
```
clc;
clear all;
close all;

% Load and preprocess the input image
img = imread("Lin-Dan.jpg");

% Convert RGB image to grayscale if necessary
if size(img, 3) == 3
    img = rgb2gray(img);
end

% Built-in MATLAB functions for histogram equalization (commented out)
% enhanced_img = histeq(img);
% figure;
% subplot(2,2,1);
% imshow(img);
% title('Input Grayscale Image');
%
% subplot(2,2,2);
% imhist(img);
% title('Input Image Histogram');
%
% subplot(2,2,3);
% imshow(enhanced_img);
% title('Enhanced Image');
%
% subplot(2,2,4);
% imhist(enhanced_img);
% title('Enhanced Image Histogram');

% Custom implementation of histogram equalization
[rows, cols] = size(img);
totalPixels = rows * cols;

% Initialize frequency array for intensity levels (0-255)
freq = zeros(256, 1);

% Count frequency of each intensity level
for row = 1:rows
    for col = 1:cols
        pixelValue = img(row, col);
        freq(pixelValue + 1) = freq(pixelValue + 1) + 1;
    end
end

% Compute probability distribution function
prob = freq / totalPixels;

% Calculate cumulative distribution function
cumProb = zeros(256, 1);
cumProb(1) = prob(1);
```

```
for idx = 2:256
    cumProb(idx) = cumProb(idx - 1) + prob(idx);
end

% Map old intensity values to new ones
grayLevels = 256;
transformFunc = round((grayLevels - 1) * cumProb);

% Create output equalized image
outputImg = zeros(rows, cols, 'uint8');

for row = 1:rows
    for col = 1:cols
        originalIntensity = img(row, col);
        outputImg(row, col) = transformFunc(originalIntensity + 1);
    end
end

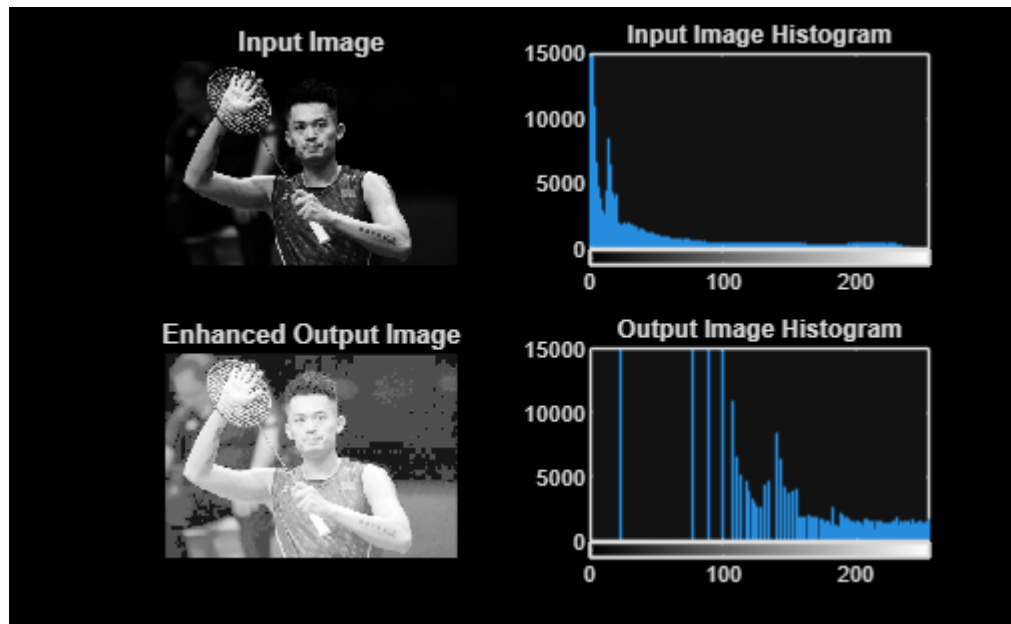
% Display results in a figure window
figure;

subplot(2,2,1);
imshow(img);
title('Input Image');

subplot(2,2,2);
imhist(img);
title('Input Image Histogram');

subplot(2,2,3);
imshow(outputImg);
title('Enhanced Output Image');

subplot(2,2,4);
imhist(outputImg);
title('Output Image Histogram');
```



Published with MATLAB® R2025b