JPEG Compression - DCT Based

Anubhav Rathore

Input : RAW Image Output : Reconstructed Image after JPEG-like compression

Notes: - Manual DCT / IDCT - Luminance (Y) compression only - Chroma subsampling (4:2:0) - This project shows JPEG Signal Compression, doesnt actually convert into JPEG Compressed file in storage. But depicts the compression technique, except file handling.

Contents

Defaults

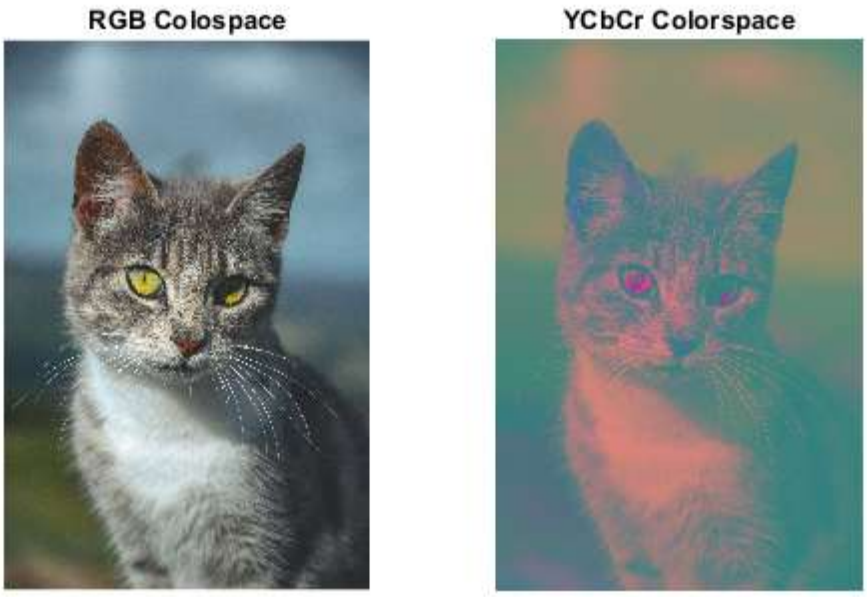
```
clear all;
clc;
close all;
```

1. Inputs, Variables, Constants

```
I = imread("cat.png");

figure("Name","Original Image");
imshow(I);
title("Original RGB Image");
```

2. RGB -> YCbCr

```
Iycbcr = rgb2ycbcr(I);

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

figure("Name","Color Space Conversion");
subplot(1,2,1), imshow(I), title("RGB Colospace");
subplot(1,2,2), imshow(Iycbcr), title("YCbCr Colorspace");
```



3. 8x8 Blocks Generation

```
blockSize = 8;
[H,W] = size(Y);
H8 = floor(H/blockSize)*blockSize;
W8 = floor(W/blockSize)*blockSize;
Y  = Y(1:H8, 1:W8);
Cb = Cb(1:H8, 1:W8);
Cr = Cr(1:H8, 1:W8);
```

4. Chroma Subsampling (4:2:0)

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

figure("Name","Chroma Subsampling (4:2:0)");
subplot(1,3,1), imshow(Y), title("Y - Luminance");
subplot(1,3,2), imshow(Cb_ds), title("Cb - Subsampled");
subplot(1,3,3), imshow(Cr_ds), title("Cr - Subsampled");
```

5. Level Shifting (Zero-Mean for DCT)

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

6. Manual DCT Matrix (8x8)

```
N = 8;
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
% DCT relation: F = D * block * D'
```

7. DCT Blockwise on Y

```
Y_dct = zeros(size(Y));

for i = 1:blockSize:H8
    for j = 1:blockSize:W8
        block = Y(i:i+7, j:j+7);
        Y_dct(i:i+7, j:j+7) = D * block * D';
```

```
    end
end
```

8. JPEG Luminance Quantization Table

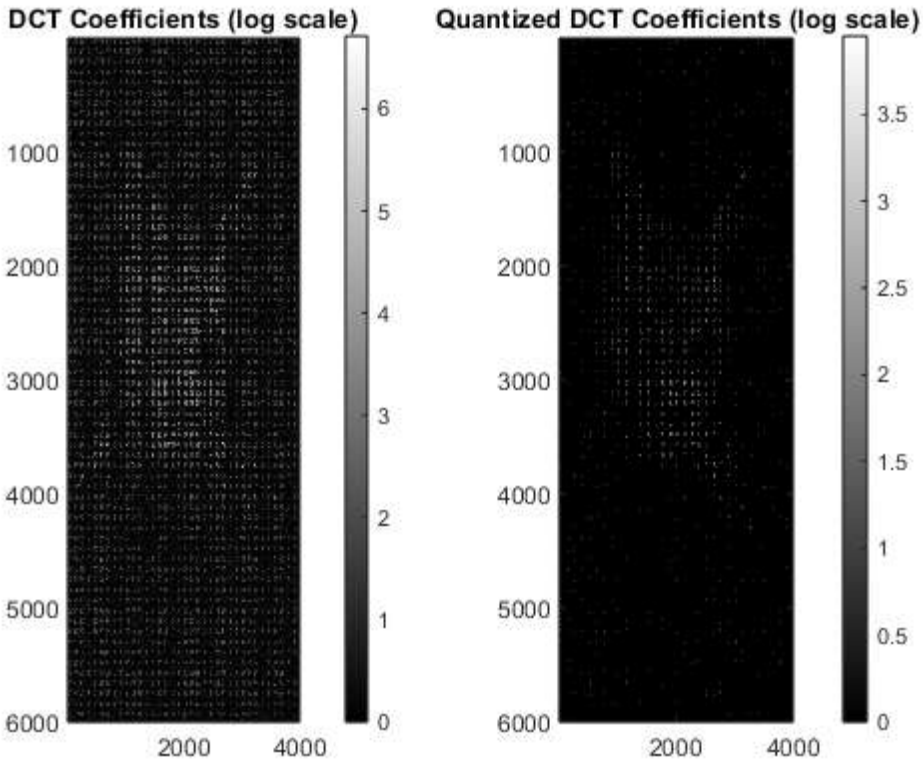
```
QY = [ ...
16 11 10 16 24 40 51 61;
12 12 14 19 26 58 60 55;
14 13 16 24 40 57 69 56;
14 17 22 29 51 87 80 62;
18 22 37 56 68 109 103 77;
24 35 55 64 81 104 113 92;
49 64 78 87 103 121 120 101;
72 92 95 98 112 100 103 99];
```

9. Quantization DCT Coeffs Blockwise

```
Y_q = zeros(size(Y_dct));

for i = 1:blockSize:H8
    for j = 1:blockSize:W8
        block = Y_dct(i:i+7, j:j+7);
        Y_q(i:i+7, j:j+7) = round(block ./ QY);
    end
end
```

10. Visualization DCT Coeffs

```
figure("Name","DCT Energy Compaction");
subplot(1,2,1);
imagesc(log(abs(Y_dct)+1)), colormap gray,  colorbar;
title("DCT Coefficients (log scale)");

subplot(1,2,2);
imagesc(log(abs(Y_q)+1)), colormap gray,  colorbar;
title("Quantized DCT Coefficients (log scale)");
```



11. De-Quantization

```
Y_dq = zeros(size(Y_q));

for i = 1:blockSize:H8
    for j = 1:blockSize:W8
        block = Y_q(i:i+7, j:j+7);
        Y_dq(i:i+7, j:j+7) = block .* QY;
    end
end
```

12. Inverse DCT

```
Y_rec = zeros(size(Y_dq));

for i = 1:blockSize:H8
    for j = 1:blockSize:W8
        block = Y_dq(i:i+7, j:j+7);
        Y_rec(i:i+7, j:j+7) = D' * block * D;
    end
end
```

13. Inverse Level Shift

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

14. Chroma Upsampling

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

Cb_rec = uint8(Cb_rec(1:H8,1:W8));
Cr_rec = uint8(Cr_rec(1:H8,1:W8));
```

15. Recombine channels and YCbCr -> RGB

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

16. Final Visual Comparison

```
figure("Name","JPEG Compression Result");

subplot(2,1,1);
imshow(I(1:H8,1:W8,:));
title("Original Image");

subplot(2,1,2);
imshow(I_rec);
title("JPEG Reconstructed Image");
```

17. Quality Metrics

```
mse = mean((double(I(1:H8,1:W8,:)) - double(I_rec)).^2, 'all');
psnr_val = 10*log10(255^2 / mse);
```

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

```
MSE  = 7.8363
PSNR = 39.19 dB
```

Published with MATLAB® R2024a