

Após esse processo de escala de cinza, para converter a imagem em valores de 0 e 1, é preciso realizar a normalização em [0,1] e posteriormente estabelecer um limiar (foi escolhido k=0.68). Deste modo, se realiza o procedimento de binarização. A Figura 1 mostra os resultados dos procedimentos.

Figura 1: imagem orinial, imagem escala de cinza, imagem binarizada

Fonte: Autoral

Código questão 1:

```
%leitura da imagem
img=imread('img1.jpg');
imshow(img(:,:,:));
title('imagem original');
figure();

img2=img;
[x, y, k]= size(img2);

%****primeiro passo, converter a imagem em escala de cinza*****
imgGray=imgEscalaCinza(img2);

imshow(imgGray);
title('imagem em escala de cinza');
figure();
imwrite(imgGray,'imgGray.jpg');


%***** normalizando a imagem para valores 0 e 1
img01=normalizacao(imgGray,255);

%***** binarizando a imagem com k=68
img012=binarizacao(img01,0.68);

imshow(img012);
title('imagem Binarizada');
figure();

function [imgGray] = imgEscalaCinza(img2)

[x, y, ~]= size(img2);

%serparando componentes R, G e B
R = img2(:,:,1);
G = img2(:,:,2);
B = img2(:,:,3);

%criando um nova matriz preenchida com 0s de mesma dimensão da imagem original
imgGray=zeros(x, y, 'uint8');

%atribui cada pixel de intensidade a eu novo local da img em cinza
for i=1:x
    for j=1:y
        imgGray(i,j)= 0.2989 * R(i,j) + 0.5870 * G(i,j) + 0.1140 * B(i,j);
    end
end
end

function [ imgNova ] = normalizacao(img2,k)

[xo,yo,ko] = size(img2);
imgNova=zeros(xo,yo,ko,'double');

for i=1 :xo
    for j=1 : yo
        imgNova(i,j,:) = double(img2(i,j,:))/double(k);
    end
end
end

function [ img01 ] = binarizacao(img2,T)

[xo,yo,ko] = size(img2);
img01 = uint8(zeros(xo,yo,ko));

% T limiar de binarização

for i=1:xo
    for j=1:yo
        if(img2(i,j,:) < T)
             img01(i,j,:) = 255;
        end
    end
end
end
```

2. **Execute a correção gama para a imagem inicial em escala de cinza e normalize a imagem para valores entre [0 e 1]. A equação de transformação é $s=r*c\hat{}y$, onde r é a intensidade de entrada, s é a intensidade de saída, c é uma constante e geralmente é definida como 1, y é o parâmetro que você pode alterar. Defina y = 0.2, y = 0.8 e y = 2.5, exiba e comente os diferentes resultados. (pág 51 e 71 gonzales pdf)**

Resposta:

A correção gama, permite que seja alterado valores de contraste e brilho da imagem. Estes procedimentos possibilitam a melhora da qualidade da imagem em alguns casos, além de permitir também, que seja alterado o brilho da imagem (GONZALEZ, WOODS, 2010).

Através da alteração da variável de valor gama, é possível aumentar e reduzir a intensidade de luz na imagem. Deste modo, para valores de y > 1, o brilho da imagem é reduzido e, para valores de y<1 e y>=0, o brilho da imagem é intensificado. Também é possível controlar o contraste da imagem, porém, para este caso, seria necessário a adição de outro termo na formula dada no exercício, de modo que ela ficaria como a seguinte.

F1. G= c.f + b (MARQUES; VIEIRA, 1999)

No qual f é a intensidade de entrada, c controla o contraste da imagem e b o brilho da imagem. No livro do Gonzales, é mostrado o modelo de formula indicado no exercício, que segue a ideia da F2.

F2. s=cr^y

Nesta, o gama fica como expoente de r. c é normalmente 1 (GONZALEZ, WOODS, 2010).

Para normalização, foi utilizada as formulas indicadas no livro texto. A normalização consiste em se dividir o maior valor do intervalo final, pela diferença dos valores mínimos de uma matriz selecionada. As formulas são as seguintes.

F3. Fm = F – min(F);

F4. Fs = k[Fm/max(Fm)]; (GONZALEZ, WOODS, 2010).

Seguindo a formula indicada no exercício, os resultados das imagens são mostrados nas Figuras 2, 3, 4. Note que conforme o valor de y aumenta, o brilho

da imagem é reduzido. Da mesma forma, para se ter uma visualização correta da imagem, foi utilizado um o intervalo de [0 255].

Figura 2: imagem com correção gama 0.2 e normalizada

Fonte: Autoral

Como dito anteriormente, o efeito da correção gama provoca uma intensificação ou redução do brilho da imagem. Com um y=0.2 o resultado foi o mostrado na Figura 2. A Figura 3 mostra o resultado para um y=0.8.

Figura 3: imagem com correção gama 0.8 e normalizada

Fonte: Autoral

A Figura 4 indica o resultado para y=2.5.

Figura 4:imagem com correção gama 2.5 e normalizada

Fonte Autoral

Se as Figura 2, 3 e 4 forem comparadas, é possível notar, uma alteração em termos de intensidade de brilho nas imagens. Essa alteração de brilho ocorre de acordo com a alteração do parâmetro y, que justamente, aplica o fator de correção gama, que influencia na qualidade e no brilho da imagem.

Código questão 2:

```
%leitura da imagem
img=imread('img1.jpg');
imshow(img(:,:,:));
title('imagem original');
figure();

[x, y, k]= size(img);

%transformando a imagem em escala de cinza
img2 = imgEscalaCinza(img);

imshow(img2);
title('imagem em escala de cinza');
figure();
imwrite(img2,'imgGray.jpg');

%***** normalizando a imagem para valores 0 e 1 (binarizando) T=200
img02=normalizacao(img2,255);
img08=normalizacao(img2,255);
img25=normalizacao(img2,255);

%correção gama
% T e c parâmetro de correção, quanto mais alto mais brilho na imagem
img02= correcaoGama (img02,0.2,1);
img08= correcaoGama (img08,0.8,1);
img25= correcaoGama (img25,2.5,1);


imshow(img02(:,:,:));
title('imagem com correção gama y = 0.2');
figure();

imshow(img08(:,:,:));
title('imagem com correção gama y = 0.8');
figure();

imshow(img25(:,:,:));
title('imagem com correção gama y = 2.5');
figure();

imwrite(img25,'imgQuest4_1.jpg');

function [imgGama] = correcaoGama (img2,T,c)

[x,y,k] = size(img2);
imgGama= zeros(x,y,k,'double');
for i=1:x
    for j=1: y
        imgGama(i,j,:) = double( c * (img2(i,j,:).^T) );
    end
end
end

function [imgGray] = imgEscalaCinza(img2)

[x, y, ~]= size(img2);
%serparando componentes R, G e B
R = img2(:,:,1);
G = img2(:,:,2);
B = img2(:,:,3);

%criando um nova matriz preenchida com 0s de mesma dimensão da imagem original
imgGray=zeros(x, y, 'uint8');

%atribui cada pixel de intensidade a eu novo local da img em cinza
for i=1:x
    for j=1:y
        imgGray(i,j)= 0.2989 * R(i,j) + 0.5870 * G(i,j) + 0.1140 * B(i,j);
    end
end
end

function [ imgNova ] = normalizacao(img2,k)

[xo,yo,ko] = size(img2);
imgNova=zeros(xo,yo,ko,'double');

for i=1 :xo
    for j=1 : yo
        imgNova(i,j,:) = double(img2(i,j,:))/double(k);
    end
end
end
```

3. Refaça a questão 2, mas utilize sua imagem colorida.

Resposta:

Foi utilizado o código desenvolvido na questão 2, sobra a imagem agora em RGB. As Figuras 5, 6 e 7, mostram os efeitos para y=0.2, y=0.8 e y=2.5 respectivamente.

Figura 5: imagem colorida com correção gama 0.2 e normalizada

Fonte: Autoral

Figura 6: imagem colorida com correção gama 0.8 e normalizada

Fonte: Autoral

Figura 7: imagem colorida com correção gama 2.5 e normalizada

Fonte: Autoral

Como dito anteriormente, a correção gama muda a qualidade da imagem e também a intensidade luminosa. Este efeito também pode ser notado utilizando-se a imagem em RGB.

Código questão 3:

```
%leitura da imagem
img=imread('img1.jpg');
imshow(img(:,:,:));
title('imagem original');
figure();

[x, y, k]= size(img);

img2=img;

%***** normalizando a imagem para valores 0 e 1 (binarizando) T=200
img02=normalizacao(img2,255);
img08=normalizacao(img2,255);
img25=normalizacao(img2,255);

%correção gama
% T e c parâmetro de correção, quanto mais alto mais brilho na imagem
img02= correcaoGama (img02,0.2,1);
img08= correcaoGama (img08,0.8,1);
img25= correcaoGama (img25,2.5,1);

imshow(img02(:,:,:));
title('imagem com correção gama y = 0.2');
figure();

imshow(img08(:,:,:));
title('imagem com correção gama y = 0.8');
figure();

imshow(img25(:,:,:));
title('imagem com correção gama y = 2.5');
figure();

imwrite(img25,'imgQuest4_2.jpg');


function [imgGama] = correcaoGama (img2,T,c)

[x,y,k] = size(img2);
imgGama= zeros(x,y,k,'double');
for i=1:x
    for j=1: y
        imgGama(i,j,:) = double( c * (img2(i,j,:).^T) );
    end
end
end



function [ imgNova ] = normalizacao(img2,k)

[xo,yo,ko] = size(img2);
imgNova=zeros(xo,yo,ko,'double');

for i=1 :xo
    for j=1 : yo
        imgNova(i,j,:) = double(img2(i,j,:))/double(k);
    end
end
end
```

4. **Crie seu algoritmo para calcular o histograma de uma imagem e utilize-a na imagem resultante da questão 2 e 3 para y=2.5.**

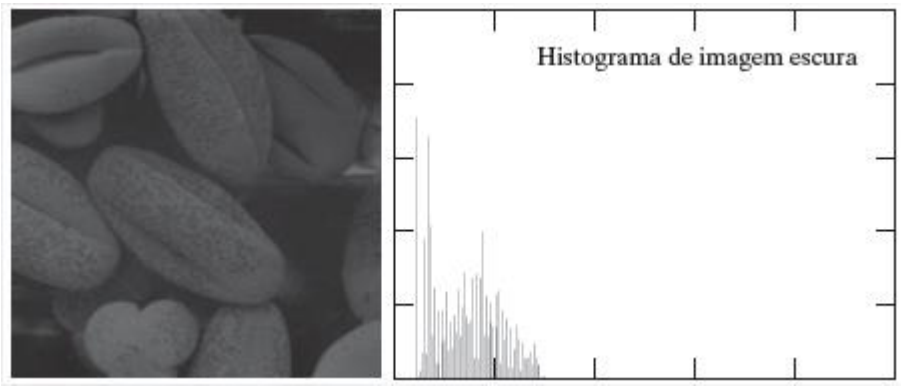
Resposta:

O histograma é uma função discreta **h(Rk) = Nk**, onde **Rk** é o k-ésimo valor de intensidade e **Nk** é o número de pixels da imagem com intensidade Rk. Geralmente, um histograma é normalizado dividindo cada componente do histograma pelo número total de pixels da imagem, dado por (M*N). (GONZALEZ, WOODS, 2010).

Deste modo, um histograma normalizado é dado por **P(Rk) = Rk/(M*N)** para **k=0, 1 ,2 ..., L-1** (L é o número máximo do intervalo de intensidade contido na imagem). A soma de todos os componentes de um histograma é igual a 1 e o valor de **P(Rk)** é a probabilidade da ocorrência de uma intensidade **Rk** em uma imagem (GONZALEZ, WOODS, 2010).

Além de fornecer estatísticas úteis, a manipulação de histogramas pode ser usada para o realce de imagens. A Figura 8 mostra um exemplo de histograma, no qual o eixo vertical corresponde a valores de intensidade (Rk). O eixo horizontal corresponde a valores de h(Rk)=Nk ou P(Rk) = Nk/(M*N) se é um histograma normalizado (GONZALEZ, WOODS, 2010).

Figura 8: Exemplo de histograma



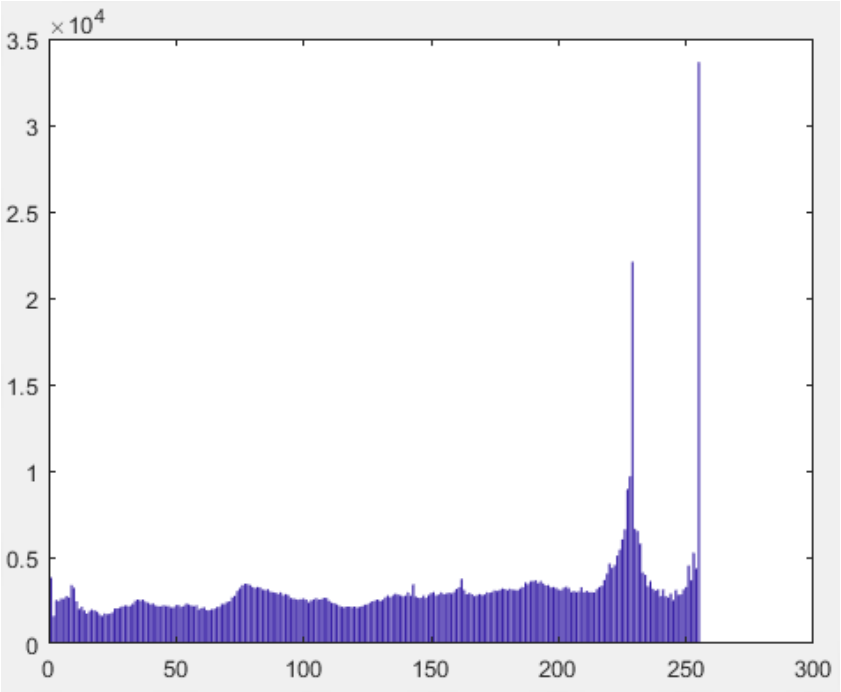
Fonte: (GONZALEZ, WOODS, 2010)

Então, os histogramas podem ser vistos como gráficos que seguem os seguintes pontos:

- **X: (h(Rk) = Nk) ou X :(p(Rk) = Nk / (M*N));**
- **Y: Rk.**

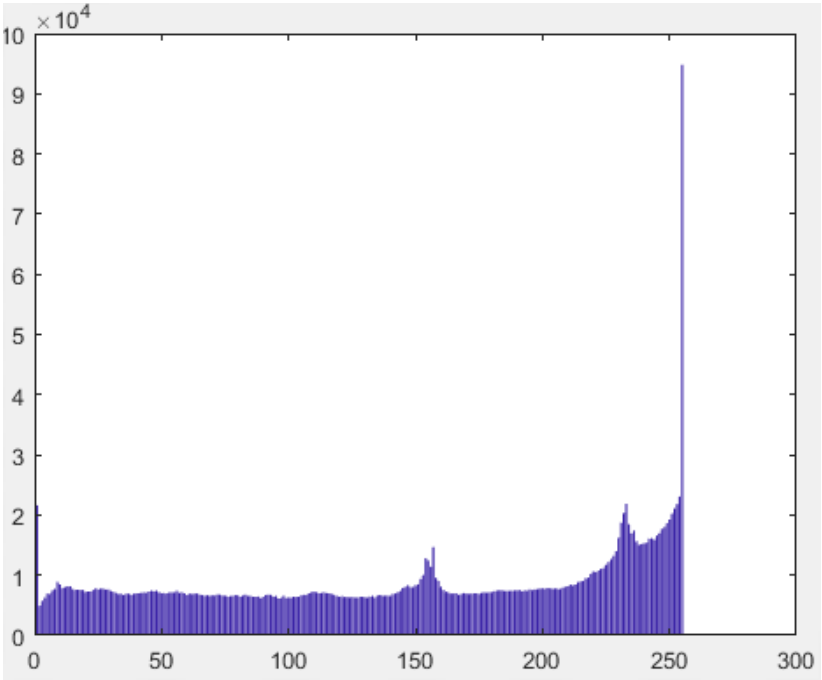
Deste modo, as Figura 9 e 10 mostram os resultados dos histogramas obtidos das imagens das questões 2 e 3 para um y=2.5.

Figura 9: Histograma imagem questão 2 y=2.5



Fonte: Autoral

Figura 10: Histograma imagem questão 3 y=2.5



Fonte: Autoral

Código questão 4:

```
%leitura da imagem %coloque os endereços das imagens geradas nas quests 2 e 3
img0=imread('imgQuest4_1.jpg');
img1=imread('imgQuest4_2.jpg');

imshow(img0(:,:,:));
title('imagem_2  y=2.5');
figure();

imshow(img1(:,:,:));
title('imagem_3  y=2.5');
figure();

Hist = histograma(img0);
bar(Hist);
title('histograma imagem questão 2 y=2.5');
figure();

Hist = histograma(img1);
bar(Hist);
title('histograma imagem questão 3 y=2.5');


function [histo] = histograma (img)

   [xo,yo,~]= size(img);
   histo(1:256)=0;

   for i=1:xo
       for j=1:yo
           k=img(i,j,:);
           histo(k+1)=  histo(k+1) + 1;
       end
   end

end
```

5. Torne a imagem de entrada mais clara ou mais escura.

Resposta:

Para tornar uma imagem mais escura ou mais clara, basta somar uma constante para a intensidade de cada pixel da imagem (GONZALEZ, WOODS, 2010). Desta forma, foi escolhida a constante C=50, a qual foi utilizada para clarear e escurecer a imagem definida na lista. A Figura 11 e 12 mostram os resultados obtidos para as operações

Figura 11: Imagem clareada utilizando C=50

Fonte: Autoral

Figura 12: Imagem escurecida utilizando C=-50

Fonte: Autoral

Código questão 5:

```
%leitura da imagem
img=imread('img1.jpg');

imshow(img(:,:,:));
title('imagem original');
figure();

img2=img;
[x, y, k]= size(img2);

%imagem mais clara
imgNova= intensidadeBrilho(img2,50);

imshow(imgNova(:,:,:));
title('imagem mais clara');
figure();

%imagem mais escura
imgNova= intensidadeBrilho(img2,-50);

imshow(imgNova(:,:,:));
title('imagem mais escura');
figure();

function [imgNova] = intensidadeBrilho (img,c)
[x, y, k]= size(img);
imgNova= img;

for i=1 : x
    for j=1 : y
        imgNova(i,j,:) = imgNova(i,j,:) + c;
    end
end
end
```

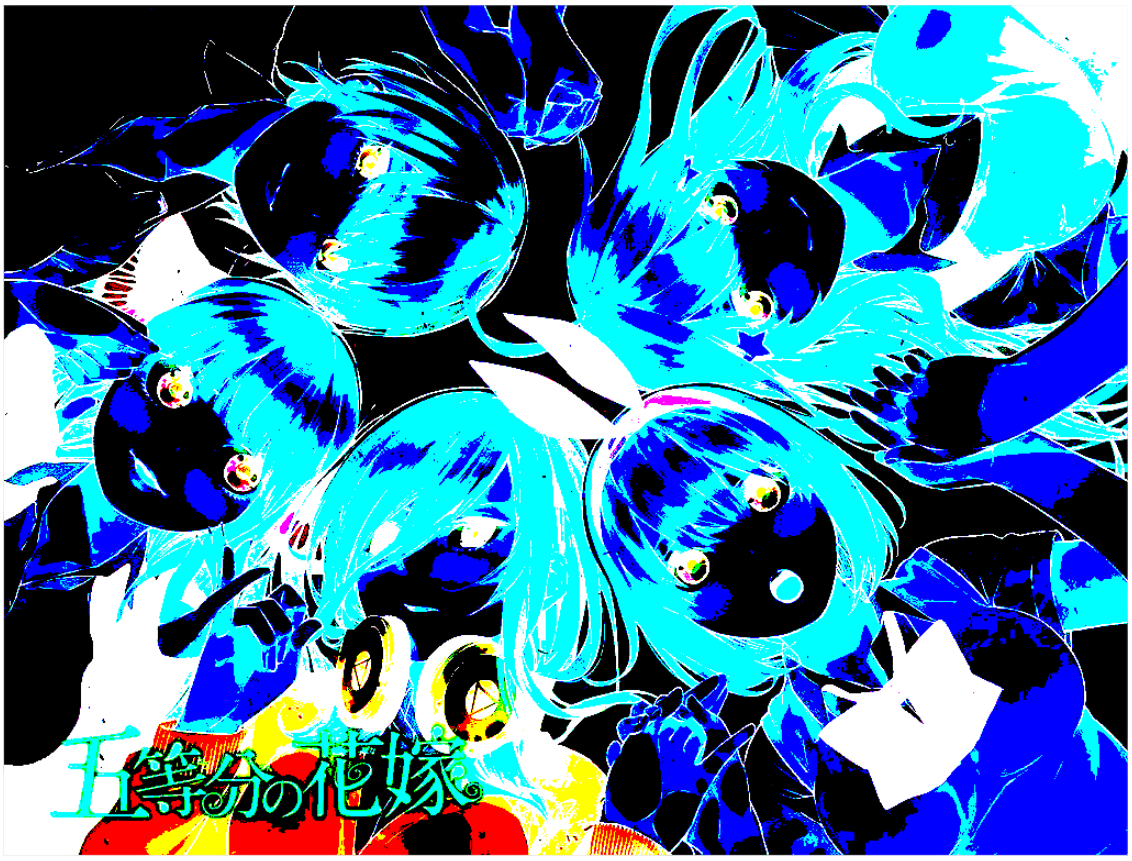
6. Escolha um limiar e aplique em todos os componentes da sua imagem.

Resposta:

Para um processo de binarização da imagem, normalmente é escolhido um limiar para os valores de intensidade de cada pixel de uma imagem. Tal limiar, funciona como um divisor que pode ser utilizado para transformar pixels acima ou abaixo em valores 0 ou 255, por exemplo. Também podem ser definidos outros cálculos como, por exemplo, aplicar uma correção gama em somente uma parte da imagem acima ou abaixo do limiar.

Para o presente exercício, foi selecionado um Limiar k=200, este, foi aplicado na imagem normal. O Resultado é mostrado na Figura 13. Lembrando que desta vez os processos foram realizados com a imagem colorida em RGB.

Figura 13: imagem com limiar k=200



Fonte: Autoral

Também foi realizado o processo para um limiar k=90, como mostrado no exemplo do exercício na lista. A Figura 14 mostra o resultado para um limiar k=90.

Figura 14: imagem com limiar k=90

Fonte: Autoral

Analisando-se a imagem de exemplo exposta no enunciado da lista. Foi identificado que a mesma é resultante da seguinte operação.

Comando. Img(:,:,2)=0;

No qual o elemento G recebe 0 na imagem. Este foi o limiar estabelecido para este caso. Na resposta apresentada, no entanto, foi estabelecido um limiar que, para cada elemento R, G e B, seja aplicado 0 ou 255 em cada bit., o efeito pode ser visto nas figuras supracitadas.

Código questão 6:

```
%leitura da imagem
img=imread('img1.jpg');
imshow(img(:,:,:));
title('imagem original');
figure();

img2=img;
[x, y, k]= size(img2);

%limiar selecionado
T=200;
img01= binarizacaoRGB(img2,T);

imshow(img01(:,:,:));
title('imagem binarizada com limiar k= ' + string(T));
figure();

%esta função binariza cada elemento R, G e B de forma individual
function [ img01 ] = binarizacaoRGB(img2,T)

[xo,yo,ko] = size(img2);
img01 = uint8(zeros(xo,yo,ko));

% T limiar de binarização
for i=1:xo
    for j=1:yo
        if(img2(i,j,1) < T)
             img01(i,j,1) = 255;
        else
            img01(i,j,1) = 0;
        end
        if(img2(i,j,2) < T)
             img01(i,j,2) = 255;
        else
            img01(i,j,2) = 0;
        end
        if(img2(i,j,3) < T)
             img01(i,j,3) = 255;
        else
             img01(i,j,3) =0;
        end
    end
end

end
```

7. Quantifique os planos de cores usando 2 bits e 4 bits visualize o efeito das operações.

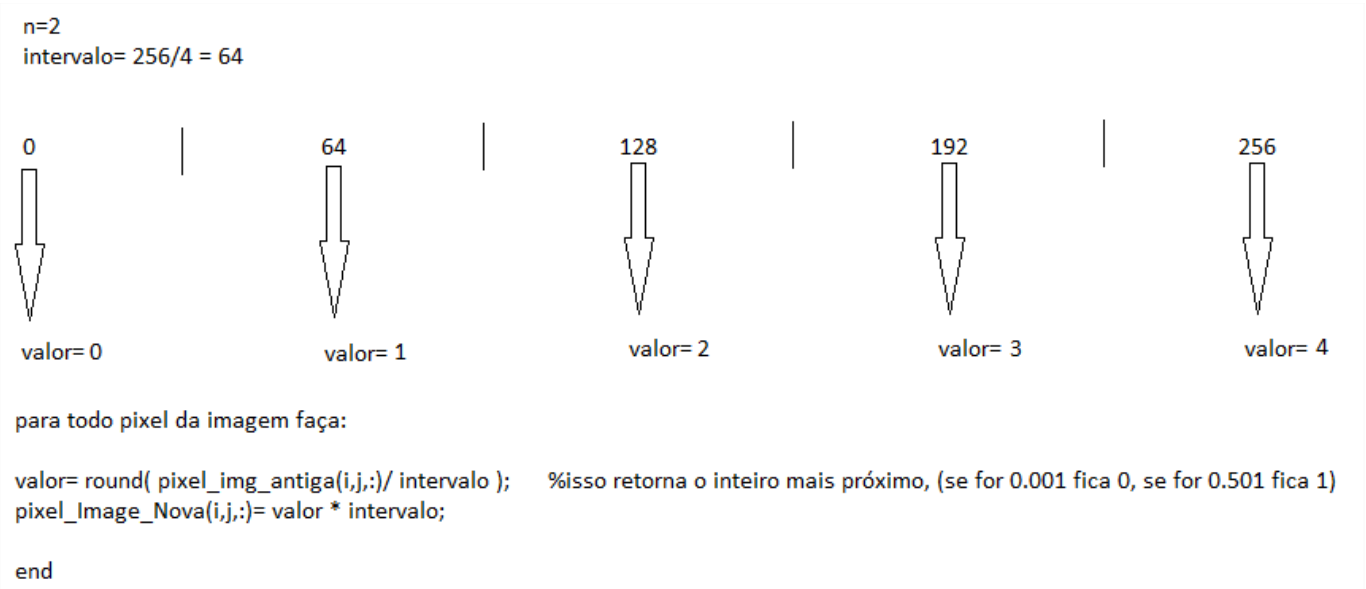
Resposta:

Até o presente momento, estivemos lidando com imagens que aproximam as cores da realidade com 2^24 possibilidades de cores diferentes em um domínio RGB (MATHWORKS, 2019). O exercício, por sua vez, solicitou que se limite essa representatividade para 2 bits e 4 bits, ou seja, 2^2=4 cores e 2^4 = 16 cores.

Deste modo, a ideia utilizada, foi a de uma amostragem, de modo que uma imagem com 2^24 cores pudesse ser discretizada para um intervalo de 2^n. Esse processo, foi definido através do cálculo de um intervalo para cada pixel em R, G e B, de modo que, por meio da função round() se pegue o inteiro mais próximo retornado pela divisão e se multiplique pelo intervalo, que acaba funcionando como o passo da amostragem.

Para tanto, foi feito uma divisão do valor de intensidade de cor 256 por 2^n (para n=2 e n=4), de modo que seja possível, identificar os intervalos das cores. A nível de informação, a Figura 15 indica o processo realizado para determinar os valores dos pixels. As Figuras 16 e 17, mostram os resultados para a imagem de 2 bits e 4 bits respectivamente.

Figura 15: processo de atribuição de valores aos pixels para n=2



Fonte: Autoral

Figura 16: imagem com 2 bits de cores

Fonte: Autoral

Figura: 17: imagem com 4 bits de cores

Fonte: Autoral

Analisando as Figuras 16 e 17, é possível notar que a Figura 13 representou melhor a imagem original utilizando 16 cores. De acordo com essa informação, é possível concluir que, quanto mais possibilidades de cores melhor será a representação da imagem original.

Isso também tem a ver com a quantidade em termos de amostragens de cores, afinal de contas, o que foi feito para transformar as imagens, foi uma discretização dos valores de 2^24 para um domínio 2^2 e 2^4 (ou seja, uma amostragem).

Código questão 7:

```
%leitura da imagem
img=imread('img1.jpg');
imshow(img(:,:,:));
title('imagem original');
figure();

img2=img;
[x, y, k]= size(img2);

img02= limitaCores(img,2);
img04= limitaCores(img,4);

imshow(img02(:,:,:));
title('imagem 2 bits');
figure();

imshow(img04(:,:,:));
title('imagem 4 bits');
figure();


function [imgNova] = limitaCores(img,n)

[x,y,k]=size(img);
imgNova=zeros(x,y,k,'uint8');

intervalo= 256/2^n;

for i=1 : x
    for j=1 : y
        %o intervalo de cores é definido através da divisão e multiplicação
        %do valor pelo intervalo definido
        imgNova(i,j,1)=round(img(i,j,1)/intervalo)*intervalo;
        imgNova(i,j,2)=round(img(i,j,2)/intervalo)*intervalo;
        imgNova(i,j,3)=round(img(i,j,3)/intervalo)*intervalo;

    end
end

end
```

REFERÊNCIA

GONZALEZ, Rafael C., WOODS, Richard E. **Processamento Digtal de Imagem.** São Paulo: Sv, 2010.

MARQUES, Ogê; VIEIRA, Hugo. **Processamento Digital de Imagens**. 1. ed. Rio de Janeiro: Brasport, 1999. 331 p.

MATHWORKS. **Reduce the Number of Colors in an Image.** 2019. Disponível em: https://www.mathworks.com/help/images/reduce-the-number-of-colors-in-an-image.html. Acesso em: 19 fev. 2020.

TEAM, Mathworks Support. **How do I convert my RGB image to grayscale without using the Image Processing Toolbox?** 2010. Disponível em: https://www.mathworks.com/matlabcentral/answers/99136-how-do-i-convert-my-rgb-image-to-grayscale-without-using-the-image-processing-toolbox. Acesso em: 15 mar. 2020.