The Run Length Encoding for RGB Images
Pratishtha Gupta\textsuperscript{1}, Varsha Bansal\textsuperscript{2}
Computer Science, Banasthali University, Jaipur, Rajasthan, India\textsuperscript{1}
Computer Science, Banasthali University, Jaipur, Rajasthan, India\textsuperscript{2}

Abstract: This document presents the basically the implementation of Run Length Encoding that is one of the lossless image compression technique. This paper gives the implementation of Run length encoding compression algorithm which is capably well-matched for RGB images data. Here considered painted and natural images for the examination of implemented scheme. By this technique image can be compressed and occupy short space in memory, and improve the performance and result of the system. There for that RUN LENGTH ENCODING split big sequences of runs that totally affects compression ratio into small sequences of runs without degrading the quality of image.

Keyword: RLE (run length encoding), image compression, R (Red), G (Green), B (blue).

Introduction of Image Compression

Image compression and processing is currently a well-known context for computer science countryside. Essentially, image compression is the processes of images that change the images into small codeword without any loss of important information. The image compression process provides the most favorable for consumption for storage, Nagarajan A. et al, [2]. The less size of images allows more images to be stored in a less memory space or disk drive. It also uses short time slot for images to be sent over the network or downloaded from web pages. Gupta G. et al [1]. In other words, the basic enthusiasm of image compression is using short amount of data to represent the innovative image without compromising with information. And reduce the size of image for decrease the transmission time.

Introduction of Run length encoding

RLE (Run-length encoding) is a very popular, simple and easy concept of data compression, in which the count of rate of same data is stored as a single data value and single count. This is most useful for the images that contains many such runs, huge number of same data value: for example, a simple RGB image such as same color occur many time. It is less helpful with RGB images that don't have many runs or same value data as it could to a great extent increase the files size. The Run length encoding technique performs a lossless compression of input images that is based on sequences of identical values (runs), Amin A., et al. [4].

Basic Working of Run Length Encoding

Here, let's take the example of image and perform the run length encoding. There will be too much long runs of white pixels, and short runs of black pixels. Here considered take a single scan line or row of image with B representing a black pixel and W representing white pixel.
RGB color combination. Image read from first pixel of image and starts compression.

fig 1. RGB image with pixel value.

- Here each cell of matrix represents the pixel.
- This algorithm scans image one row at a time. This technique shows result as above manner.
- For example the result is shows as a ;"2R 2G 3B" that is represents 2R means 2 pixels of red color.
  2G means 2 pixels of green color.
  3B means 3 pixels of blue color. And so on.

The basic steps of proposed algorithm of Run Length Encoding are as follow

Compression

**Step 1:** Firstly, Input the colored source image file.

**Step 2:** Find out the size of source image by following statement

\[
\text{[row, col, dim]} = \text{size}(I);
\]

**Step 3:** Read pixel values from first pixel of source image by help of this statement \(X = \text{impixel}(I, i, j)\);

Here \(i=\text{row};\)

\(j=\text{columns};\)

\(I=\text{Image};\)

**Step 4:** Read next pixel value, if current pixel is end of the image then exit from loop otherwise

(i).If next pixel value is same from previous than \(\text{Count} = \text{count} + 1;\)

(ii). Else if mismatch in value of next pixel as the previous than save as the new value of pixel in array.

**Step 5:** Read and count all the value of pixel.

**Step 6:** Go to step 4 until all pixel read.

**Step 7:** Display the result array with intensity value.

Decompression

**Step 1:** Firstly Read compressed array which store the intensity value and attain the image size.

**Step 2:** Generate the vacant array for reconstruction of compressed image.

New array ( ; ; 1:3);

**Step 3:** For reconstructing compressed image,

a. Construct the \(i^{th}\) row of compressed image with putting run length value in reconstruct array from compressed array.

b. Then construct \(i+1^{th}\) row then next row and so on.

**Step 4:** Step 3 is repeated until reconstruct array fill by value of compressed array.

**Step 5:** Reconstruct array, store as a decompressed image file.

Test Result

Input: This function takes source colored images.

Output: This function provides the compressed image file.

**Figure 2. bag.jpg**

Before compression of Bag.jpg image
Before compression for Thumbnail.jpg image
Size row =
699
Col = 697 = 11692872 bits
11692872/8 = 1461609 bytes
After compression of above image
Compression percentage is 20.29%

### Examination of Run Length Encoding Technique for Colored Image

According to result of Run Length Encoding scheme, This technique of compression works efficiently where large areas of similar pixel value takes place in image data. In the image data ‘Bag.jpg’, RLE compression technique shows much more compression percentage that is 82.5% because here is large number of pixels have same value in it. Now, consider the ‘Thumbnail.jpg’ image, this image slightly compressed than ‘Bag.jpg’ image data because there are many colors present in it.

### Conclusion

This Document provide a working of Run Length Encoding compression technique (RLE) of RGB images data. It is the unambiguous from of algorithm that remove the pixel value from image data. Compression is very much useful and important part of Image Processing filed. Fundamentally these methodology will discover complete use in Medical image, GIS images (geographical information system), because these type of image has large area of identical pixel pattern.

## Table 1

<table>
<thead>
<tr>
<th>Name</th>
<th>Row (i)</th>
<th>Col (j)</th>
<th>RGB bits</th>
<th>Total size before compression</th>
<th>Size in bytes</th>
<th>Array size in row</th>
<th>Array size in col</th>
<th>Compression calculation</th>
<th>Compression Percentages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bag.jpg</td>
<td>235</td>
<td>300</td>
<td>16*3=48</td>
<td>1692000 bits</td>
<td>360990 bytes</td>
<td>1397184</td>
<td>96</td>
<td>1397184/1692000*100</td>
<td>82.5%</td>
</tr>
<tr>
<td>Thumbnail.jpg</td>
<td>699</td>
<td>697</td>
<td>16*3=48</td>
<td>11692872 bits</td>
<td>1461609 bytes</td>
<td>2373408</td>
<td>96</td>
<td>2373408/11692872*100</td>
<td>20.29%</td>
</tr>
</tbody>
</table>

IN RUN LENGTH ENCODING SCHEME” International Journal of Engineering Trends and Technology - July to Aug Issue 2011

## References


