K. J. Somaiya College of Engineering, Mumbai-77
(Autonomous College Affiliated to University of Mumbai)

Batch: A-4 **Roll No.:** 16010122151

Experiment No. 10

Grade: AA / AB / BB / BC / CC / CD /DD

Signature of the Staff In-charge with date

Title: Image compression by lossless technique (Run Length Coding).

Objective: To understand image compression by lossless technique (Run Length Coding).

Expected Outcome of Experiment:

CO	Outcome
CO5	Design and develop applications based on 1-D and 2-D digital signals.

Books/ Journals/ Websites referred:

1. http://www.mathworks.com/support/
2. www.math.mtu.edu/~msgocken/intro/intro.html
3. www.mccormick.northwestern.edu/docs/efirst/matlab.pdf
4. A.Nagoor Kani “Digital Signal Processing”, 2nd Edition, TMH Education.

Pre Lab/ Prior Concepts:

Variable length code can be used to remove coding redundancy. One of the way to remove the inter pixel redundancy is run length coding. When inter pixel redundancy is removed by using run length coding the mapper transforms the input data in to usually non visual format. This operation is reversible and may or may not reduce directly the amount of data required to represent the image. Run length coding is the example of a mapping that directly results in data compression in the initial stage of overall source encoding process.
By eliminating coding and inter pixel redundancy lossless compression is achieved. Here by applying run length coding interpixel redundancy is removed and relative data redundancy is calculated as:

$$R_D=1-1/C_R$$

where C_R is the compression ratio.

ALGORITHM:
Compression:

- Read the binary (monochrome) Image.
- Write the runs of the pixel in a text file.
- Use the text file created in a step 2 and writes each run by using 8 bits in output file to create compressed image.
- Use the compress image as input
- Find Compression Ratio $C_R = N_1/N_2$, where N_1 is original image size and N_2 is compressed image data as text file
- Find Redundancy as $R_D = 1 – 1/C_R$

Decompression:

- Expand the runs to create an intensity matrix of the monochrome image
- Write decompressed image using the matrix generated in step above.

Implementation

```
L = 255;
[filename, pathname] = uigetfile({'*.bmp;*.jpg;*.png', 'Image Files
(*.bmp, *.jpg, *.png)'}, 'Select an Image');
if isequal(filename, 0)
   error('No image selected. Please select an image to continue.');
end
```

```
img = imread(fullfile(pathname, filename));
if size(img, 3) == 3
   img = rgb2gray(img);
end
figure;
contrast_images = cell(1, 3);
for k = 1:3
   r1 = input(['Enter the value for r1 for set ', num2str(k), ': ']);
   s1 = input(['Enter the value for s1 for set ', num2str(k), ': ']);
   r2 = input(['Enter the value for r2 for set ', num2str(k), ': ']);
   s2 = input(['Enter the value for s2 for set ', num2str(k), ': ']);


   a = s1 / r1;
   b = (s2 - s1) / (r2 - r1);
   c = (L - 1 - s2) / (L - 1 - r2);
   output_image = zeros(size(img));


   for i = 1:numel(img)
       r = img(i);
       if r <= r1
           s = a * r;
       elseif r > r1 && r <= r2
           s = s1 + b * (r - r1);
       else
           s = s2 + c * (r - r2);
       end
       output_image(i) = round(s);
   end

   output_image = uint8(output_image);
   contrast_images{k} = output_image;
   subplot(1, 4, k + 1);
   imshow(output_image);
   title(sprintf('Set %d: r1=%.2f, s1=%.2f, r2=%.2f, s2=%.2f, a=%.3f, 
b=%.3f, c=%.3f', k, r1, s1, r2, s2, a, b, c));
end
binaryImg = contrast_images{3};
[height, width] = size(binaryImg);
encoded = [];
for i = 1:height
   rowData = binaryImg(i, :);
   pixelVal = rowData(1);
   runLength = 0;

   for j = 1:width
       if rowData(j) == pixelVal
           runLength = runLength + 1;
       else
           encoded = [encoded; pixelVal, runLength];
           pixelVal = rowData(j);
           runLength = 1;
       end
   end
   encoded = [encoded; pixelVal, runLength];
end
fileID = fopen('encoded_runs.txt', 'w');
for i = 1:size(encoded, 1)
   fprintf(fileID, '%d %d\n', encoded(i, 1), encoded(i, 2));
end
fclose(fileID);
```

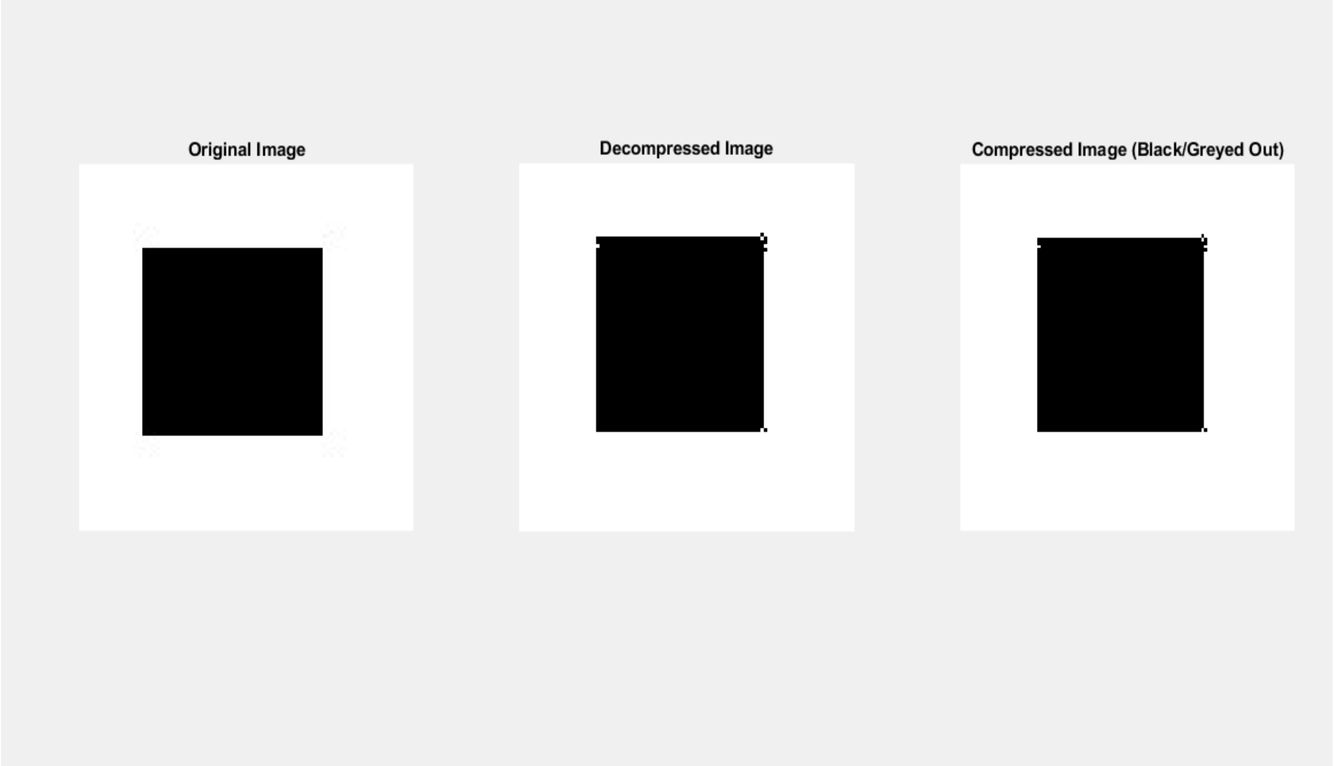
```
fileID = fopen('compressed_image.bin', 'wb');
for i = 1:size(encoded, 1)
   val = encoded(i, 1);
   len = encoded(i, 2);

   if len > 127
       error('Run length exceeds 127.');
   end
   byte = bitshift(val, 7) + len;
   fwrite(fileID, byte, 'uint8');
end
fclose(fileID);
originalSize = height * width;
compressedInfo = dir('compressed_image.bin');


if isempty(compressedInfo) || compressedInfo.bytes == 0
   error('Compression failed.');
end
compressedSize = compressedInfo.bytes * 8;
compressionRatio = originalSize / compressedSize;
redundancy = 1 - (1 / compressionRatio);
fprintf('Compression Ratio: %.2f\n', compressionRatio);
fprintf('Redundancy: %.2f\n', redundancy);
fileID = fopen('compressed_image.bin', 'rb');
if fileID == -1
   error('Compressed file not found.');
end
compressedData = fread(fileID, 'uint8');
fclose(fileID);
if isempty(compressedData)
   error('No data in compressed file.');
end
decodedImg = [];
for i = 1:length(compressedData)
   byte = compressedData(i);
   pixelVal = bitshift(byte, -7);
   runLength = bitand(byte, 127);
   decodedImg = [decodedImg, repmat(pixelVal, 1, runLength)];
end
if numel(decodedImg) ~= originalSize
   error('Decompressed data size mismatch.');
end
decodedImg = reshape(decodedImg, height, width);
subplot(1, 3, 1);
imshow(img); title('Original Image');
subplot(1, 3, 2);
imshow(decodedImg, []); title('Decompressed Image');
colormap gray;
subplot(1, 3, 3);
imshow(decodedImg, 'DisplayRange', []);
title('Compressed Image (Black/Greyed Out)');
disp('Decompression complete!');
```

OUTPUT:

```
>> expnew
Enter the value for r1 for set 1: 220
Enter the value for s1 for set 1: 230
Enter the value for r2 for set 1: 240
Enter the value for s2 for set 1: 100
Enter the value for r1 for set 2: 150
Enter the value for s1 for set 2: 250
Enter the value for r2 for set 2: 80
Enter the value for s2 for set 2: 90
Enter the value for r1 for set 3: 120
Enter the value for s1 for set 3: 160
Enter the value for r2 for set 3: 200
Enter the value for s2 for set 3: 270
Compression Ratio: 4.75
Redundancy: 0.79
Decompression complete!
>>
```



Conclusion:-

By eliminating coding and inter pixel redundancy lossless compression is achieved. Here by applying run length coding interpixel redundancy is removed and relative data redundancy is calculated as:

$$R_D = 1 - 1/C_R$$

where C_R is the compression ratio.

Post Lab Questions

1) Compare Lossy and lossless compression.

Feature	Lossy compression	Lossless compression
Definition	Reduces file size by removing some data permanently.	Reduces file size without losing original data.
Data loss	Yes, some data is permanently lost.	No, all data can be perfectly restored.
Compression Ratio	High (smaller file sizes).	Lower compared to lossy.
Quality	Reduced quality due to loss of information.	Maintains original quality.
Examples	JPEG, MP3, MP4, H.264	PNG, GIF, TIFF, ZIP
Use Cases	Images, videos, audio (where small size is more important than perfect accuracy).	Text, medical images, legal documents (where accuracy is critical).
Reversibility	Irreversible (original cannot be recovered).	Reversible (can restore original file).