JPEG-Like Image Compression using DCT

Table of Contents

Author : Sandesh Jat

Input : RGB Image Output : Reconstructed Image after JPEG-style compression

Highlights: - Manual DCT / IDCT implementation - Luminance (Y) channel compression - Chroma subsampling (4:2:0) - Demonstrates JPEG compression flow (no actual .jpg encoding)

Initial Setup

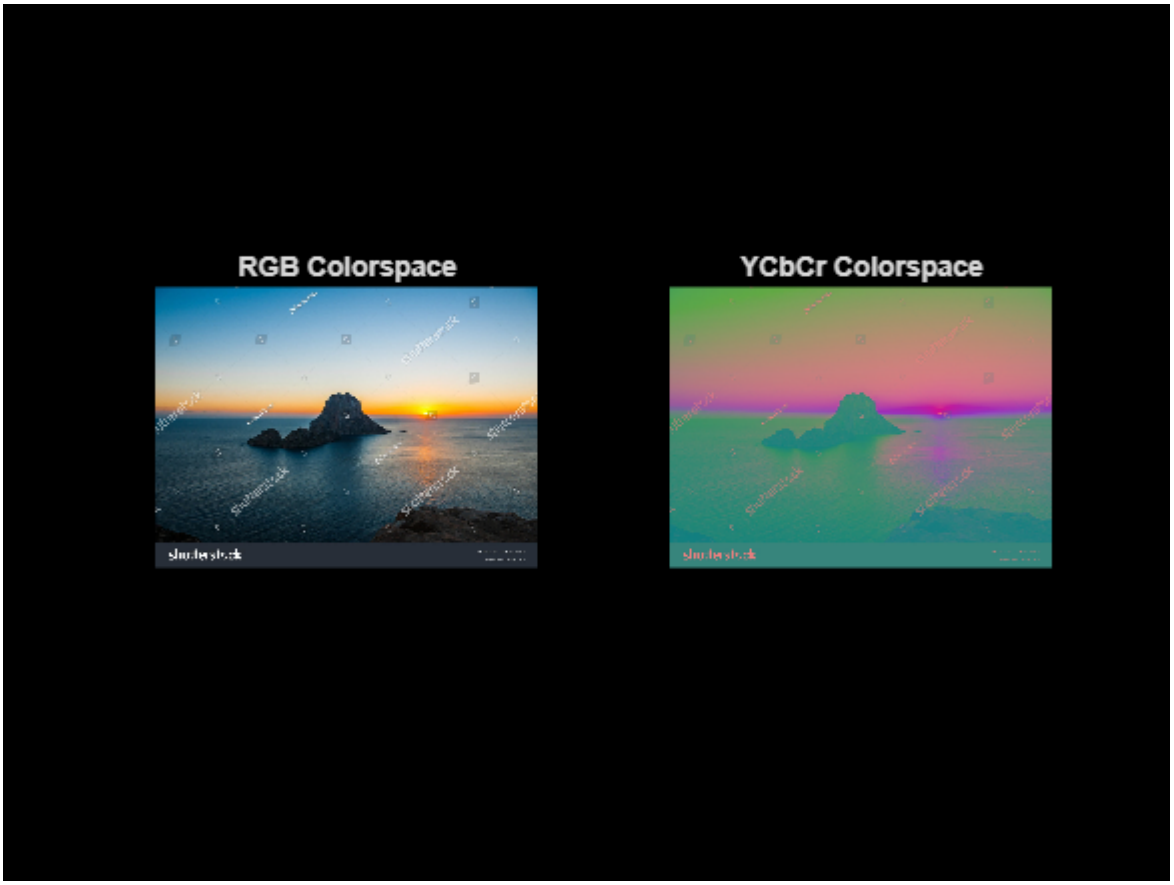
```
clc;
clear;
close all;
```

1. Read Input Image

```
imgRGB = imread("Img.png");

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

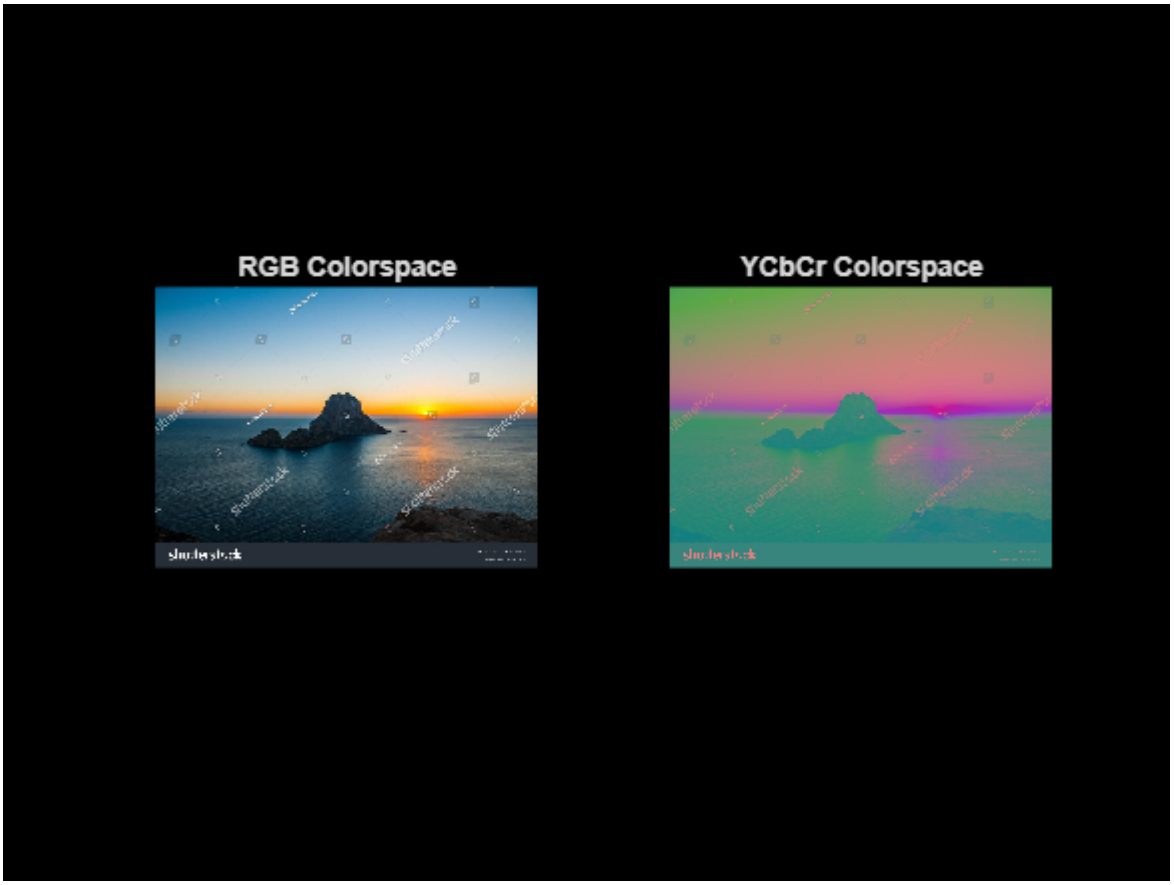
2. Convert RGB to YCbCr

```
imgYCbCr = rgb2ycbcr(imgRGB);

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

figure("Name","Color Space Conversion");
subplot(1,2,1), imshow(imgRGB),   title("RGB Colorspace");
subplot(1,2,2), imshow(imgYCbCr), title("YCbCr Colorspace");
```



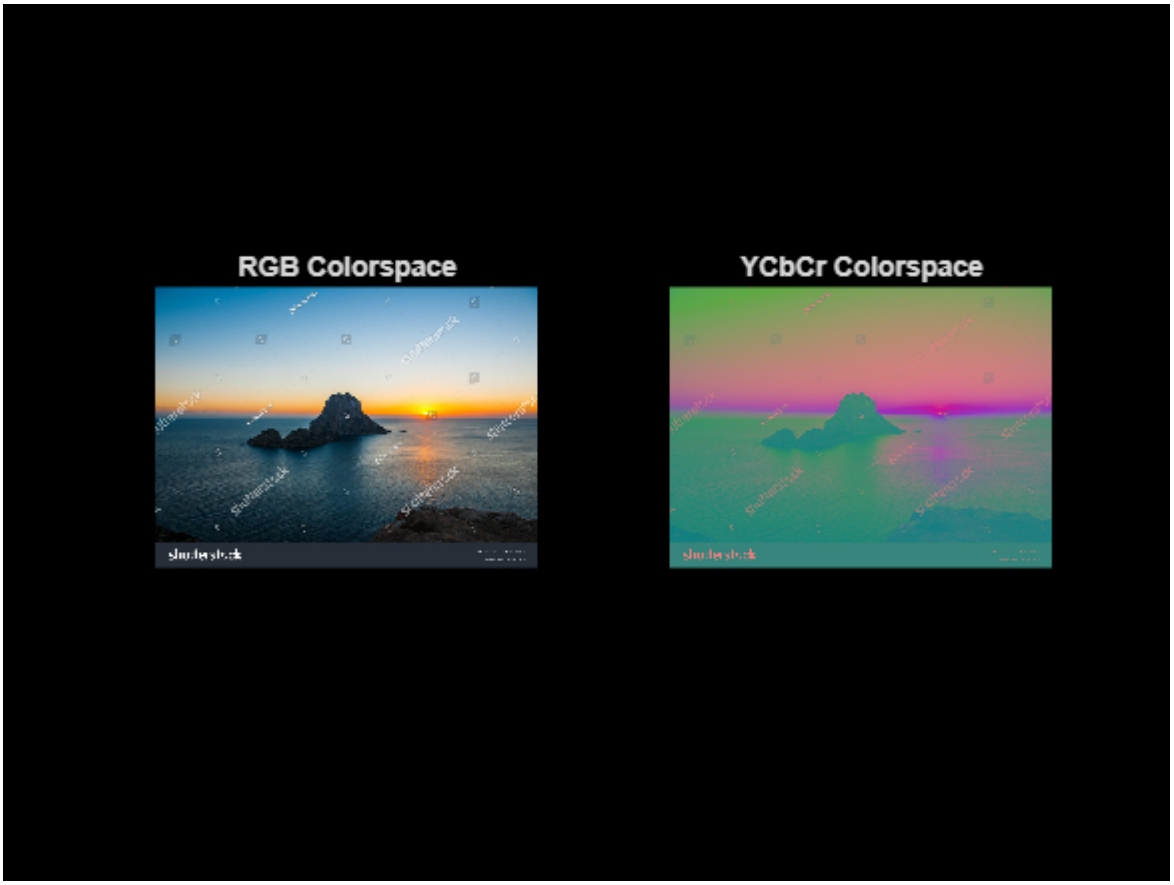
3. Generate 8x8 Blocks

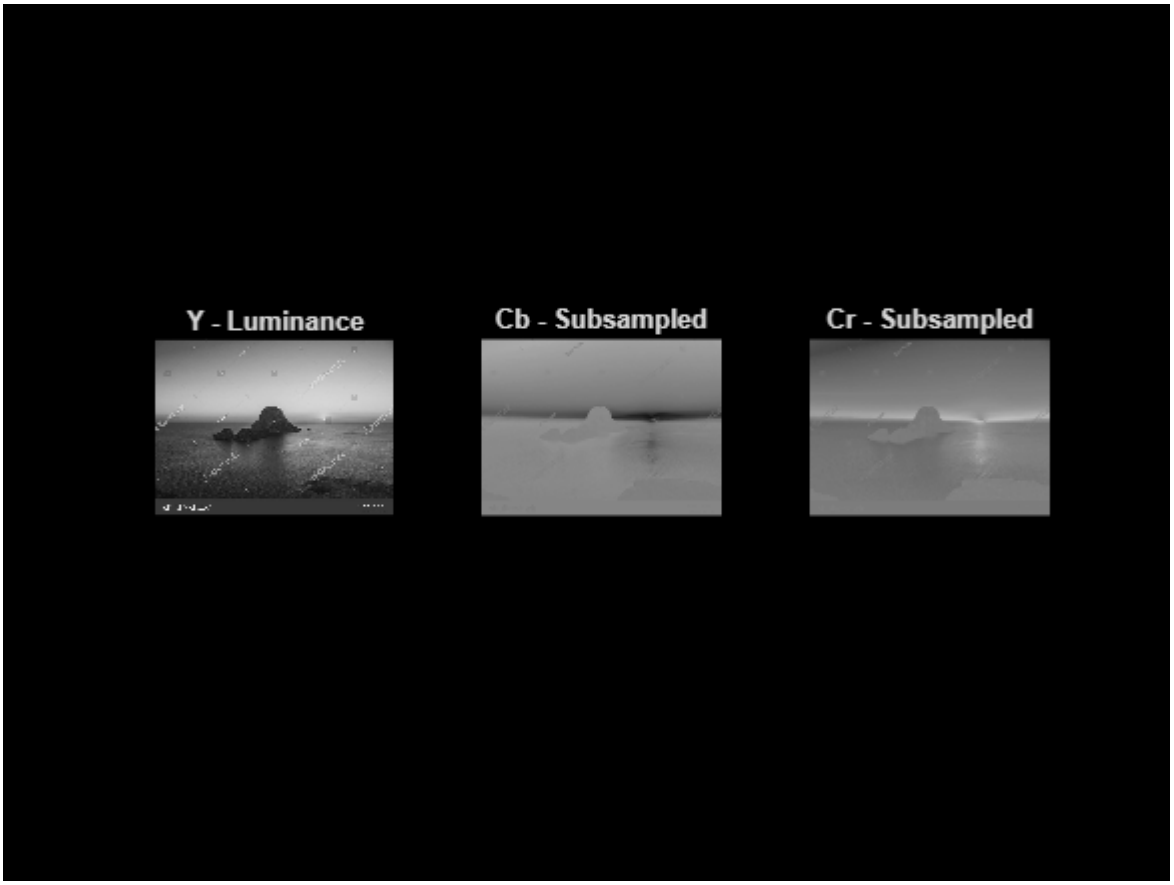
```
blk = 8;
[imgH, imgW] = size(Y);

imgH8 = floor(imgH/blk) * blk;
imgW8 = floor(imgW/blk) * blk;

Y  = Y(1:imgH8, 1:imgW8);
Cb = Cb(1:imgH8, 1:imgW8);
Cr = Cr(1:imgH8, 1:imgW8);
```



4. Chroma Subsampling (4:2:0)

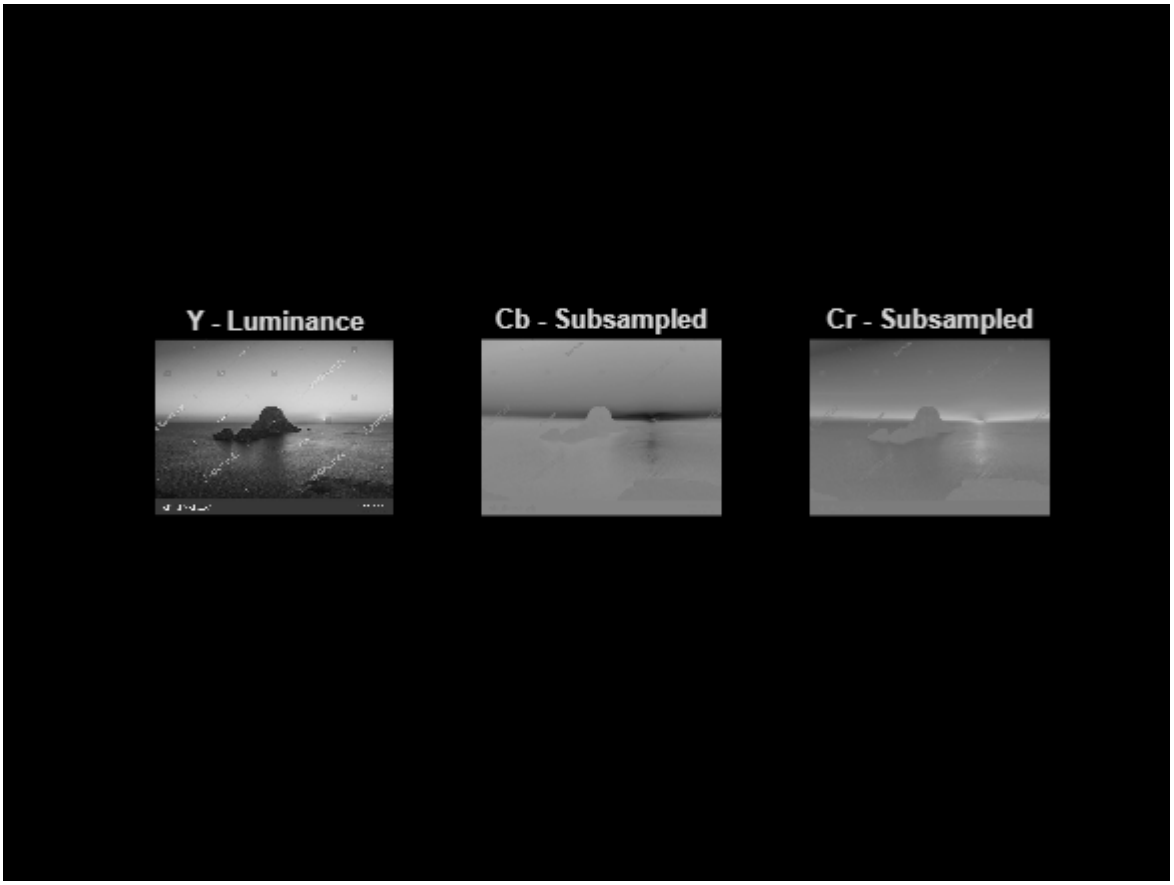
```
Cb_sub = Cb(1:2:end, 1:2:end);
Cr_sub = Cr(1:2:end, 1:2:end);

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


RGB Colorspace
YCbCr Colorspace



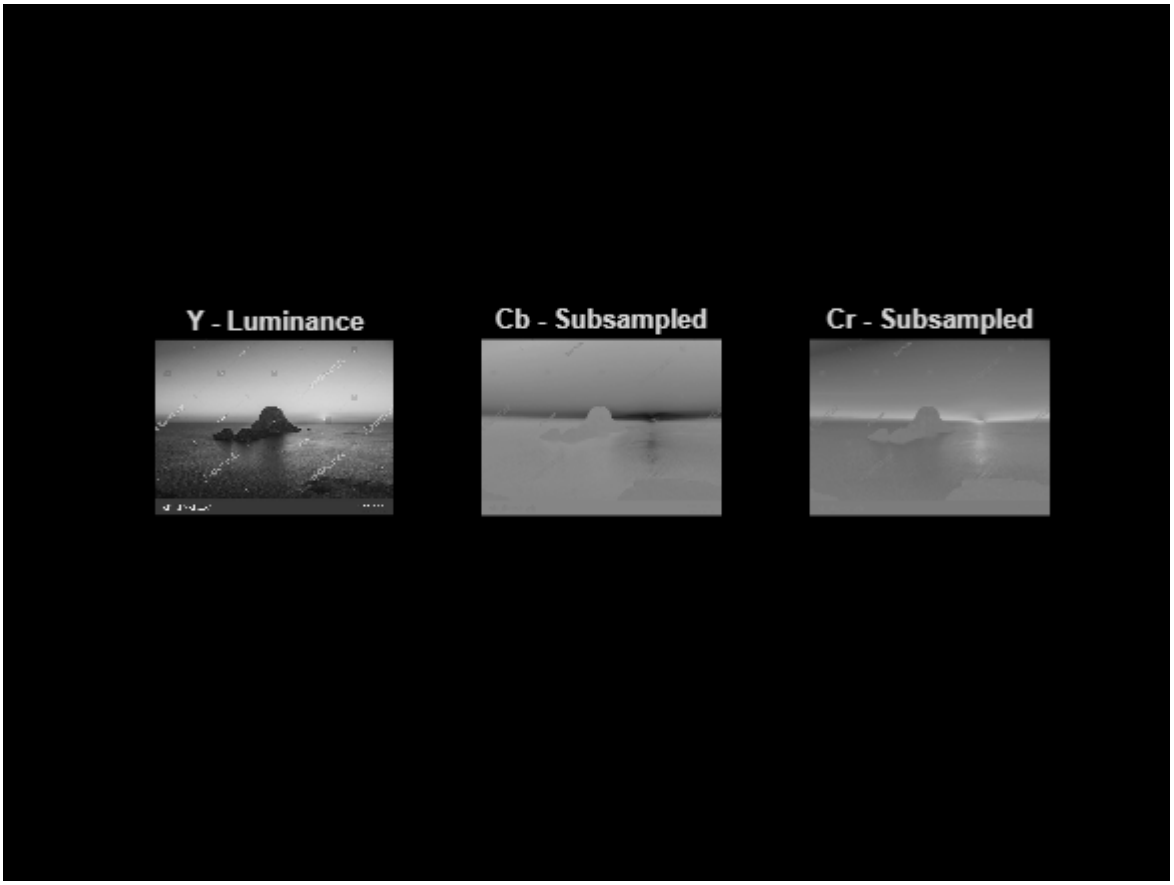
5. Level Shift (Zero Mean)

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



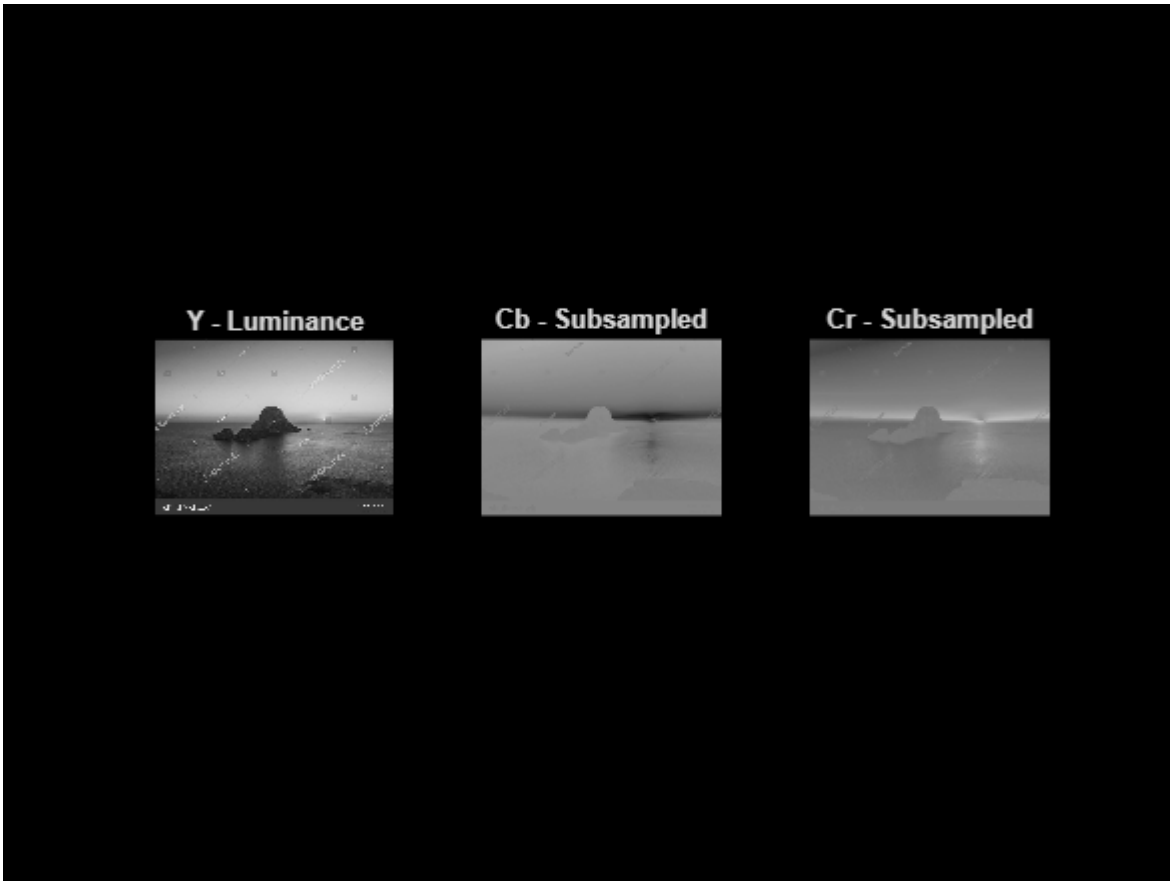
6. Construct 8x8 DCT Matrix

```
N = 8;
DCT = zeros(N);

for u = 0:N-1
    for x = 0:N-1
        if u == 0
            scale = sqrt(1/N);
        else
            scale = sqrt(2/N);
        end
        DCT(u+1, x+1) = scale * cos((2*x+1)*u*pi/(2*N));
    end
end
% Forward DCT: F = DCT * block * DCT'
```



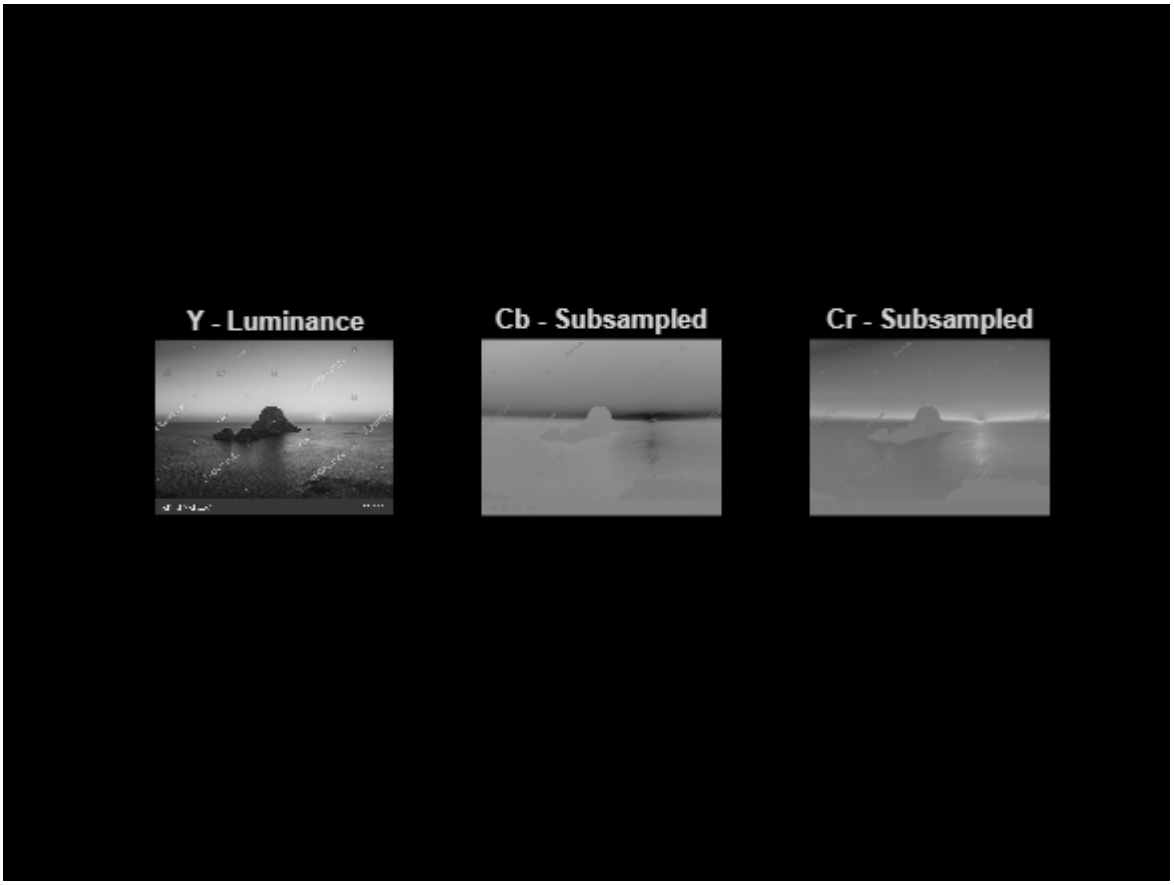
7. Block-wise DCT on Y Channel

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

for r = 1:blk:imgH8
    for c = 1:blk:imgW8
        blkY = Y(r:r+7, c:c+7);
        Y_DCT(r:r+7, c:c+7) = DCT * blkY * DCT';
    end
end
```



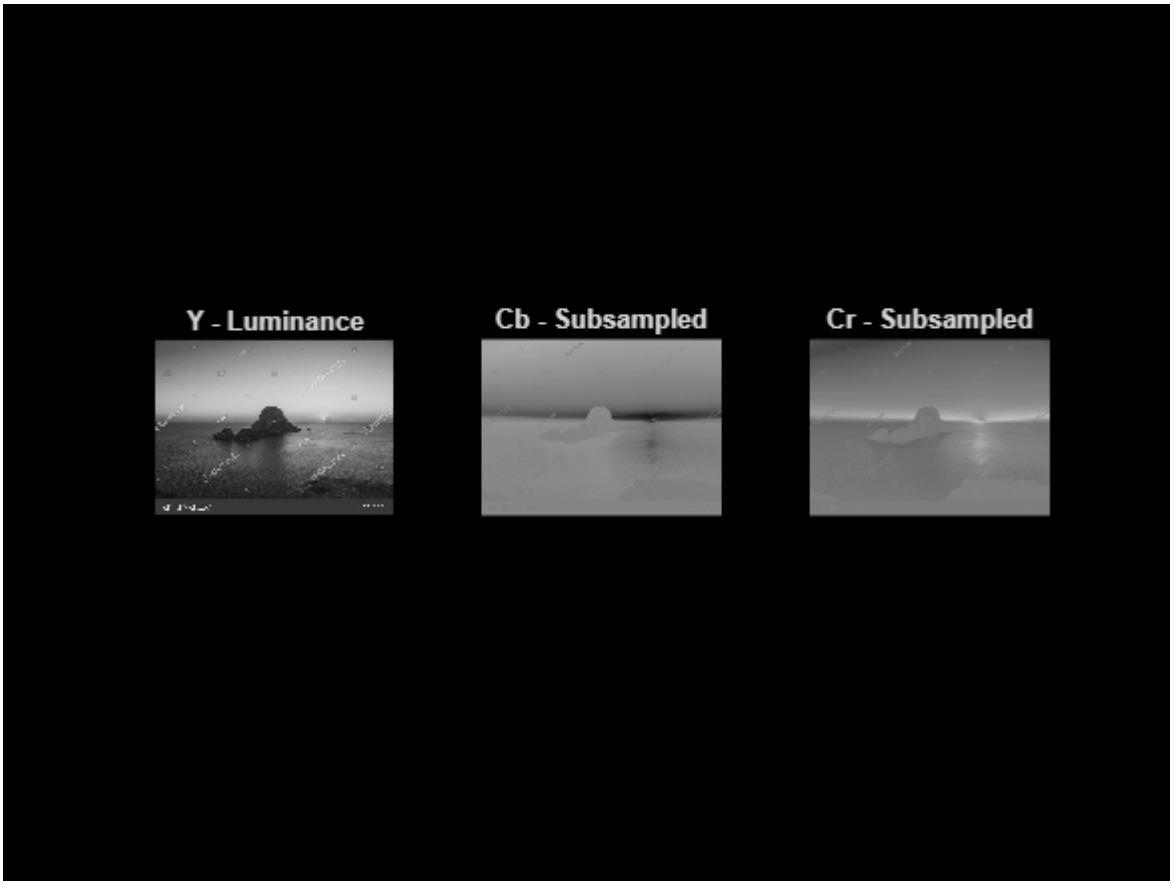
8. JPEG Luminance Quantization Matrix

```
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];
```



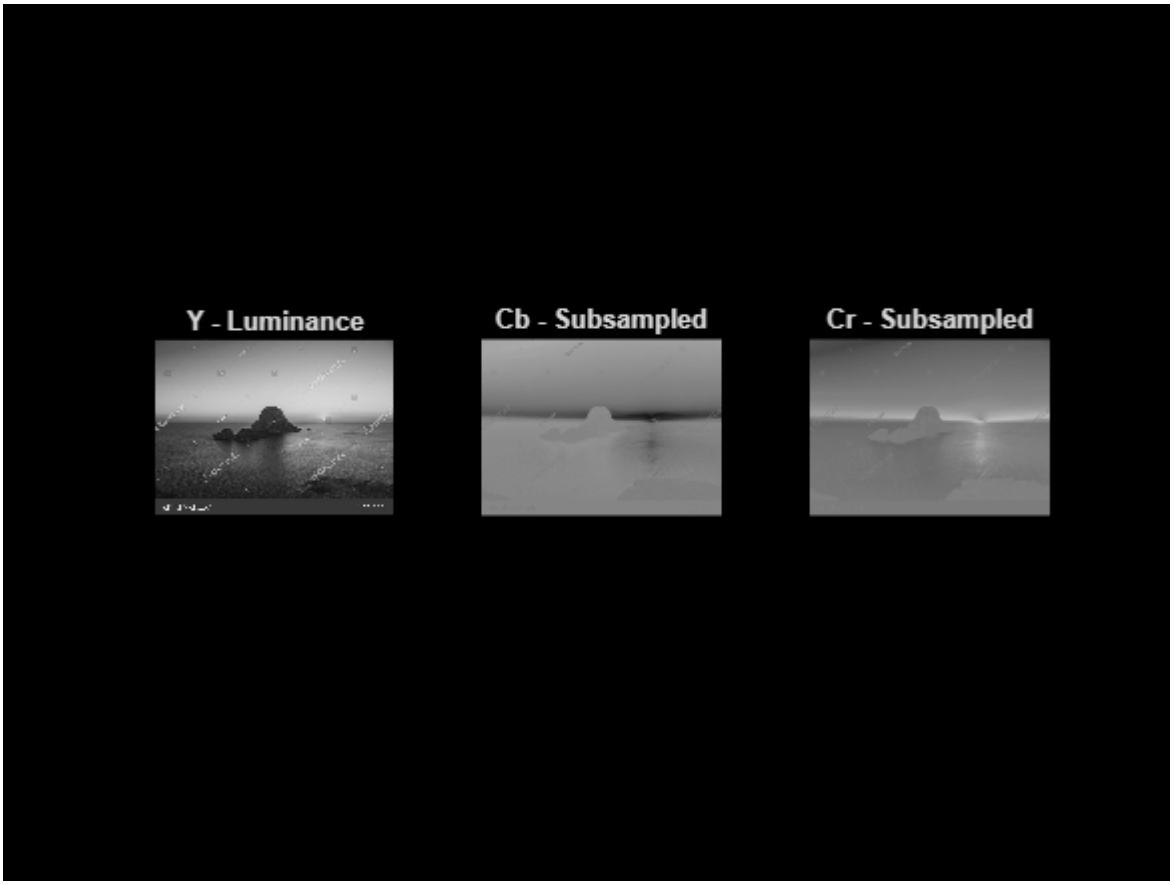
9. Quantize DCT Coefficients

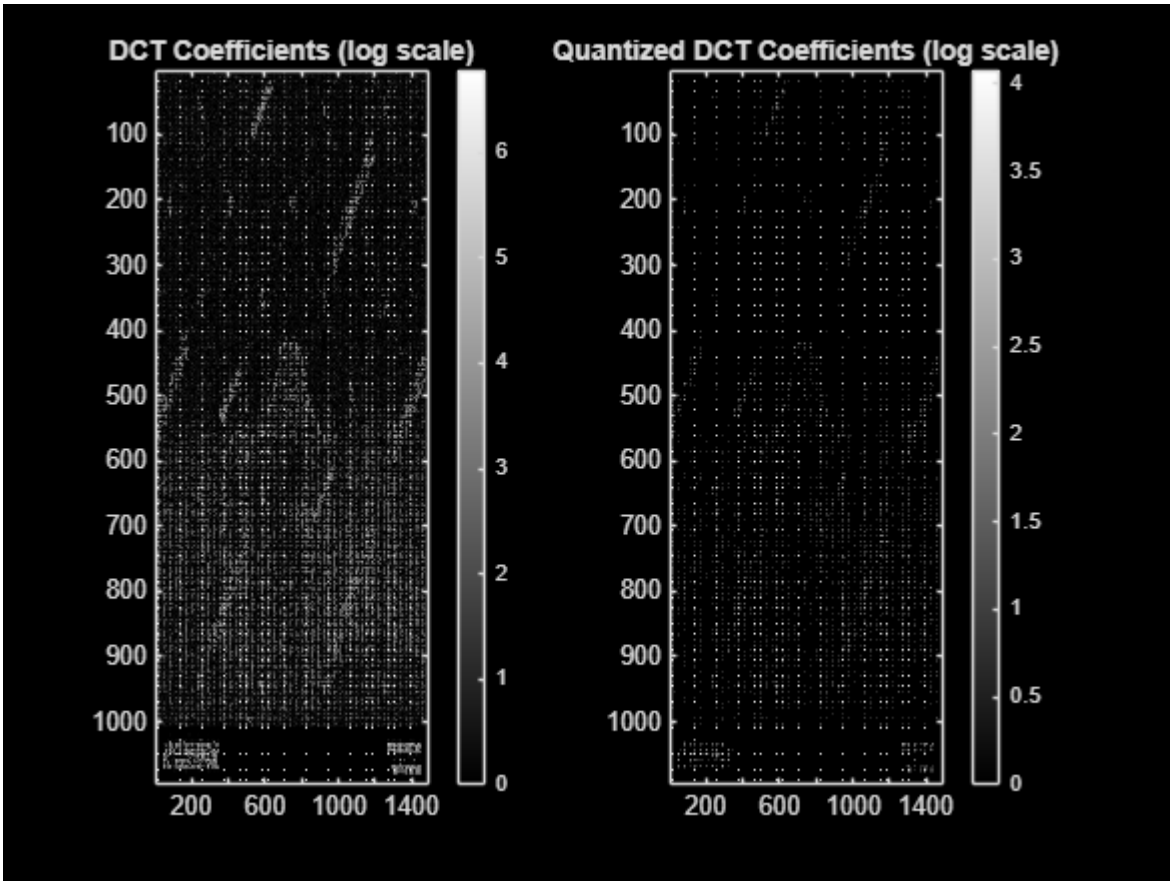
```
Y_Q = zeros(size(Y_DCT));

for r = 1:blk:imgH8
    for c = 1:blk:imgW8
        blkDCT = Y_DCT(r:r+7, c:c+7);
        Y_Q(r:r+7, c:c+7) = round(blkDCT ./ QY);
    end
end
```



10. DCT Energy Visualization

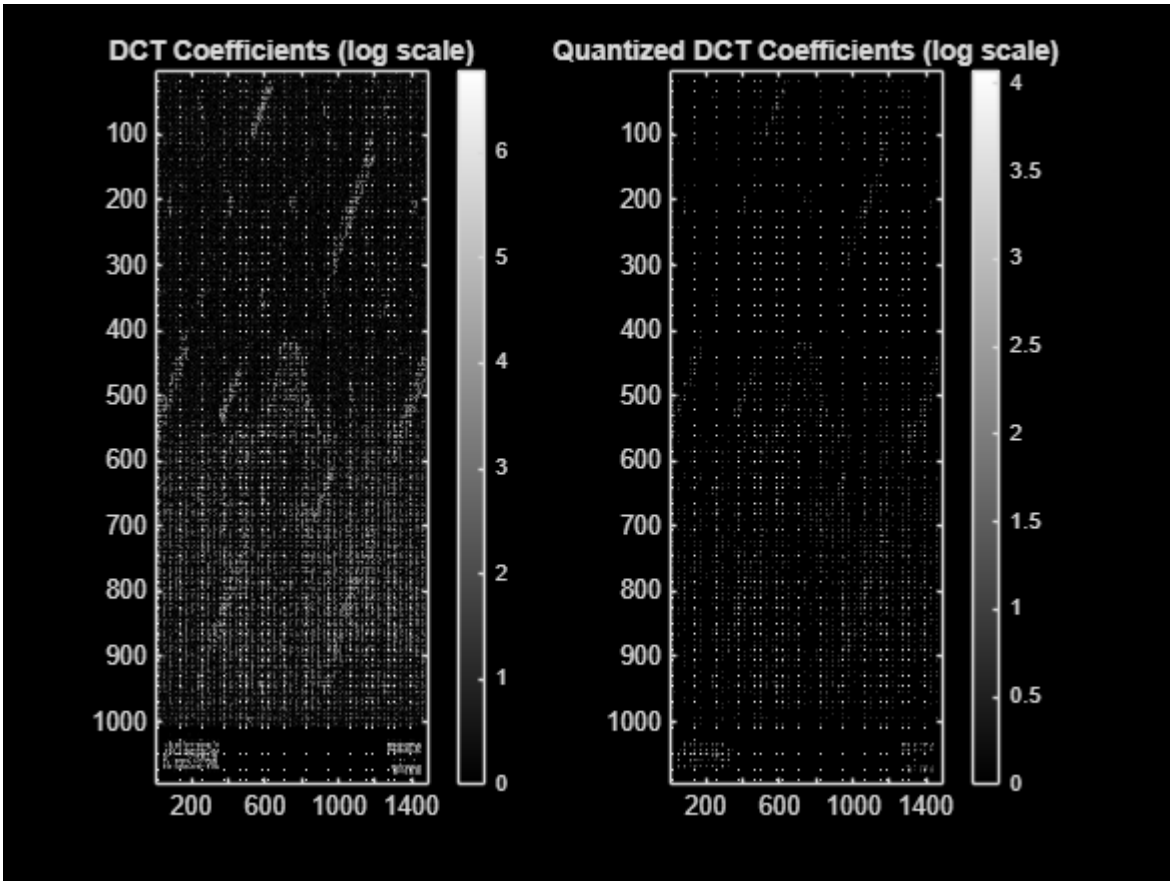
```
figure("Name","DCT Energy Compaction");

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)");
```


Y - Luminance
Cb - Subsampled
Cr - Subsampled



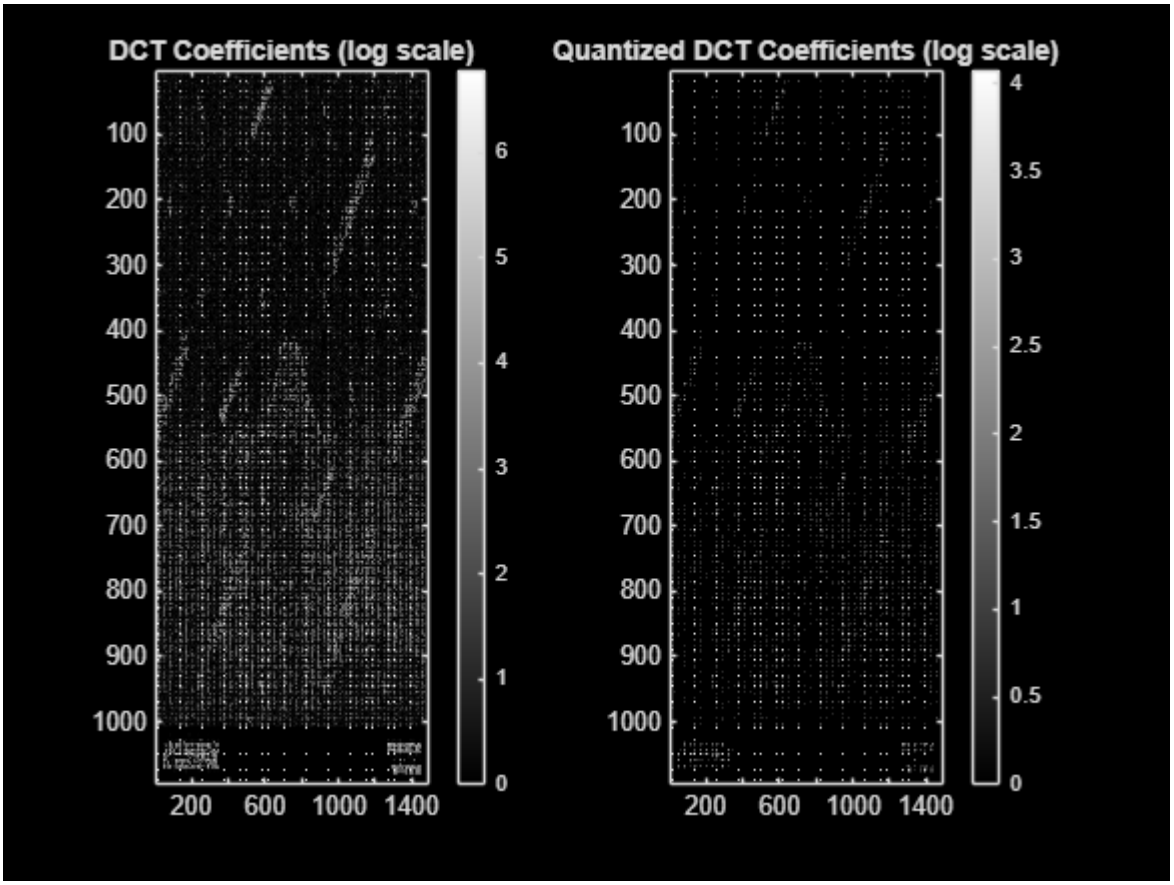
11. De-Quantization

```
Y_DQ = zeros(size(Y_Q));

for r = 1:blk:imgH8
    for c = 1:blk:imgW8
        blkQ = Y_Q(r:r+7, c:c+7);
        Y_DQ(r:r+7, c:c+7) = blkQ .* QY;
    end
end
```



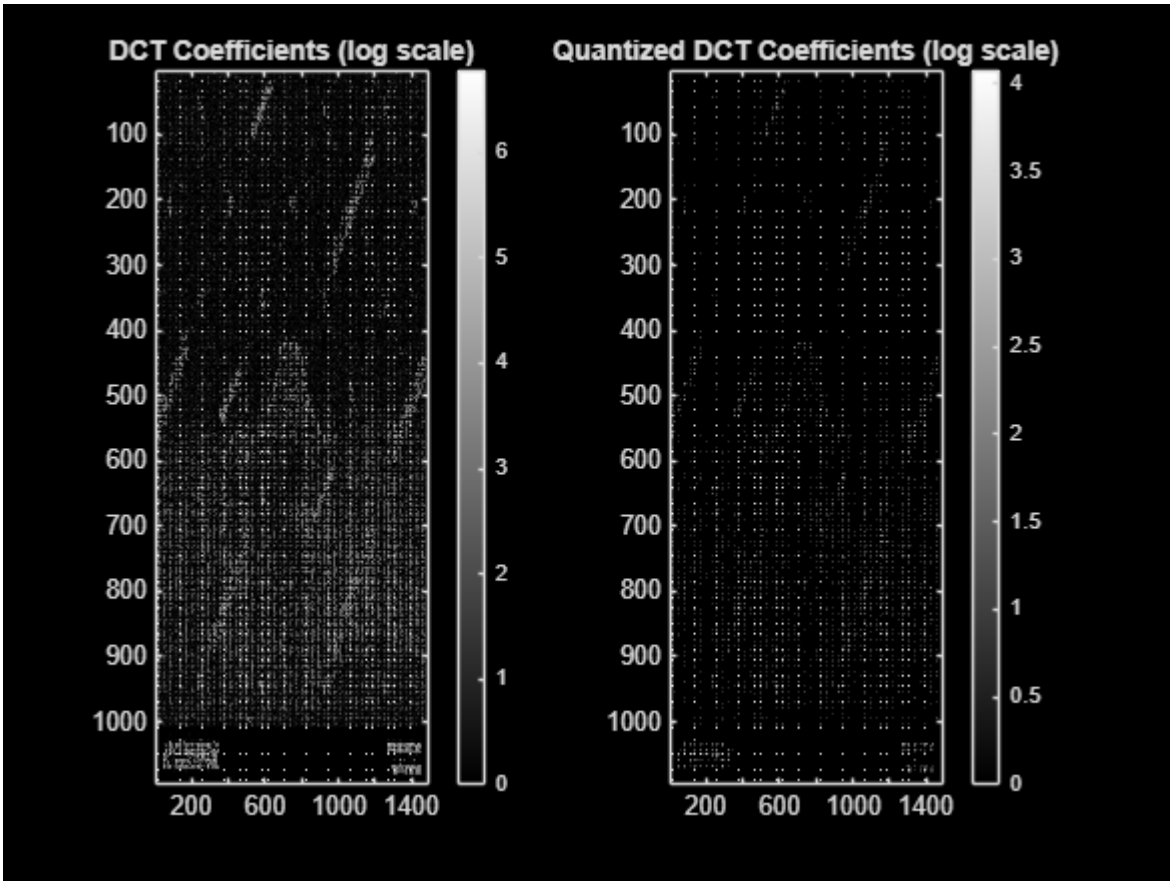
12. Inverse DCT

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

for r = 1:blk:imgH8
    for c = 1:blk:imgW8
        blkIDCT = Y_DQ(r:r+7, c:c+7);
        Y_rec(r:r+7, c:c+7) = DCT' * blkIDCT * DCT;
    end
end
```



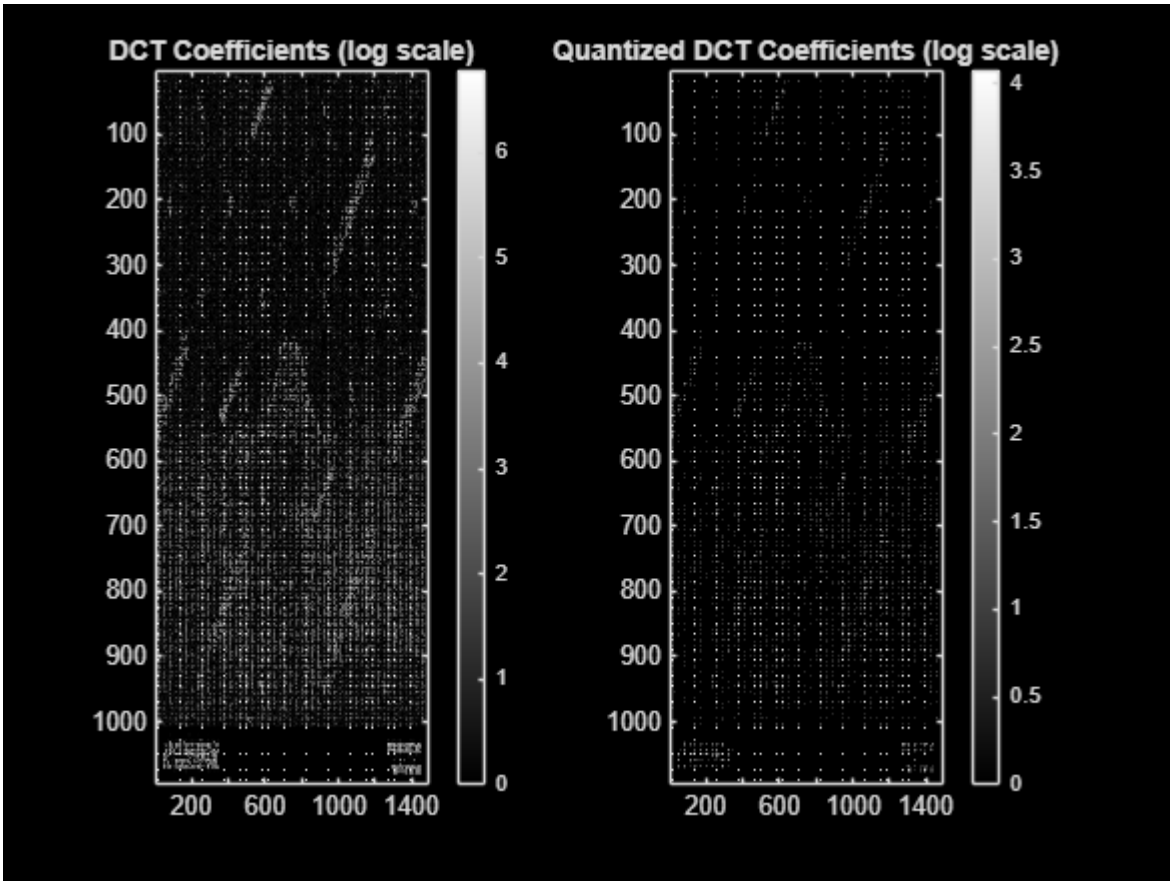
13. Inverse Level Shift

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



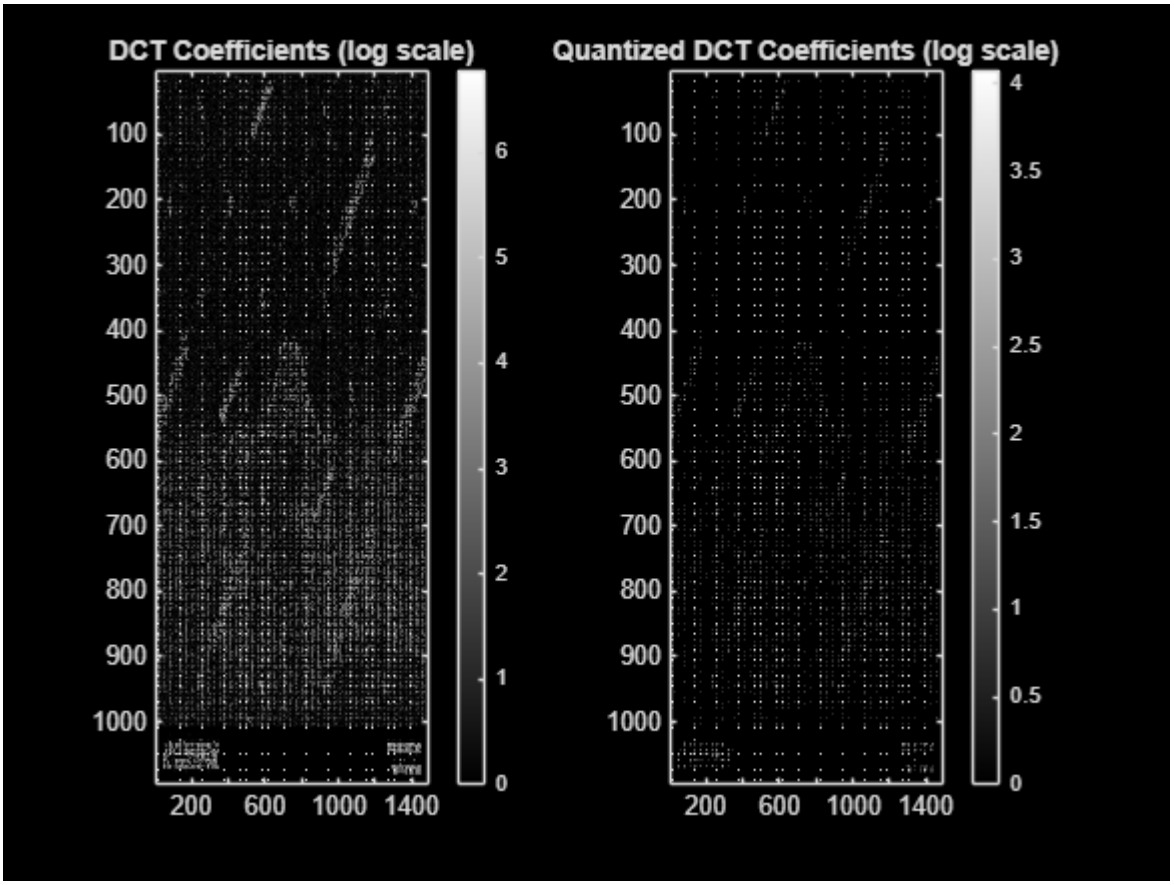
14. Chroma Upsampling

```
Cb_rec = imresize(Cb_sub, 2, 'bilinear');
Cr_rec = imresize(Cr_sub, 2, 'bilinear');

Cb_rec = uint8(Cb_rec(1:imgH8, 1:imgW8));
Cr_rec = uint8(Cr_rec(1:imgH8, 1:imgW8));
```



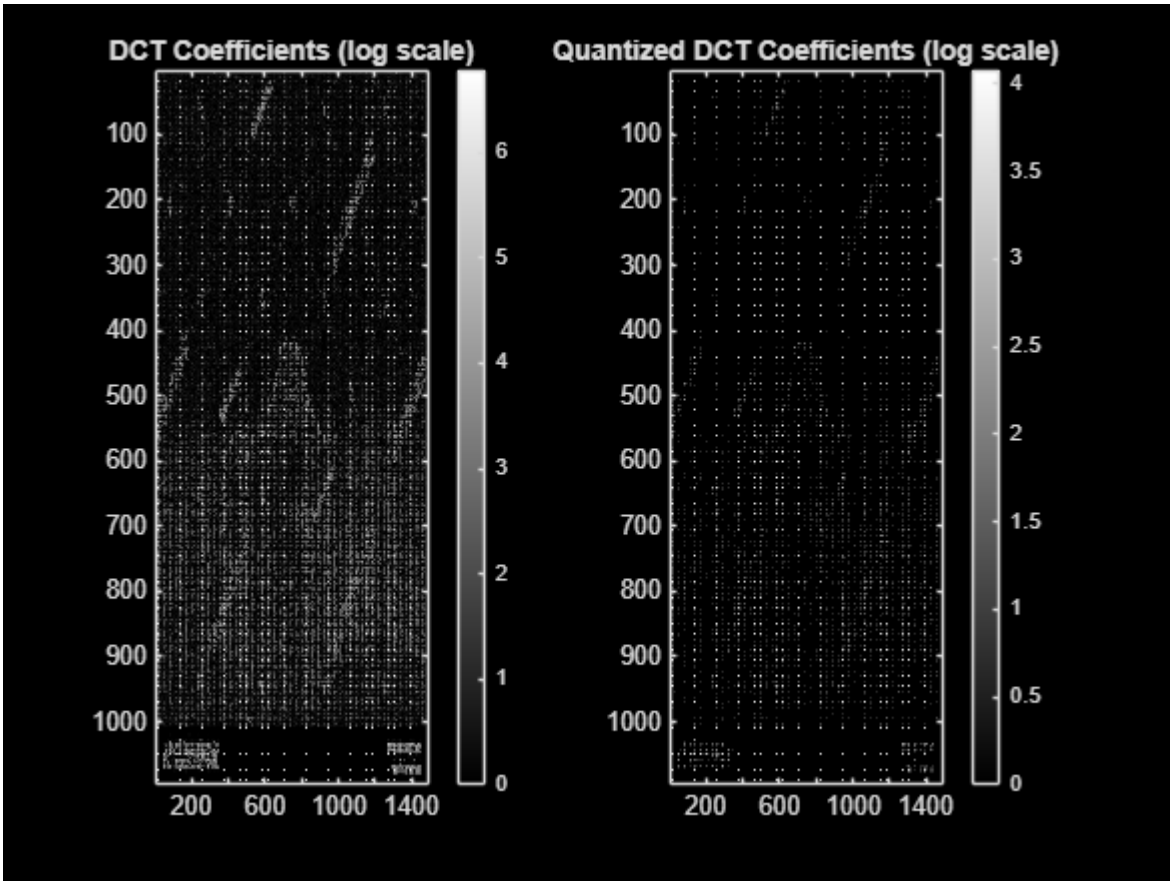
15. Reconstruct RGB Image

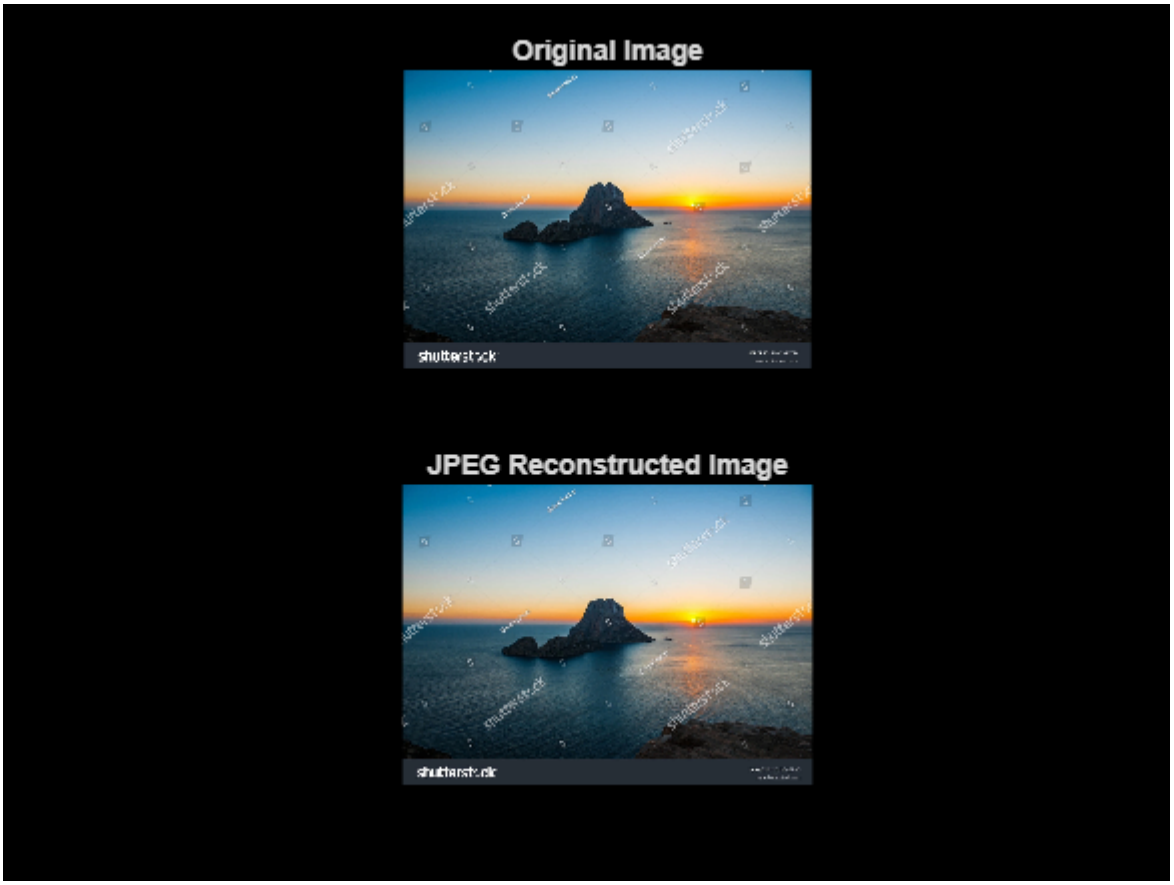
```
imgYCbCr_rec = cat(3, Y_rec, Cb_rec, Cr_rec);
imgRGB_rec   = ycbcr2rgb(imgYCbCr_rec);
```



16. Visual Comparison

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

subplot(2,1,1);
imshow(imgRGB(1:imgH8, 1:imgW8, :));
title("Original Image");

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

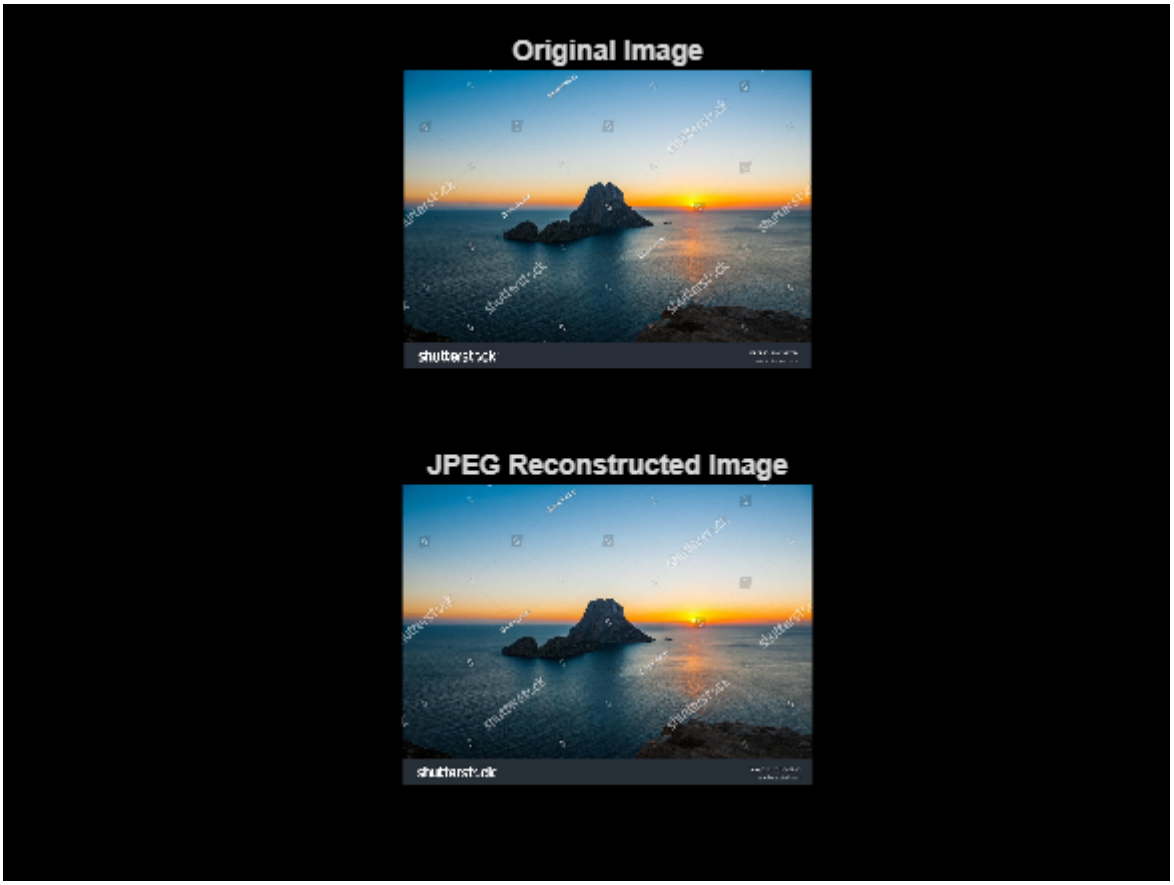

DCT Coefficients (log scale)
100
200
300
400
500
600
700
800
900
1000
200
600
1000
1400
6
5
4
3
2
1
0
Quantized DCT Coefficients (log scale)
100
200
300
400
500
600
700
800
900
1000
200
600
1000
1400
4
3.5
3
2.5
2
1.5
1
0.5
0



17. Quality Metrics

```
errMSE  = mean((double(imgRGB(1:imgH8,1:imgW8,:)) - double(imgRGB_rec)).^2,
'all');
errPSNR = 10 * log10(255^2 / errMSE);

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

```
MSE  = 12.2148
PSNR = 37.26 dB
```



Published with MATLAB® R2025b