

The background is a dark teal color. It features several decorative elements: a small blue and purple pyramid in the top left; a horizontal row of light blue chevrons in the top left; a large, tilted blue and purple pyramid in the center right; a series of parallel diagonal lines in light blue and orange in the top right; and several other smaller blue and purple pyramids scattered around the bottom and right edges.

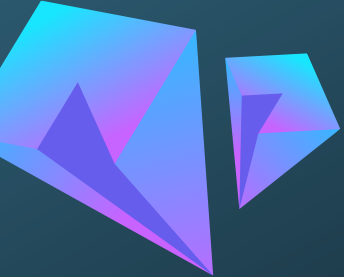

Minor Project 1

Presentation 2

Presented by
Ashwin Bansal (201A002)
Shivam Srivastava (201A016)

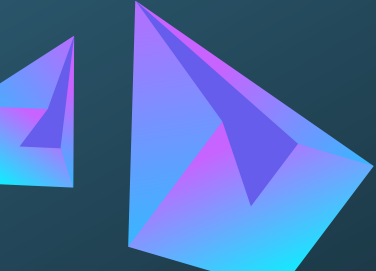



Table of Content

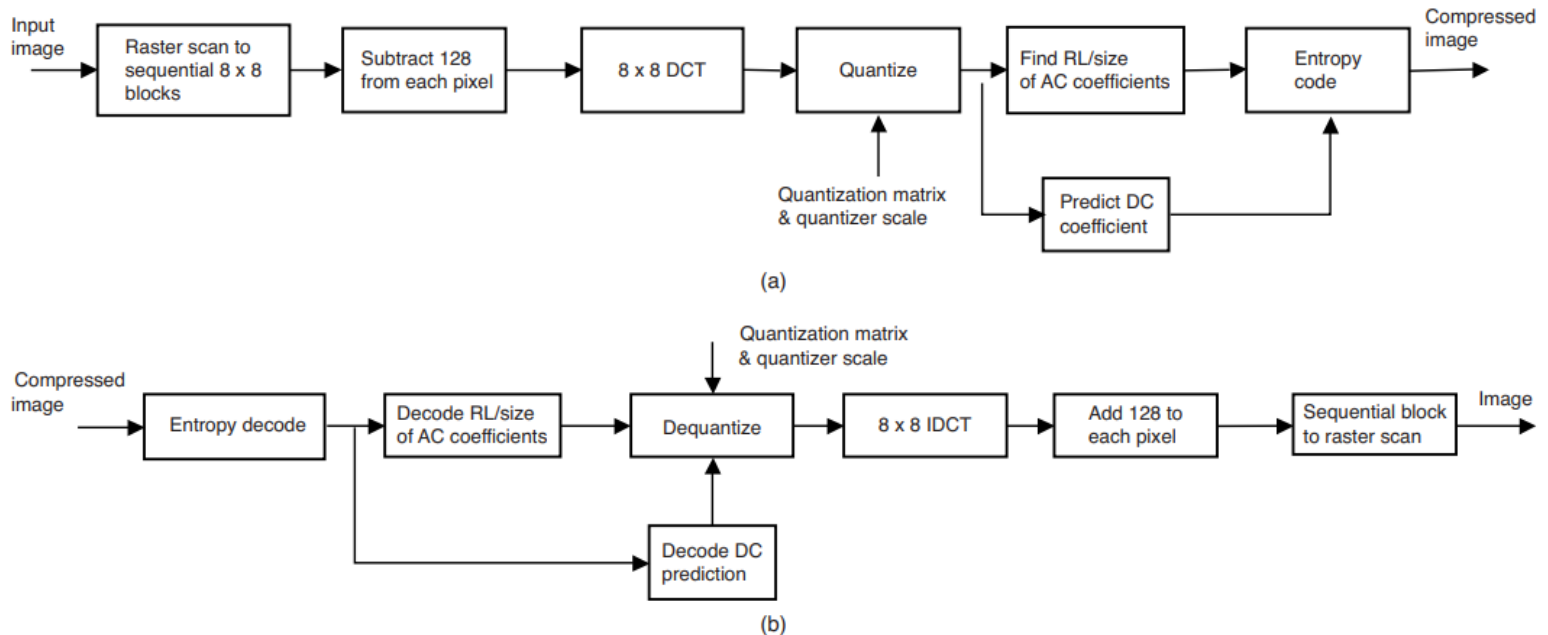
- Preview
 - Model breakdown
 - Code Explained
 - Conclusion
 - References
- 
- 



Preveiw

- Explained what is data compression.
 - Explained what is image compression
 - Presented baseline model of jpeg 2000.
 - How we are going to implement it in our project.
- 
- 

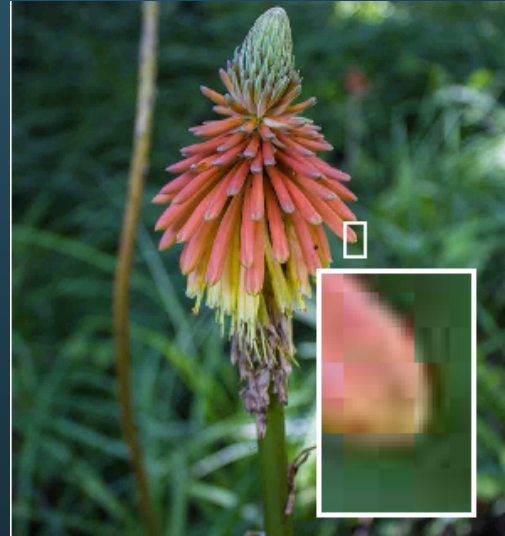
JPEG 2000 Model



We implemented each block of the above model using MatLab to apply algorithm of image compression and make changes to it.

Working

- Color space conversion
- Chrominance downscaling
- Discrete cosine transform
- Quantization
- Run length and huff man encoding



Code Explained

- Pixel collector
- Downscaling
- DCT compression
- Quantization
- Removing zeroes

```
I = imread('image.png');
ImageSize = 8*prod(size(I));
Y_d = rgb2ycbcr( I );
% Downsample:
Y_d(:,:,2) = 2*round(Y_d(:,:,2)/2);
Y_d(:,:,3) = 2*round(Y_d(:,:,3)/2);
% DCT compress:
A = zeros(size(Y_d));
B = A;
for channel = 1:3
    for j = 1:8:size(Y_d,1)-7
        for k = 1:8:size(Y_d,2)-7
            II = Y_d(j:j+7,k:k+7,channel);
            freq = chebfun.dct(chebfun.dct(II).').';
            freq = Q.*round(freq./Q);
            A(j:j+7,k:k+7,channel) = freq;
            % do the inverse at the same time:
            B(j:j+7,k:k+7,channel) = chebfun.idct(chebfun.idct(freq).').';
        end
    end
end
b = A(:);
b = b(:);
b(b==0)=[]; %remove zeros.
b = floor(255*(b-min(b))/(max(b)-min(b)));
symbols = unique(b);
prob = histcounts(b,length(symbols))/length(b);
dict = huffmandict(symbols, prob);
enco = huffmanenco(b, dict);
FinalCompressedImage = length(enco);

FinalCompressedImage/ImageSize
```




Conclusion

This algorithm is trying to make image data consume less space but having a great quality retained till end.



References

- W. K. Pratt, Digital Image Processing, 2nd edition, John Wiley, New York, 6-15, 1991
 - A. K. Jain, Fundamentals of Digital Image Processing, Prentice Hall, Englewood Cliffs, NJ, 48-54, 1989.
 - N. Ahmed, T. Natarajan, and K. R. Rao, “Discrete cosine transform,” Comp., C-23, 90–93, 1974
- 

The background features several abstract geometric shapes, including triangles and polygons, in shades of blue, purple, and pink. These shapes are scattered across the frame, with some appearing as large, prominent elements and others as smaller, more subtle details. A series of small, light blue chevron-like shapes are arranged in a horizontal line near the top left. In the bottom right corner, there are several thin, parallel diagonal lines in a light blue color.

Thank You
