Image Compression Project

Saksham Rathi, Kavya Gupta, Shravan Srinivasa Raghavan
(22B1003) (22B1053) (22B1054)

CS663: Digital Image Processing
Under Prof. Ajit Rajwade

Indian Institute of Technology Bombay
Autumn 2024

Contents

Problem Statement

The problem statement of this project has been taken from the following website:

CS663: Digital Image Processing

We have built an image compression engine along the lines of the JPEG algorithm. Along with this, we have implemented PCA (both for coloured and grayscale images) algorithm. We have thoroughly studied a tier-1 conference paper **Edge-Based Image Compression with Homogeneous Diffusion** and implemented the algorithm proposed in the paper.

All the algorithms were tested on a variety of image datasets. The results were compared and analyzed to understand the performance of the algorithms.

Basic Implementation

Here are the steps which were performed as part of the basic implementation:

- Computation of the 2D DCT coefficients of non-overlapping image patches
- Implementation of the quantization step
- Implementation of the Huffman tree
- Writing data to an appropriate file format (.bin) and plotting RRMSE vs BPP

Here is the expression of RMSE:

$$\text{RMSE} = \frac{\sqrt{\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M}(I_{\text{orig}}(i)(j) - I_{\text{recon}}(i)(j))^2}}{\sqrt{\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M}(I_{\text{orig}}(i)(j))^2}} \tag{1}$$

where I_{orig} is the original image and I_{recon} is the reconstructed image. BPP stands for the size of the image in bits divided by the number of pixels.

Quality Comparison

Here is the comparison of the reconstucted and the original image for a quality factor of 2:

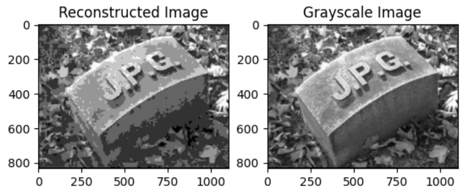

Figure: Original and Reconstructed Image

Quality Comparison

Here is the comparison of the reconstucted and the original image for a quality factor of 10:

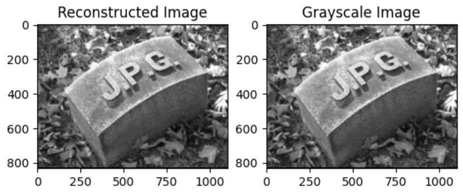

Figure: Original and Reconstructed Image

Quality Comparison

Here is the comparison of the reconstucted and the original image for a quality factor of 50:

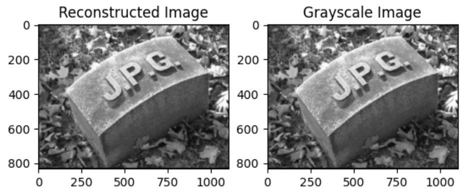

Figure: Original and Reconstructed Image

Quality Comparison

Here is the comparison of the reconstucted and the original image for a quality factor of 80:

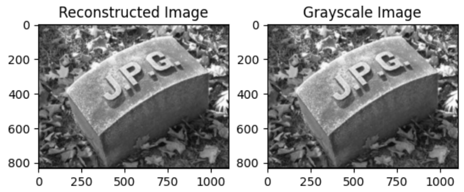

Figure: Original and Reconstructed Image

RMSE vs BPP

For the basic implementation, we have used the dataset from the miscellaneous category of the msrcorid dataset. We picked random 20 images and used 20 quality factors (in the range of 1 to 100) to plot the RMSE vs BPP graph. Here is the graph:

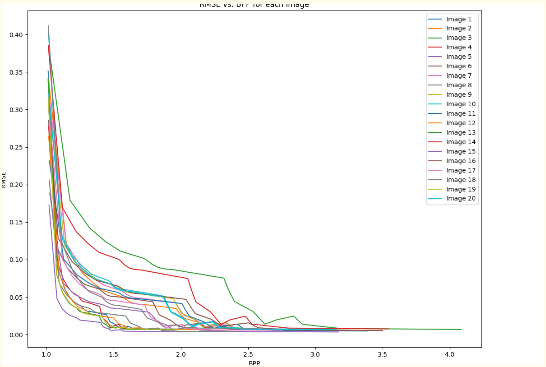

Figure: RMSE vs BPP

Comparison of Basic vs JPEG

The following plot shows how the basic algorithm performs as compared to the JPEG algorithm.

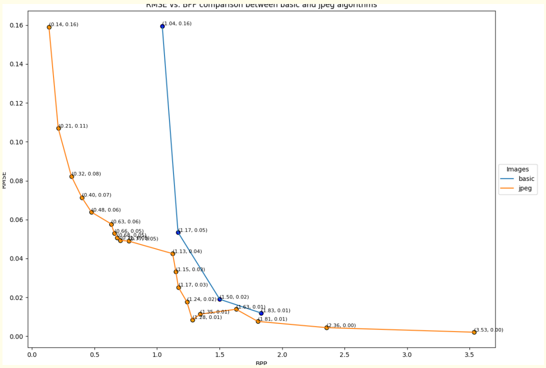

Figure: Comparison of Basic and JPEG

Run Length Encoding

The quantized DCT coefficients are arranged in a zigzag order. This pattern leaves a bunch of consecutive zeros at the end.

In runlength encoding, we replace the consecutive zeros with a pair of numbers: the number of zeros and the value of the next non-zero element. This reduces the size of the data to be stored.

RMSE vs BPP

The dataset of images from the miscellaneous category of the msrcorid dataset was used to plot the RMSE vs BPP graph. Here is the graph:

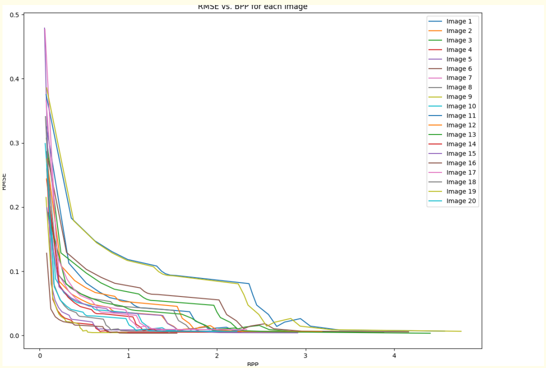

Figure: RMSE vs BPP

Comparison of Basic and Run Length Encoding

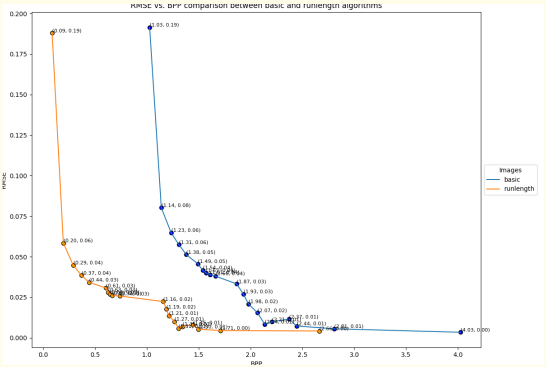

Figure: Comparison of Basic and Run Length Encoding

Comparison of Run Length Encoding and JPEG Algorithm

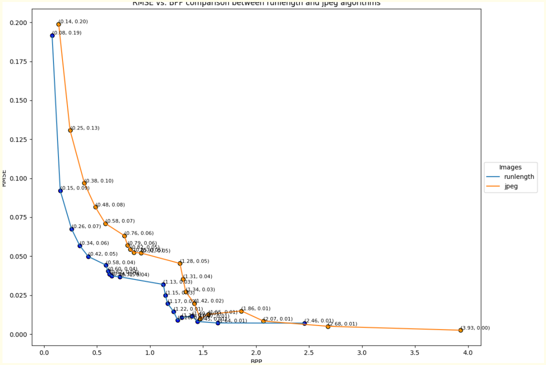

Figure: Comparison of Basic and Run Length Encoding

Edge-Based Image Compression

This paper is based on the fact that edges contain semantically important information in an image. They present a lossy compression method for cartoon-like images.

Encoding Steps

- Firstly, we need to detect edges in the image. The paper proposes Marr-Hildreth edge detector. However, based on our initial experiments, we found that the Canny edge detector works better.
- Now we store the result of edge detection using JBIG(Joint Bi-level Image Experts Group) encoding.
- Then, we encode the contour pixel values. Basically, we store the values from both sides of the edge. The extracted pixel values can be uniformly quantised to 2^q different values, where $q \in \{1, 2, \ldots, 8\}$. Another parameter d, ensures that only every d^{th} value along an edge is stored.
- The quantised and subsampled pixel value signal is then compressed by a PAQ compression method (lossless compression archiver).
- Now, we need to store the encoded data along with all the values of the hyper-parameters.

Decoding Steps

- We split our encoded file into the JBIG data and PAQ data part.
- Both parts are then decoded by using the JBIG and the PAQ method, respectively.
- We reconstruct the quantised colours obtained by the PAQ part. The pixel values between the sampled points are computed by using linear interpolation along each edge.
- The JBIG data provides a bi-level edge image of the original image size. Given the image size and the edge pixel positions, the pixel values are arranged around the edges in the same order in which they have been encoded.
- Now, we have decoded the edge locations and the pixel values surrounding the edges. We use diffusion based inpainting to fill the remaining values.

Individual Contributions

Saksham Rathi

Basic Implementation
Paper understanding and implementation
Code Modularity
Report Completion

Kavya Gupta

Basic Implementation
Coloured JPEG
Paper understanding and implementation

Shravan Srinivasa Raghavan

PCA
Paper understanding and implementation

Innovations Incorporated

-

Conclusion

References

- CS663: Image Compression Slides
- Course Textbook: “Digital Image Processing” by Rafael C. Gonzalez and Richard Woods, 3rd edition
- Osman Gokhan Sezer, Onur G. Guleryuz and Yucel Altunbasak, “Approximation and Compression With Sparse Orthonormal Transforms”, IEEE Transactions on Image Processing, 2015
- Sample Image Compression Code
- Marr-Hildreth Edge Detector