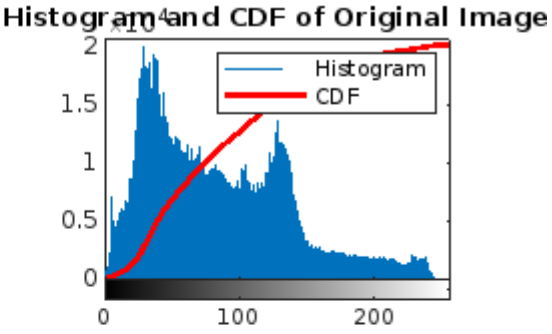
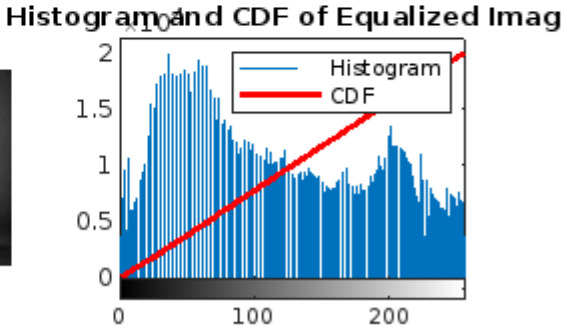
```
% Prompt the user to select an image file
[selectedFile, selectedPath] = uigetfile({'*.jpg;*.jpeg;*.png;*.bmp', 'Image
Files (*.jpg, *.jpeg, *.png, *.bmp)'}, 'Select an Image File');
if isequal(selectedFile, 0)
    disp('No file selected. Exiting...');
    return; % Exit if no file is selected
end
imagePath = fullfile(selectedPath, selectedFile);

% Read the input image
originalImage = imread(imagePath);
if size(originalImage, 3) == 3
    originalImage = rgb2gray(originalImage); % Convert to grayscale if the
image is RGB
end

% Get the dimensions of the image
[imageHeight, imageWidth] = size(originalImage);

% Calculate the histogram for the original image
originalHistogram = zeros(256, 1); % Initialize histogram
for row = 1:imageHeight
    for col = 1:imageWidth
        pixelValue = originalImage(row, col) + 1; % Adjust for 1-based
indexing
        originalHistogram(pixelValue) = originalHistogram(pixelValue) + 1;
    end
end

% Normalize the histogram to get the probability density function (PDF)
originalPDF = originalHistogram / (imageHeight * imageWidth);

% Calculate the cumulative distribution function (CDF)
originalCDF = cumsum(originalPDF);

% Map the intensities to equalized values
equalizedMapping = round(originalCDF * 255);

% Create the histogram-equalized image
equalizedImage = zeros(size(originalImage));
for row = 1:imageHeight
    for col = 1:imageWidth
        equalizedImage(row, col) = equalizedMapping(originalImage(row, col) +
1); % Adjust for 1-based indexing
    end
end
equalizedImage = uint8(equalizedImage); % Convert to uint8 for display

% Calculate the histogram for the equalized image
equalizedHistogram = zeros(256, 1); % Initialize histogram
for row = 1:imageHeight
    for col = 1:imageWidth
```

```
        pixelValue = equalizedImage(row, col) + 1; % Adjust for 1-based
indexing
        equalizedHistogram(pixelValue) = equalizedHistogram(pixelValue) + 1;
    end
end

% Normalize the histogram to get the PDF for the equalized image
equalizedPDF = equalizedHistogram / (imageHeight * imageWidth);

% Calculate the CDF for the equalized image
equalizedCDF = cumsum(equalizedPDF);

% Display the results
figure;

% Original Image and its histogram
subplot(2, 2, 1);
imshow(originalImage);
title('Original Image');

subplot(2, 2, 2);
imhist(originalImage);
hold on;
plot(originalCDF * max(originalHistogram), 'r', 'LineWidth', 2); % Scale CDF
for visualization
legend('Histogram', 'CDF');
title('Histogram and CDF of Original Image');

% Equalized Image and its histogram
subplot(2, 2, 3);
imshow(equalizedImage);
title('Equalized Image');

subplot(2, 2, 4);
imhist(equalizedImage);
hold on;
plot(equalizedCDF * max(equalizedHistogram), 'r', 'LineWidth', 2); % Scale
CDF for visualization
legend('Histogram', 'CDF');
title('Histogram and CDF of Equalized Image');
```


Original Image


Histogram and CDF of Original Image
x 10^4
2
1.5
1
0.5
0
0
100
200
Histogram
CDF


Equalized Image


Histogram and CDF of Equalized Imag
x 10^4
2
1.5
1
0.5
0
0
100
200
Histogram
CDF

Published with MATLAB® R2024b