**Computação Gráfica**

* **Trabalho 2:** Exercícios de Manipulação de Imagens no Octave

Bruna Costa Cons

201110341611

**Exercício 1**

% A função ‘imread’ armazena cada imagem em cada variável

low = imread("Einstein\_low\_contrast.png");

med = imread("Einstein\_med\_contrast.png");

high = imread("Einstein\_high\_contrast.png");

% A seguir, abrimos a janela de uma nova figura com o título ‘Einstein Low Contrast’

figure('Name','Einstein Low Contrast','NumberTitle','off')

% O ‘subplot’ dividiu verticalmente em dois pedaços o espaço da figure criada anteriormente

subplot(2,2,[1,2]);

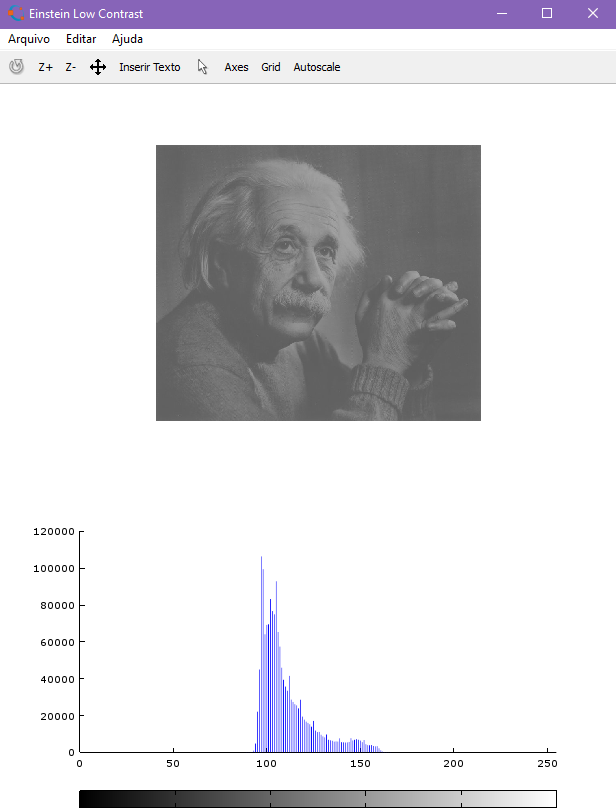
% Após inserir o ’subplot’, o próximo ‘imshow’ irá dispor a imagem ‘low’ no local referenciado, ou seja, na parte superior da figure

imshow(low)

subplot(2,2,[3,4]);

% A seguir, o ‘imhist’ irá dispor o histograma da imagem ‘low’ na parte inferior da figure

imhist(low)



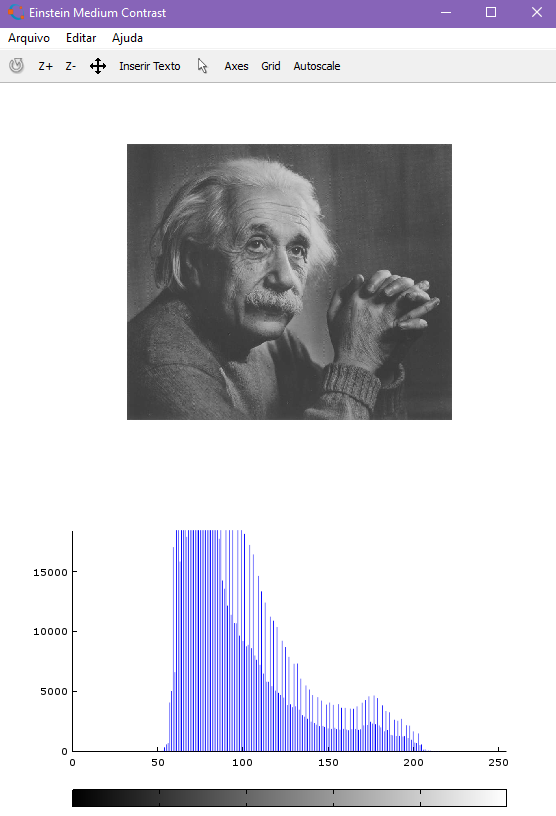
figure('Name','Einstein Medium Contrast','NumberTitle','off')

subplot(2,2,[1,2]);

imshow(med)

subplot(2,2,[3,4]);

imhist(med)



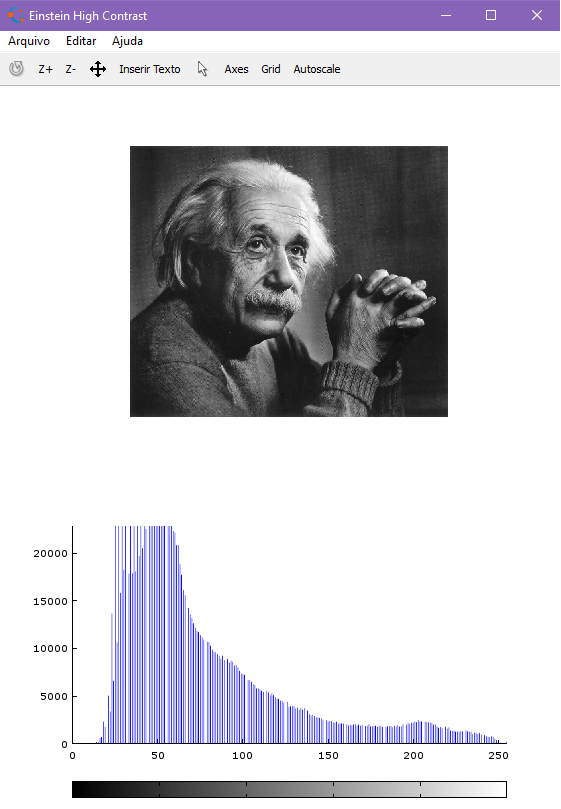
figure('Name','Einstein High Contrast','NumberTitle','off')

subplot(2,2,[1,2]);

imshow(high)

subplot(2,2,[3,4]);

imhist(high)



new\_low = imadjust(double(low)/255,[0.35 0.65],[]);

% A imagem editada chama-se ‘new\_low’

% O ‘imadjust’ aumentou o contraste de ‘low’, buscando se aproximar da imagem ‘high’

% Por questões de incompatibilidade de classes das imagens, foi necessário converter ‘low’ para double

% Uma nova figura é criada para mostrar a imagem editada pelo ‘imadjust’ e o seu histograma

figure('Name','Einstein Low Contrast Edited','NumberTitle','off')

subplot(2,2,[1,2]);

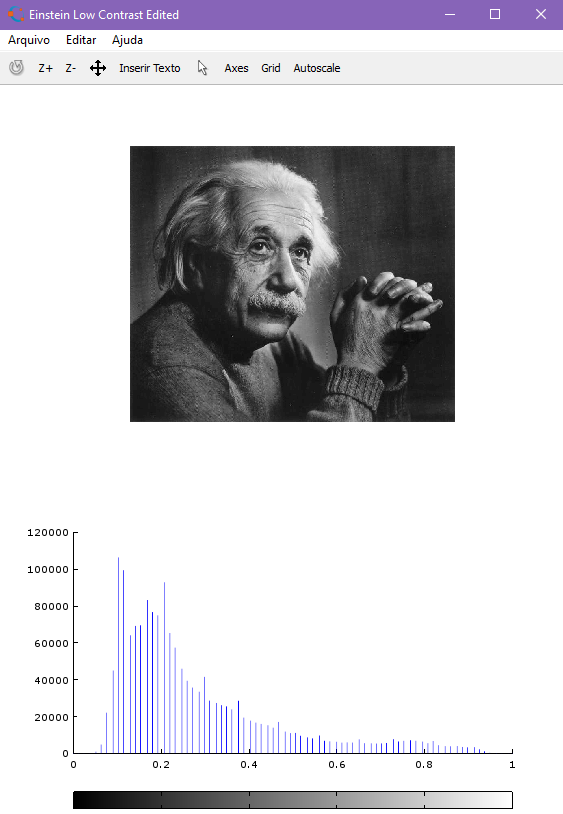
% ‘imshow’ irá mostrar a imagem editada na parte superior da figure

imshow(new\_low)

subplot(2,2,[3,4]);

% ‘imhist’ irá mostrar o histograma da imagem editada na parte inferior da figure

imhist(new\_low)



**Exercício 2**

leme = imread("leme.bmp");

% Por ser RGB, a imagem ‘leme’ pode ser separada por bandas (dimensões das cores: vermelho, verde e azul), sendo armazenadas em suas respectivas variáveis (R, G e B)

R = leme(:, :, 1);

G = leme(:, :, 2);

B = leme(:, :, 3);

% Através do ‘imadjust’ utilizamos a função gama para deixar mais claras as áreas escuras da imagem original, sem que as áreas mais claras mudem muito de intensidade. Isto é feito em cada banda da imagem

L1 = imadjust(double(R)/255,[], [], 0.6);

L2 = imadjust(double(G)/255,[], [], 0.6);

L3 = imadjust(double(B)/255,[], [], 0.6);

% Uma nova imagem é criada: ‘new\_leme’ ao concatenar as bandas ajustadas da imagem ‘leme’

new\_leme = cat(3, L1, L2, L3);

figure('Name','Leme Edited','NumberTitle','off')

subplot(2,2,[1,2]);

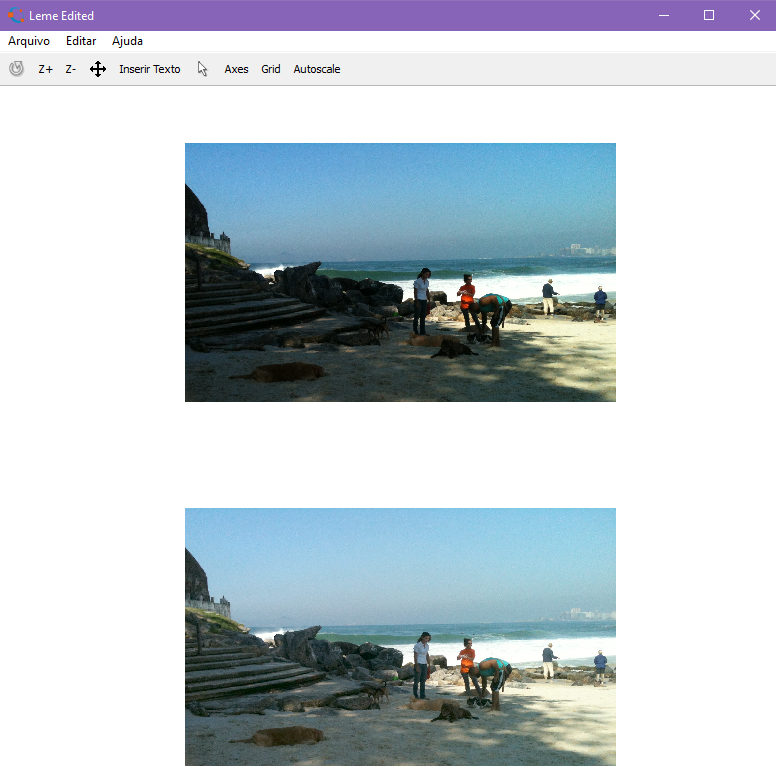
% A seguir, o ‘imshow’ irá mostrar a imagem original (‘leme’) na parte superior da figure

imshow(leme)

subplot(2,2,[3,4]);

% A seguir, o ‘imshow’ irá mostrar a imagem editada (‘new\_leme’) na parte inferior da figure

imshow(new\_leme)



**Exercício 3**

% ScriptEx3

% Primeiro carregamos o pacote de imagem que foi instalado no Octave; isso irá nos permitir usar diversas das funções do script

pkg load image;

resp = 'S';

% O loop de while facilita caso o usuário deseje inserir nova foto após utilizar o filtro de suavização, o que é garantido com a flag ‘resp’, que armazena a resposta do usuário

while resp == 'S' || resp == 's',

arq = input('Insira o nome de arquivo de uma imagem entre aspas: ');

% Uma caixa de texto irá abrir informando erro quando o arquivo que o usuário inseriu não for encontrado no diretório atual, ou seja, o diretório em que o script está, encerrando o programa

if ~exist(arq, 'file')

resp = 'N';

errorMessage = sprintf('Erro: o arquivo %s nao existe no diretorio atual.', arq);

uiwait(warndlg(errorMessage));

return;

endif

% Caso arquivo exista, ele será armazenado em ‘img’

img = imread(arq);

% Um averaging filter foi utilizado como filtro de suavização linear

f = ones(5,5) / 25;

% As bandas da imagem RGB foram separadas e armazenadas em variáveis diferentes

R = img(:, :, 1);

G = img(:, :, 2);

B = img(:, :, 3);

% Após isso, o filtro foi aplicado em cada banda

I1 = imfilter(R, f);

I2 = imfilter(G, f);

I3 = imfilter(B, f);

% Uma nova imagem é criada: ‘new\_img’, ao concatenar as bandas filtradas de ‘img’

new\_img = cat(3, I1, I2, I3);

figure('Name', 'Suavizacao Linear - Filtro de Media', 'NumberTitle', 'off')

subplot(2,2,[1,2]);

imshow(img)

subplot(2,2,[3,4]);

imshow(new\_img)

% A seguir, o valor da flag ‘resp’ é atualizado, pois dependendo da resposta do usuário (se for S ou s), ele poderá inserir uma nova imagem para fazer filtragem

resp = input('Deseja inserir nova foto? (S/N) ', ' ');

endwhile

% Caso o usuário não deseje filtrar novas imagens, o loop do while termina e ‘Tchau!’ é exibido

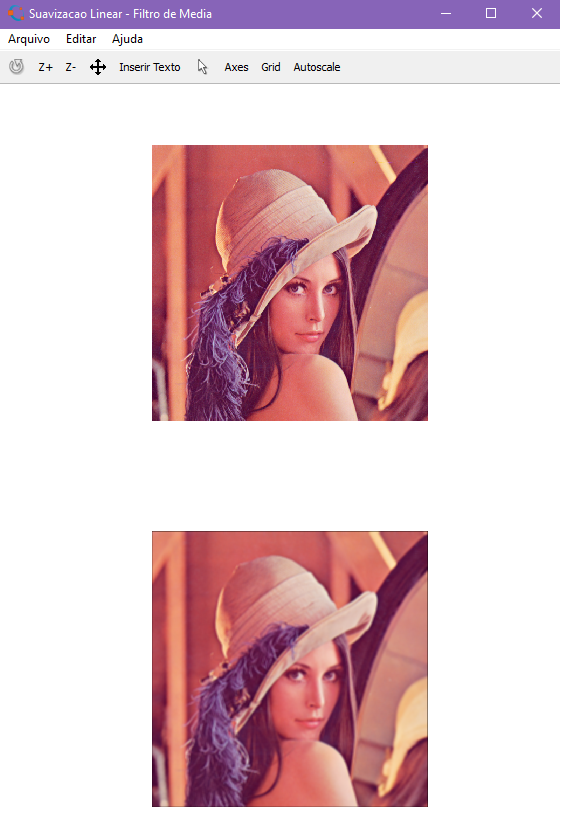
disp('Tchau!');

****

Imagem



Imagem

****

Imagem

**Exercício 4**

% ScriptEx4

pkg load image;

resp = 'S';

while resp == 'S' || resp == 's',

loop = 1;

arq = input('Insira o nome de arquivo de uma imagem entre aspas: ');

if ~exist(arq, 'file')

resp = 'N';

errorMessage = sprintf('Erro: o arquivo %s nao existe no diretorio atual.', arq);

uiwait(warndlg(errorMessage));

return;

endif

img = imread(arq);

% A seguir, mais um loop while é utilizado, para que o usuário possa escolher qual filtro ele deseja utilizar na foto inserida

while loop == 1,

filtro = input('Deseja utilizar qual filtro? [1 - Movimento / 2 - Laplaciano / 3 - Diminuicao de Nitidez] ');

if (filtro == 1)

% Um filtro de movimento é utilizado aqui

f = fspecial('motion', 50, 45);

loop = 0;

else if (filtro == 2)

% Este é o filtro Laplaciano, com alfa = 0,1

f = fspecial('laplacian', 0.1);

loop = 0;

else if (filtro == 3)

% Aqui utilizamos um filtro que deixa a imagem granulada, reduzindo sua nitidez e introduzindo ruídos na mesma

f = fspecial('unsharp', 0.2);

loop = 0;

else

% Nesta opção, o usuário informou um número que não possui filtro correspondente, que é qualquer número além de 1, 2 ou 3

disp('Favor inserir 1, 2 ou 3.');

loop = 1;

endif

endwhile

% A seguir, a imagem RGB inserida pelo usuário, ‘img’, é separada em bandas

R = img(:, :, 1);

G = img(:, :, 2);

B = img(:, :, 3);

% Então, cada banda é editada com o filtro que foi escolhido previamente

I1 = imfilter(R, f);

I2 = imfilter(G, f);

I3 = imfilter(B, f);

% Uma nova imagem é criada: ‘new\_img’, concatenando as bandas filtradas de ‘img’

new\_img = cat(3, I1, I2, I3);

figure('Name','Filtro','NumberTitle','off')

subplot(2,2,[1,2]);

% A imagem original é exibida na parte superior da figure

imshow(img)

subplot(2,2,[3,4]);

% A imagem filtrada é exibida na parte inferior da figure

imshow(new\_img)

resp = input('Deseja inserir nova foto? (S/N)', ' ');

endwhile

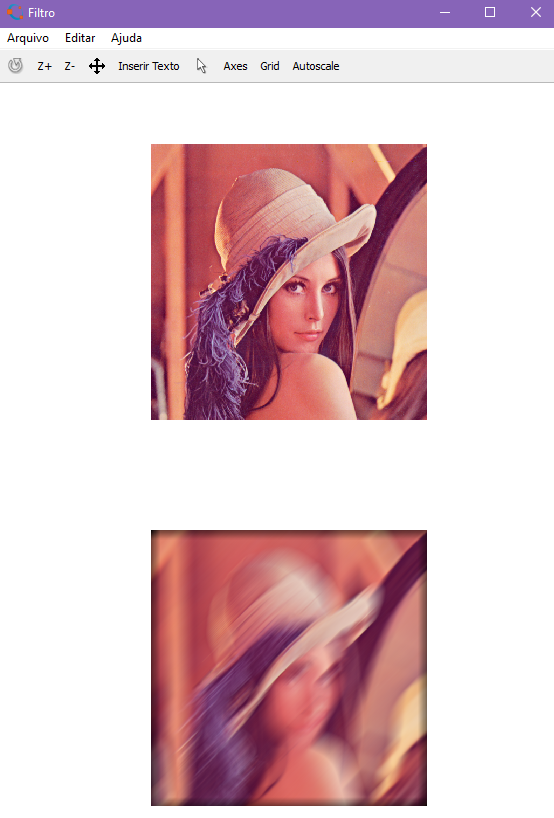
disp('Tchau!');



Filtro de Movimento – Imagem 1



Filtro de Movimento – Imagem 2



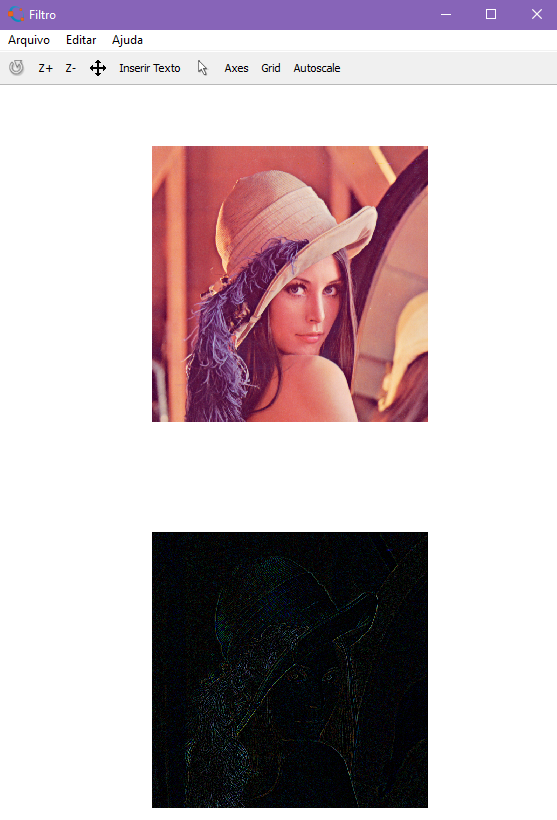
Filtro de Movimento – Imagem 2

****

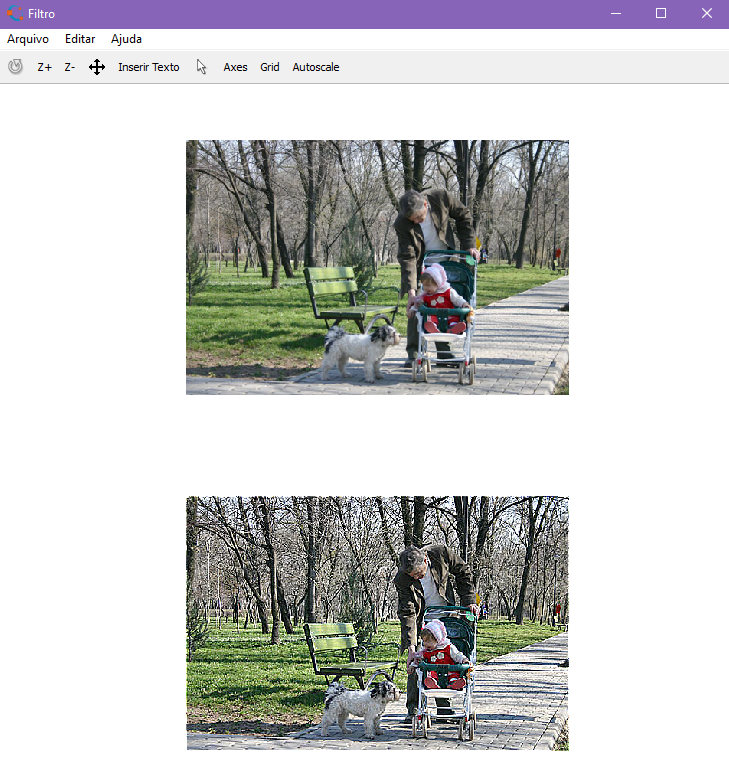
Filtro Laplaciano – Imagem 1

****

Filtro Laplaciano – Imagem 2

****

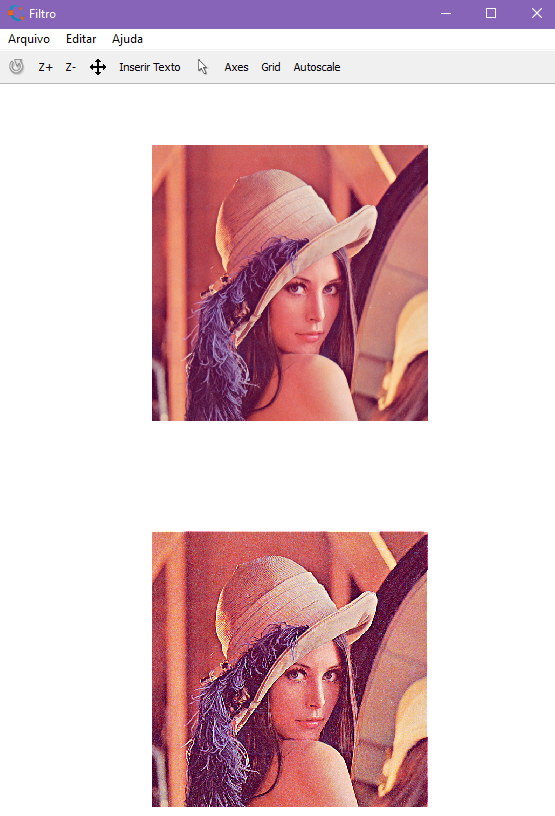
Filtro Laplaciano – Imagem 3

****

Filtro de Diminuição de Nitidez – Imagem 1

****

Filtro de Diminuição de Nitidez – Imagem 2

****

Filtro de Diminuição de Nitidez – Imagem 3

**Exercício 5**

coins = imread("Coins.png");

imhist(coins)

% Analisando o histograma, a parte da imagem do background vai até a intensidade 80

bi = roicolor(coins, 80, 255);

% Criamos uma imagem binária ‘bi’ a partir da função ROI (region of interest), que seleciona a área do histograma que se deseja selecionar. É como se cortasse o histograma em 80, deixando 1’s na região de interesse (de 80 a 255) e 0’s fora da região de interesse (de 0 a 80: o background)

new\_coins = bi.\*coins;

% Multiplicando a imagem ‘coins’ pela imagem binária criada, encontramos uma nova imagem com o background todo preto: ‘new\_coins’

figure('Name','Coins','NumberTitle','off')

subplot(2,2,[1,2]);

% A imagem original ‘coins’ é exibida na parte superior da figura

imshow(coins)

subplot(2,2,[3,4]);

% A imagem com background preto ‘new\_coins’ é exibida na parte inferior da figura

imshow(new\_coins)



% Demonstração das imagens com histogramas

figure('Name','Coins Histogram','NumberTitle','off')

subplot(2,2,[1,2]);

% O histograma da imagem original ‘coins’ é exibido na parte superior da figura

imhist(coins)

subplot(2,2,[3,4]);

% O histograma da imagem com background preto ‘new\_coins’ é exibido na parte inferior da figura

imhist(new\_coins)

