**CPS621 Winter2022**
**Lab06 Report**
**Name: Tusaif Azmat, Student#: 500660278.**

Work to Do

## Part 2 – Download a high-quality image, like desktop wallpaper images. It is better to use an uncompressed image.

**Explanation:**

I downloaded a forest image. The initial resolution was 4K; I used Photoshop to reduce the resolution to 1080P. The image's file format is JPG.

Link: https://wallpaperaccess.com/full/773426.jpg

## Part 3 – Use Matlab imfinfo() function to check the resolution, bit depth, and coding method of the file.

```
% CPS621 Lab 6

% 3. use imfinfo() to analyze the image
Image_Info = imfinfo('forest.jpg');
fprintf("Information about 'forest.jpg'");
fprintf("Image's Width is %d pixels and Height is %d pixels\n", Image_Info.Width,
Image_Info.Height);
fprintf("Image's Bit Depth is %d bits\n", Image_Info.BitDepth);
fprintf("Image's Coding Method is %s\n", Image_Info.CodingMethod);
fprintf("Image's File Size is %d bytes or %0.2f Megabytes\n", Image_Info.FileSize,
((Image_Info.FileSize*(1/1048576))));
fprintf("Image's File Format is %s\n", Image_Info.Format);
```

**Figure 1:** Source code for Part 3. [3]

```
>> Lab6Script
Information about 'forest.jpg'Image's Width is 1920 pixels and Height is 1080 pixels
Image's Bit Depth is 24 bits
Image's Coding Method is Huffman
Image's File Size is 2507628 bytes or 2.39 Megabytes
Image's File Format is jpg
```

**Figure 2:** Information about the chosen image. [3]

**Explanation:**

Using the imfinfo() function. WE found out that the image's resolution is 1920 x 1080 pixels. The Width is 1920 pixels and Height is 1080 pixels. The image's coding method is Huffman. The file size is 2.39 Megabytes or 2507628 bytes. The file format is jpg. The Bit depth is 24 bits.

**Part 3 – Then calculate the storage space need for this image without compression.**

```
% Calculate the storage space need for this image without compression
% size = [((Vertical pixels x Horizontal pixels) x (bit depth)) / 8] bytes
Calculate_Image_Size = ((Image_Info.Width * Image_Info.Height) * (Image_Info.BitDepth))
/ 8;
fprintf("Storage space needed for this image without compression is %d bytes or %0.2f
Megabytes\n", Calculate_Image_Size, ((Calculate_Image_Size*(1/1048576))));
clear;
```

**Figure 3:** I calculated the storage space needed for this image without compression using Matlab script.

```
Storage space needed for this image without compression is 6220800 bytes or 5.93 Megabytes
```

**Figure 4:** The storage space needed was calculated to be around 5.93 Megabytes or 6220800 bytes. [4]

**Explanation:**

We can calculate the storage space needed by using the below formula: Image Size = ((Width x Height) x (Bit depth)) / ( 8 ) bytes The above formula gives the image size in bytes. To convert it into Megabytes, just divide it by 1048576. Image Size = ((1920 x 1080) x 24) / 8 = 6220800 bytes or 5.93 Megabytes. And so, the storage space needed for the image without compression is 5.93 Megabytes or 6220800 byes.

**Part 4 – Convert the image to an uncompressed TIFF image using Matlab (use imwrite() function).**

```
% 4. Convert to an uncompressed Tiff image
imwrite(imread('forest.jpg'), 'forest_uncompress.tif');
Image_Info = imfinfo('forest_uncompress.tif');
fprintf("Information about 'forest_uncompress.tif'\n");
fprintf("Image's Width is %d pixels and Height is %d pixels\n", Image_Info.Width,
Image_Info.Height);
fprintf("Image's Bit Depth is %d bits\n", Image_Info.BitDepth);
fprintf("Image's File Size is %d bytes or %0.2f Megabytes\n", Image_Info.FileSize,
((Image_Info.FileSize*(1/1048576))));
fprintf("Image's File Format is %s\n", Image_Info.Format);
clear;
```

**Figure 5:** The source code for Part 4. [1] [2] [3]

```
Information about 'forest_uncompress.tif'
Image's Width is 1920 pixels and Height is 1080 pixels
Image's Bit Depth is 24 bits
Image's File Size is 6268448 bytes or 5.98 Megabytes
Image's File Format is tif
```

**Figure 6:** Information about the uncompress image generated from source code in Figure 5.

**Explanation:**

I used the imwrite function to save the .JPG image to a .TIF uncompress image. Then, I used infinfo() function to get info on the uncompress TIF image. The Resolution is unchanged; it is 1920 x 1080 pixels. The bit depth remained unchanged, it is 24 bits. The file format is tif and file size is 6268448 bytes or 5.98 Megabytes.

### Part 4 – Compare the size of the file with your calculation in step 3.

```
Storage space needed for this image without compression is 6220800 bytes or 5.93 Megabytes
```

**Figure 7:** Calculated image size for an uncompressed image of the original 'forest.jpg' image.

```
Information about 'forest_uncompress.tif'
Image's Width is 1920 pixels and Height is 1080 pixels
Image's Bit Depth is 24 bits
Image's File Size is 6268448 bytes or 5.98 Megabytes
Image's File Format is tif
```

**Figure 8:** Info on uncompressed TIF image 'forest_uncompress.tif'. [4]

**Explanation:**

I calculated the storage space required for an uncompressed image to be around 622080 bytes or 5.93 Megabytes. Using the imfinfo() function I found out that the image size of the uncompressed TIF image is 6268448 bytes or 5.98 Megabytes. So, my calculated image size value of 5.93 Megabytes is very close to 5.96 Megabytes. They don't match but both of those values are very similar to each other. They reason they don't match could be due to the .TIF file format, it might be reducing the file size without losing image quality.

### Part 5 – Save the image in JPEG format using 10 equally distributed quality values from 10 to 100 (use imwrite() function).

```
%5. Save image in JPEG using 10 quality values from 10 to 100
Image = imread('forest_uncompress.tif');
imwrite(Image, 'forest_quality_100.jpg', 'Quality', 100);
imwrite(Image, 'forest_quality_90.jpg', 'Quality', 90);
imwrite(Image, 'forest_quality_80.jpg', 'Quality', 80);
imwrite(Image, 'forest_quality_70.jpg', 'Quality', 70);
imwrite(Image, 'forest_quality_60.jpg', 'Quality', 60);
imwrite(Image, 'forest_quality_50.jpg', 'Quality', 50);
imwrite(Image, 'forest_quality_40.jpg', 'Quality', 40);
imwrite(Image, 'forest_quality_30.jpg', 'Quality', 30);
imwrite(Image, 'forest_quality_20.jpg', 'Quality', 20);
imwrite(Image, 'forest_quality_10.jpg', 'Quality', 10);
```

**Figure 9:** Source code for Part 5. 10 images were saved from the original image; each of them was save using a different quality factor from the original image.

**Explanation:**

I used the imwrite function to save the original image to a reduced quality image. Initially, I used the imread function to read the original image first. I used the 'Quality' factor of the imwrite function to reduce the quality from 100 to specify values form 100 to 10.

## Part 6 – Compare the quality of the compressed files visually, then quantitatively using the structural similarity (SSIM) index (use ssim() function).

```
% 6. Compare the quality of the compressed files
I_100 = imread('forest_quality_100.jpg');
[ssimval_100, ssimmap_100] = ssim(I_100,Image);

I_90 = imread('forest_quality_90.jpg');
[ssimval_90, ssimmap_90] = ssim(I_90,Image);

I_80 = imread('forest_quality_80.jpg');
[ssimval_80, ssimmap_80] = ssim(I_80,Image);

I_70 = imread('forest_quality_70.jpg');
[ssimval_70, ssimmap_70] = ssim(I_70,Image);

I_60 = imread('forest_quality_60.jpg');
[ssimval_60, ssimmap_60] = ssim(I_60,Image);

I_50 = imread('forest_quality_50.jpg');
[ssimval_50, ssimmap_50] = ssim(I_50,Image);

I_40 = imread('forest_quality_40.jpg');
[ssimval_40, ssimmap_40] = ssim(I_40,Image);

I_30 = imread('forest_quality_30.jpg');
[ssimval_30, ssimmap_30] = ssim(I_30,Image);
I_20 = imread('forest_quality_20.jpg');
[ssimval_20, ssimmap_20] = ssim(I_20,Image);

I_10 = imread('forest_quality_10.jpg');
[ssimval_10, ssimmap_10] = ssim(I_10,Image);

fprintf("Image Quality 100, SSIM value is %f\n", ssimval_100);
fprintf("Image Quality 90, SSIM value is %f\n", ssimval_90);
fprintf("Image Quality 80, SSIM value is %f\n", ssimval_80);
fprintf("Image Quality 70, SSIM value is %f\n", ssimval_70);
fprintf("Image Quality 60, SSIM value is %f\n", ssimval_60);
fprintf("Image Quality 50, SSIM value is %f\n", ssimval_50);
fprintf("Image Quality 40, SSIM value is %f\n", ssimval_40);
fprintf("Image Quality 30, SSIM value is %f\n", ssimval_30);
fprintf("Image Quality 20, SSIM value is %f\n", ssimval_20);
fprintf("Image Quality 10, SSIM value is %f\n", ssimval_10);

montage({ssimmap_100,ssimmap_90,ssimmap_80,ssimmap_70,ssimmap_60,ssimmap_50,ssimmap_40,
ssimmap_30,ssimmap_20,ssimmap_10});
title("SSIM Map for Images, Quality from 100 to 10 from Left to Right, Top to
Bottom.");
```

**Figure 9:** Source code for Part 6. I used imread and ssim functions to get ssim value and Map images.

```
Image Quality 100, SSIM value is 0.990714
Image Quality 90, SSIM value is 0.972134
Image Quality 80, SSIM value is 0.953330
Image Quality 70, SSIM value is 0.939494
Image Quality 60, SSIM value is 0.928092
Image Quality 50, SSIM value is 0.918177
Image Quality 40, SSIM value is 0.907254
Image Quality 30, SSIM value is 0.891587
Image Quality 20, SSIM value is 0.866207
Image Quality 10, SSIM value is 0.808711
```

**Figure 10:** The SSIM values for all the 10 images generated using a different quality factor. This is the output from Figure 9 source code.

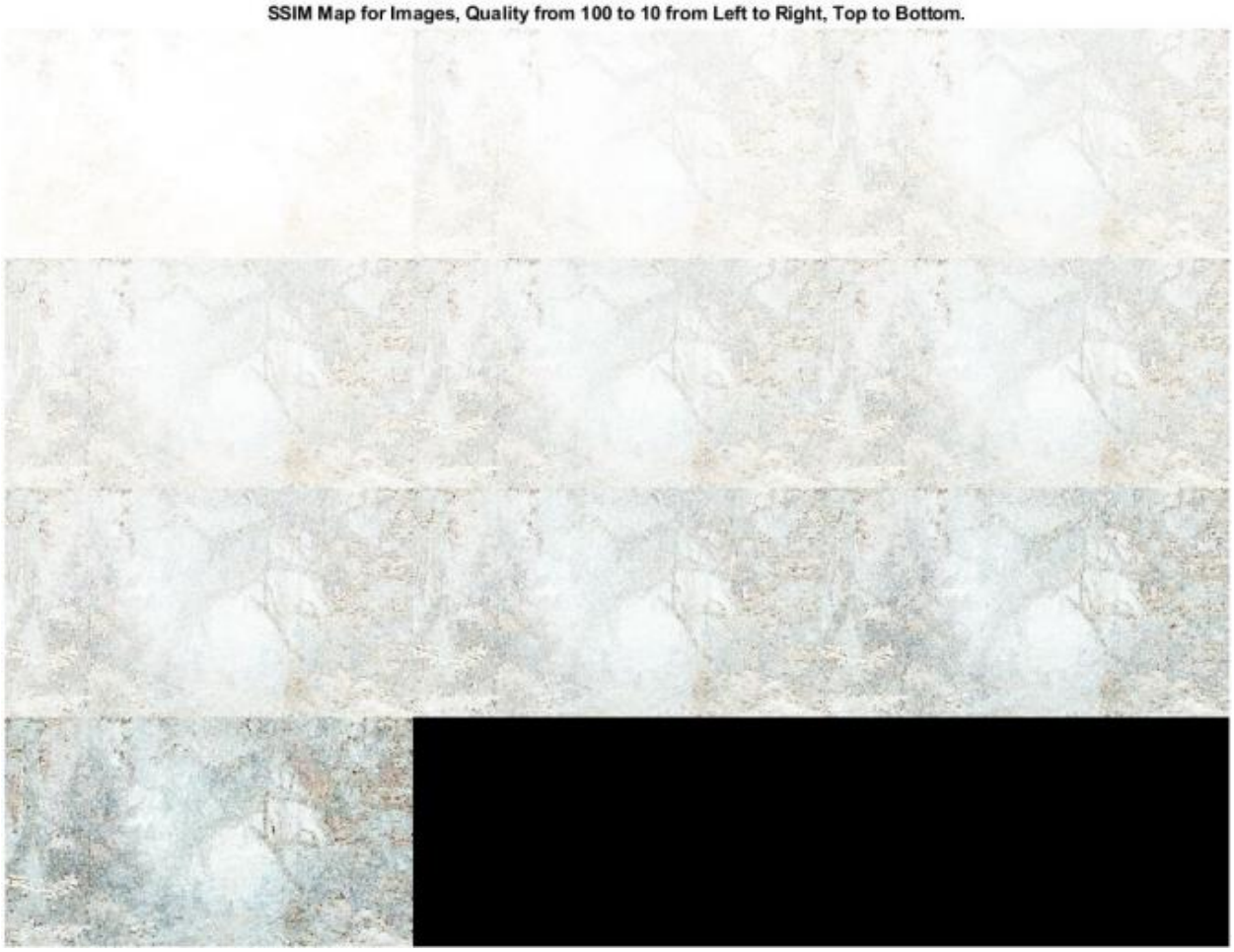


**Figure 10:** The SSIM Map images for the image generated with different quality factor. The top left image has a quality factor of 100. From left to right, top to bottom, the quality decreases form 100 to 10. The last image at the bottom has a quality of 10.

**Explanation:**

By looking at the compressed images that those image with quality of 10 looks the worst while the image with quality of 90 looks same as the original image. As the quality factor decreases from 100 to

10, the image quality also decreases. The SSIM values also decrease from 0.99 for Quality of 100 to 0.80 for quality of 10. The SSIM Map also shows this,

## Report – Questions & Answers

1) A summary of JPEG compression algorithm, including DCT, quantization, and entropy coding, etc.

   JPEG is an image compression standard. Developed by 'Joint Photographic Experts Group', is widely used on the internet to share and display images. Since it is a lossy file format, converting an image to JPEG will result in that said image to lose some quality. The main advantage of JPED is reducing file size. JPEG employs a lossy image compression 'transform coding method' that uses DCT (Discrete Cosine Transform). DCT converts a signal or an image from spatial time domain to the frequency domain. The DCT is used for JPED compression to reduce high-frequency contents and change it into a bit string.

   The DCT method for JPEG looks for three things:

   1. "Spatial redundancy", intensity values don't change widely within small area.
   2. Humans don't notice loss of high spatial frequency components, so the DCT method reduces high spatial frequency contents.
   3. Visual acuity, accuracy in looking at closely spaced lines. It is much greater for gray than for color.

   Quantization is a step in JPEG compression method; its aim is to reduce total number of bits needed for a compressed image. It divides each entry in the frequency spce block by a value then rounds it.
   Entropy coding is the last step in JPED compression; it is used to reduce the file size even greater. Huffman coding is used for this step.
2) The details of your computation in step 3. Is it equal to the real size of the TIFF file? Make a brief discussion on your finding.
   I calculated the size of an uncompressed image using the below formula: Image Size = ((Width x Height ) x (Bit depth)) / ( 8 ) bytes
   The calculated image size was around 5.93 Megabytes or 6220800 bytes. Image Size = ((1920 x 1080) x 24) / 8 = 6220800 bytes or 5.93 Megabytes.
   The actual image size for TIF file format was 6268448 bytes or 5.98 Megabytes. The reason the calculated file size and the actual file size don't match is probably because the TIF file format might contain additional data that we can't really calculate using the above formula. The TIF file format might require some additional data related to the file format.
3) A table showing the sizes of the original image, the TIFF image, and the 10 compressed JPEG images.

| Image Name | Image File Format | Image File Size |
|---|---|---|
| Original Image 'forest.jpg' | .jpg | 2.39 MB (2,507,628 bytes) |
| TIF image 'forest_uncompress.tif' | .tif | 5.97 MB (6,268,448 bytes) |
| forest_quality_100.jpg | .jpg | 2.38 MB (2,503,894 bytes) |
| forest_quality_90.jpg | .jpg | 972 KB (995,419 bytes) |
| forest_quality_80.jpg | .jpg | 661 KB (677,675 bytes) |
| forest_quality_70.jpg | .jpg | 520 KB (533,467 bytes) |
| forest_quality_60.jpg | .jpg | 435 KB (446,203 bytes) |
| forest_quality_50.jpg | .jpg | 378 KB (387,717 bytes) |
| forest_quality_40.jpg | .jpg | 326 KB (333,994 bytes) |
| forest_quality_30.jpg | .jpg | 270 KB (276,986 bytes) |
| forest_quality_20.jpg | .jpg | 203 KB (208,108 bytes) |
| forest_quality_10.jpg | .jpg | 121 KB (124,182 bytes) |

**Table 1:** Shows file size for the images generated/used in this lab.

4) Show four images with quality factors 10, 20, 50, and 100, respectively.

**Figure 11:** Image with quality factor of 10. You can see the bad quality in the center.

**Figure 12:** Image with quality factor of 20. You can see the bad quality in the center.

**Figure 13:** Image with quality factor of 50.

**Figure 13:** Image with quality factor of 100.

5) Do you perceive a visual difference among the 10 compressed JPEG files? Make a brief analysis of your finding.

Yes, I can see the image quality among the 10 compressed JPEG files. The image with a quality factor of 10 has the worst image quality. I can't really tell a difference between the images with quality factors of 90 and 100. If you zoom in really close, you can tell a difference between the images with quality factors from 100 to 10.

6) Show the curve of ssim quality scores of the 10 compressed JPEG images (see the link below for reference). Make a brief analysis of the result.

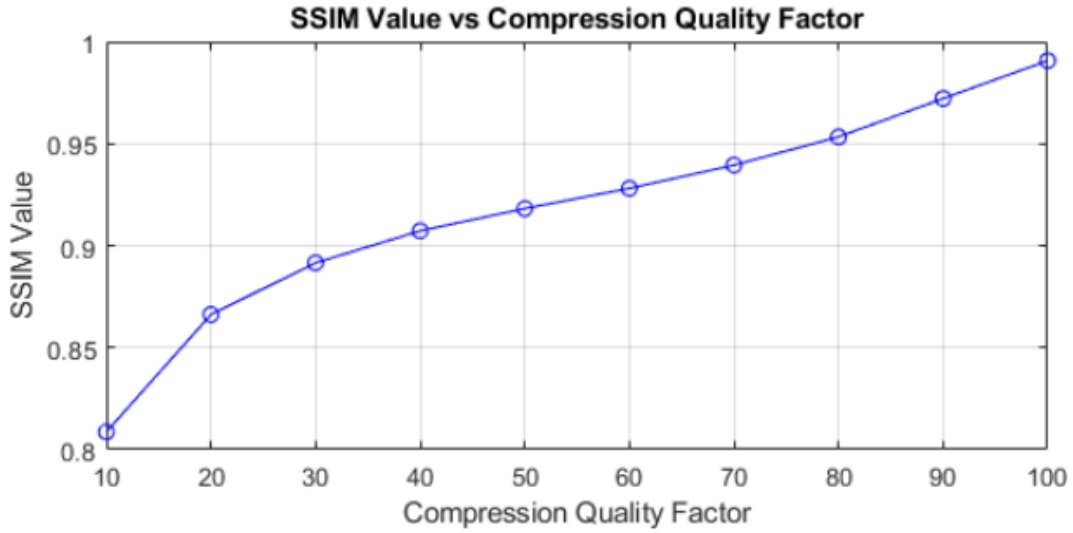


**Figure 14:** Graph shows SSIM value vs Compression Quality factor.

```
I = imread('forest.jpg');
ssimValues = zeros(1,10);
qualityFactor = 10:10:100;
for i = 1:10
    imwrite(I,'compressedImage.jpg','jpg','quality',qualityFactor(i));
    ssimValues(i) = ssim(imread('compressedImage.jpg'),I);
end
plot(qualityFactor,ssimValues,'b-o');
xlabel('Compression Quality Factor');
ylabel('SSIM Value');
title('SSIM Value vs Compression Quality Factor');
grid on;
```

**Figure 15:** Source code for Figure 14 Graph [1]

### Explanation:

As the compression quality factor increases, the SSIM value increases. For a quality factor of 100, the SSIM value should be 1.

### References


[1] Compare Image Quality at Various Compression Levels - MATLAB & Simulink. (n.d.). Matlab. Retrieved April 09, 2022, from https://www.mathworks.com/help/images/compare-image-quality-at-variouscompression-levels.html

[2] Information about graphics file - MATLAB imfinfo. (n.d.). Matlab. Retrieved April 09, 2022, from https://www.mathworks.com/help/matlab/ref/imfinfo.html

[3] Write image to graphics file - MATLAB imwrite. (n.d.). Matlab. Retrieved April 09, 2022, from https://www.mathworks.com/help/matlab/ref/imwrite.html

[4] Read image from graphics file - MATLAB imread. (n.d.). Matlab. Retrieved April 09, 2022, from https://www.mathworks.com/help/matlab/ref/imread.html