JPEG Compression – DCT Based (Two Images)

Table of Contents

Input : Dog & Cat RGB Images Output : JPEG-like reconstructed images + PSNR, MSE

Defaults

```
clc;
clear all;
close all;
```

1. Read Input Images

```
I_dog = imread("dog.jpg");
I_cat = imread("cat.jpg");

% Resize to multiple of 8
I_dog = imresize(I_dog, [256 256]);
I_cat = imresize(I_cat, [256 256]);

% Put images in a cell array for easy looping
images = {I_dog, I_cat};
names  = {"Dog Image", "Cat Image"};
```

JPEG Parameters

```
blockSize = 8;
N = 8;
```

```
% Manual DCT Matrix
D = zeros(N);
for u = 0:N-1
    for x = 0:N-1
        if u == 0
            alpha = sqrt(1/N);
        else
            alpha = sqrt(2/N);
        end
        D(u+1, x+1) = alpha * cos((2*x+1)*u*pi/(2*N));
    end
end

% JPEG Luminance Quantization Table
QY = [ ...
16 11 10 16 24 40 51 61;
12 12 14 19 26 58 60 55;
14 13 16 24 40 57 69 56;
14 17 22 29 51 87 80 62;
18 22 37 56 68 109 103 77;
24 35 55 64 81 104 113 92;
49 64 78 87 103 121 120 101;
72 92 95 98 112 100 103 99];
```

========== MAIN LOOP ==========

```
for idx = 1:2

    I = images{idx};
```

RGB → YCbCr

```
    Iycbcr = rgb2ycbcr(I);
    Y  = Iycbcr(:,:,1);
    Cb = Iycbcr(:,:,2);
    Cr = Iycbcr(:,:,3);
```

Chroma Subsampling (4:2:0)

```
    Cb_ds = Cb(1:2:end, 1:2:end);
    Cr_ds = Cr(1:2:end, 1:2:end);
```

Level Shift

```
    Y = double(Y) - 128;
```

DCT

```
[H, W] = size(Y);
Y_dct = zeros(H, W);

for i = 1:blockSize:H
    for j = 1:blockSize:W
        block = Y(i:i+7, j:j+7);
        Y_dct(i:i+7, j:j+7) = D * block * D';
    end
end
```

Quantization

```
Y_q = zeros(size(Y_dct));
for i = 1:blockSize:H
    for j = 1:blockSize:W
        block = Y_dct(i:i+7, j:j+7);
        Y_q(i:i+7, j:j+7) = round(block ./ QY);
    end
end
```

De-Quantization

```
Y_dq = zeros(size(Y_q));
for i = 1:blockSize:H
    for j = 1:blockSize:W
        block = Y_q(i:i+7, j:j+7);
        Y_dq(i:i+7, j:j+7) = block .* QY;
    end
end
```

Inverse DCT

```
Y_rec = zeros(size(Y_dq));
for i = 1:blockSize:H
    for j = 1:blockSize:W
        block = Y_dq(i:i+7, j:j+7);
        Y_rec(i:i+7, j:j+7) = D' * block * D;
    end
end
```

Inverse Level Shift

```
Y_rec = uint8(min(max(Y_rec + 128, 0), 255));
```

Chroma Upsampling

```
Cb_rec = imresize(Cb_ds, 2, 'bilinear');
Cr_rec = imresize(Cr_ds, 2, 'bilinear');
```

Reconstruct RGB Image

```
Iycbcr_rec = cat(3, Y_rec, uint8(Cb_rec), uint8(Cr_rec));
I_rec = ycbcr2rgb(Iycbcr_rec);
```

Quality Metrics

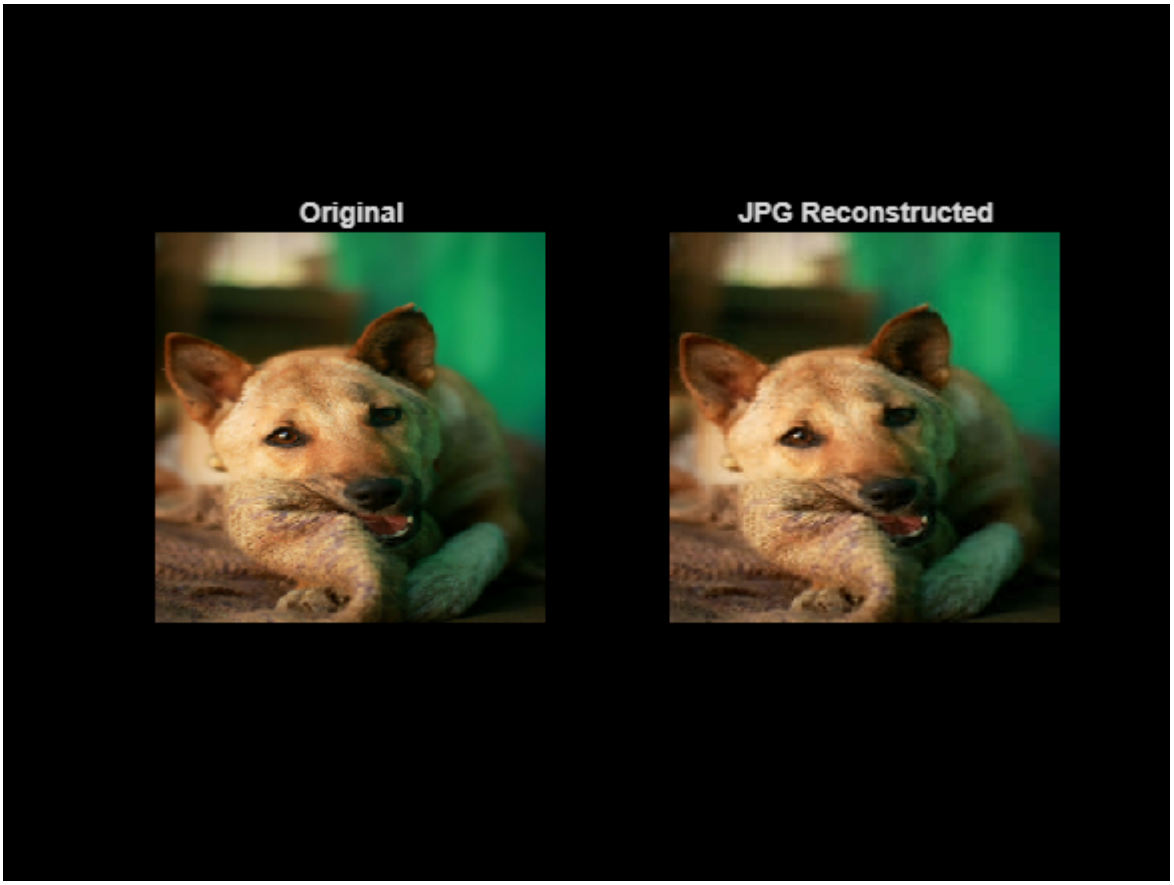
```
mse = mean((double(I) - double(I_rec)).^2, 'all');
psnr_val = 10 * log10(255^2 / mse);
```

Display Results

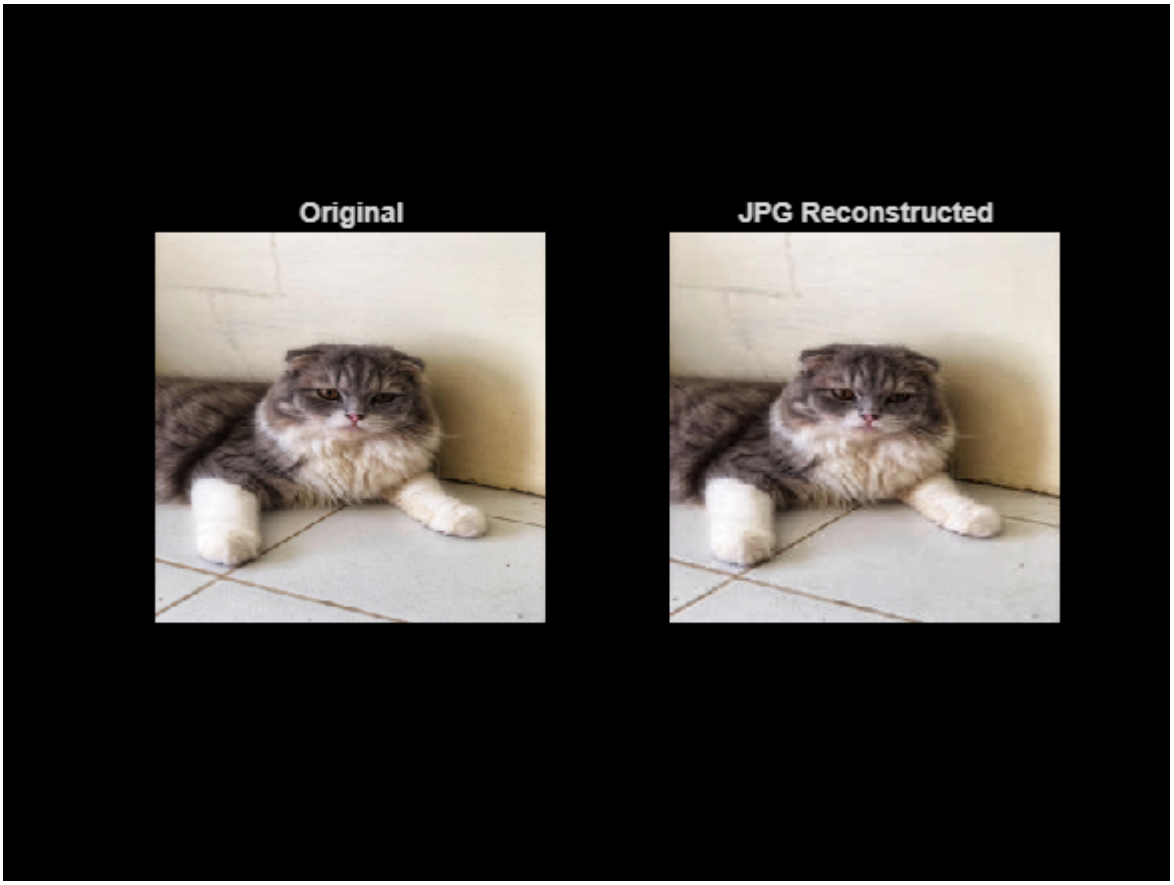
```
figure("Name", names{idx});
subplot(1,2,1), imshow(I), title("Original");
subplot(1,2,2), imshow(I_rec), title("JPG Reconstructed");

fprintf("\n%s\n", names{idx});
fprintf("MSE  = %.4f\n", mse);
fprintf("PSNR = %.2f dB\n", psnr_val);
```

```
Dog Image
MSE  = 17.8523
PSNR = 35.61 dB
```




```
Cat Image
MSE  = 15.6584
PSNR = 36.18 dB
```




```
end
```

Published with MATLAB® R2025b