JPEG IMAGE COMPRESSION IN MATLAB

ABSTRACT:

Original image generated by a camera sensor is a very large store, and therefore is not efficient. The present scenario requires the use of efficient image compression techniques to fulfill faster image transfers and low storage space requirements. JPEG is often applied for the processing and storage of full-color images with realistic elements and brightness and color transitions.Also this format is used for storing and transmitting graphical digital content (photos, scanned copies of digitized pictures).

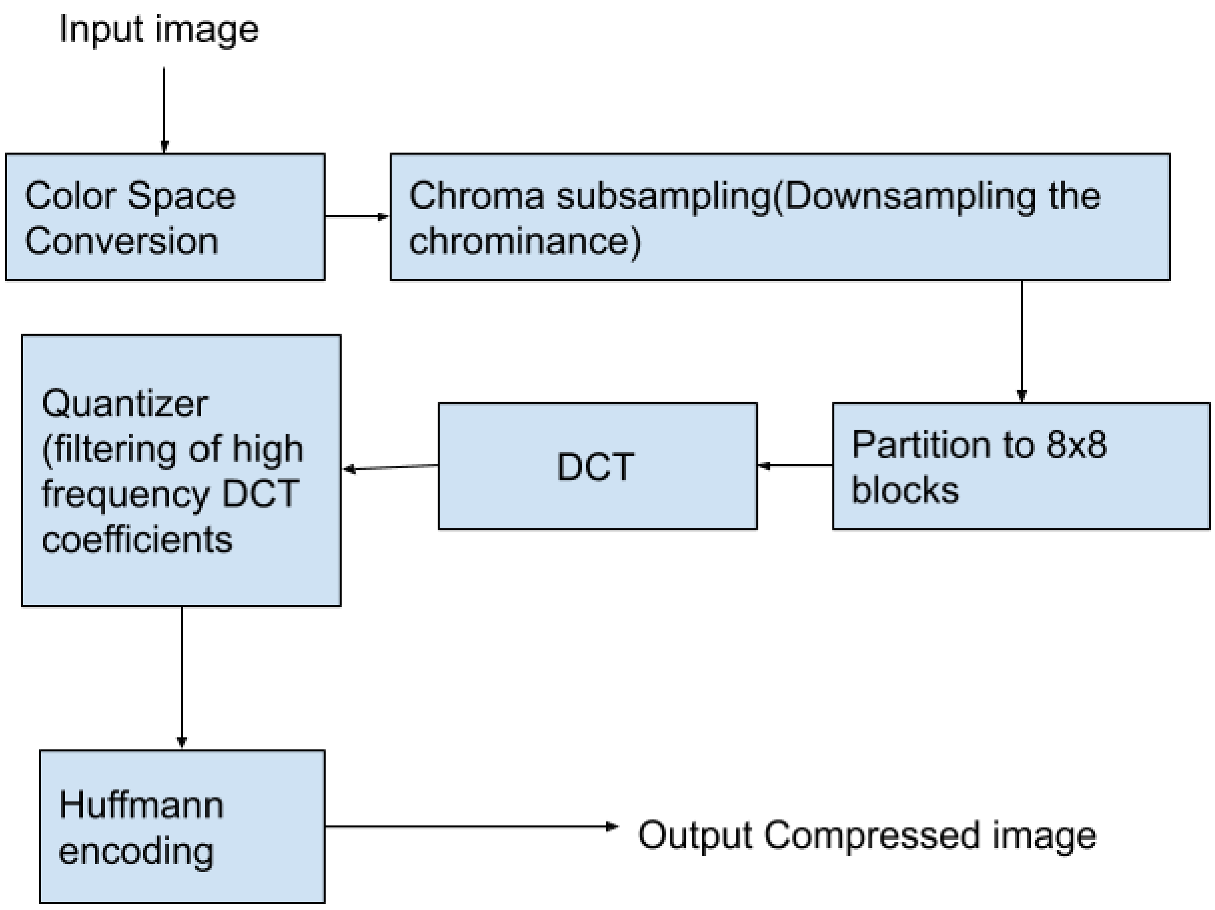
It is the most convenient for transmission of compressed images on the Internet, because it takes less space in comparison to other formats. However, JPEG is ideal for home photo storage, photo transmission through the Internet or posting on the site.

INTRODUCTION:

JPEG is an image compression standard to store image in compressed format. It represents the Joint Photographic Experts Group. Excellent quality of JPEG is that it achieves high compression ratio and quality is with almost no loss. JPEG format is very popular, and is used in a large-sized image switching a plurality of devices such as digital cameras, and is selected in the bandwidth-limited environments, such as the format of the Internet. JPEG algorithm is best suited for photos and realistic scenes with smooth changes in tone and color painting. JPEG is not suitable for use with many edges and sharp changes, since this may result in many image artifacts in the resulting image. In these cases, it is best to use a lossless format such as PNG, TIFF or GIF. For this reason, JPEG is not in use for medical and scientific applications, where the image needs to be exact and slight error results into no reproduction of captured data.

JPEG image may accept further losses, if it is frequently edited, and then save it. The operation of decompression and recompression can further reduce image quality. To solve this problem, the image should be edited and saved in a lossless format, only converted to JPEG format, just before the final transport to the required media. This ensures minimal loss due to frequent savings. Saved as JPEG image files usually have extensions such as .jpg, jpeg, or .jpeg .

BLOCK DIAGRAM:


Input image
Color Space Conversion
Chroma subsampling(Downsampling the chrominance)
Partition to 8x8 blocks
DCT
Quantizer (filtering of high frequency DCT coefficients
Huffmann encoding
Output Compressed image

THEORY AND IMPLEMENTATION:

Color Space Conversion:

On this step the colors of the RGB (Red, Green, Blue) image are converted to YCbCr (Luminance, Chrominance Blue, Chrominance Red). During this conversion we take three standard channels (RGB) and convert them into a different representation which is based on a luminance (brightness) channel and two opposing color channels. These three channels are typically less interdependent then the RGB channels. That allows us to store them with different resolution. If the image is CMYK or grayscale, it is processed without any color conversion.

Chroma Subsampling:

It is widely known that we are much more sensitive to changes in luminance than in chrominance. It means that we may neglect larger changes in the chrominance without affecting our perception of the image. Therefore, we can store the color information at lower resolution than the luminance information. This method can be applied only to images in YCbCr color space. CMYK images use all the channels to store color information. Therefore each color channel is compressed and quantized with the similar quality. Grayscale images have no color information and do not need being converted.

Segmentation into Blocks:

The image is divided into 8 by 8 blocks of pixels. Since each block is processed without reference to the others, further steps will be described relatively to a single block.

Discrete Cosine Transform:

Each of the color component data (e.g. Y, Cb, Cr for YCbCr or C, M, Y, K for CMYK) undergoes the Discrete Cosine Transform (DCT). The DCT is similar to the Fourier Transform in the sense that it produces a kind of a spatial frequency spectrum.

Quantization:

The amplitudes of the frequency components are quantized. The human vision is much more sensitive to small variations in color or brightness over large areas (low-frequency components) than to variations which occur on every pixel (high-frequency components). Therefore, the high-frequency components are stored with a lower accuracy than the low-frequency ones. The quality setting of the encoder affects the resolution of each frequency component. Higher values of quality setting correspond to higher quality images and larger file sizes.

Huffman encoding:

Huffman coding is a method that takes symbols (e.g. bytes, DCT coefficients, etc.) and encodes them with variable length codes that are assigned according to statistical probabilities. A frequently-used symbol will be encoded with a code that takes up only a couple bits, while symbols that are rarely used are represented by symbols that take more bits to encode. Here the resulting data for all 8 by 8 blocks of pixels is compressed with a lossless algorithm, a variant of Huffman encoding.

Mean Square Error and Peak SNR:
Comparing restoration results requires a measure of image quality. Two commonly used measures are *Mean-Squared Error* and *Peak Signal-to-Noise Ratio*. The mean-squared error (MSE) between two images *g*(*x*,*y*) and h(x,y) of size MXN is:

$$MSE = \frac{1}{MN} \sum_{n=1}^{M} \sum_{m=1}^{N} [h(n.m) - g(n,m)]^{2}$$

One problem with mean-squared error is that it depends strongly on the image intensity scaling. A mean-squared error of 100.0 for an 8-bit image (with pixel values in the range 0-255) looks dreadful; but a MSE of 100.0 for a 10-bit image (pixel values in [0,1023]) is barely noticeable.

Peak Signal-to-Noise Ratio (PSNR) avoids this problem by scaling the MSE according to the image range.

$$PSNR = -10 log_{10} \frac{MSE}{S^2}$$

where *S* is the maximum pixel value. PSNR is measured in decibels (dB). The PSNR measure is also not ideal, but is in common use. Its main failing is that the signal strength is estimated as S^2 , rather than the actual signal strength for the image. PSNR is a good measure for comparing restoration results for the same image.

OBSERVATIONS:

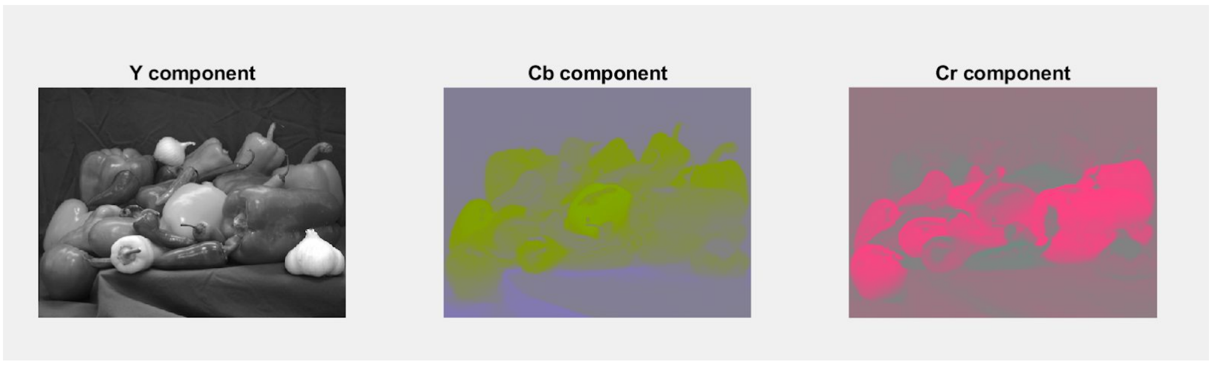


Figure 1: Color components

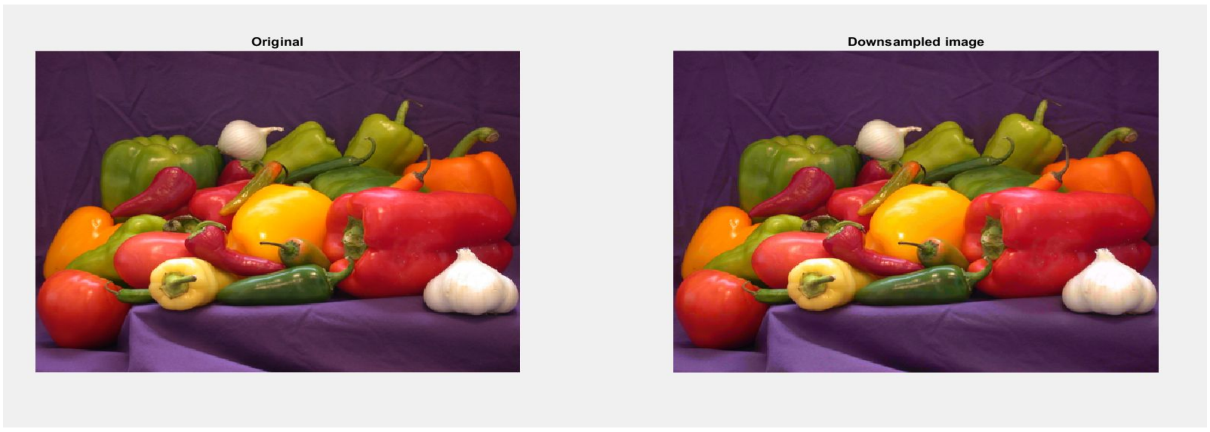


Figure 2: Downsampling image by factor 10

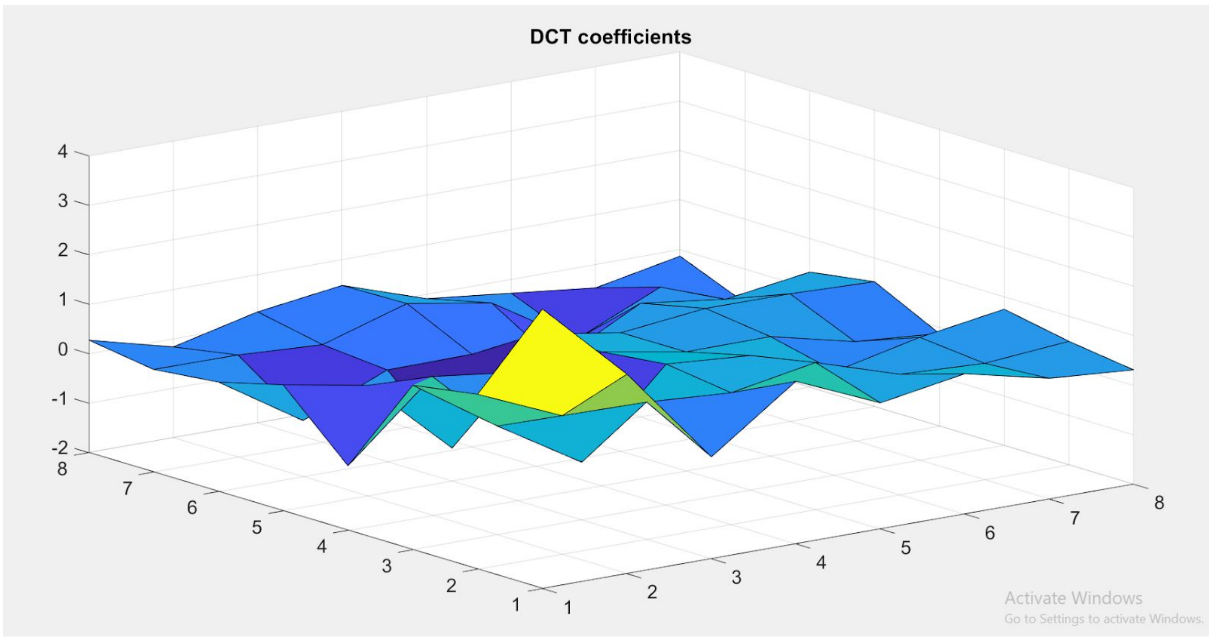


Figure 3: DCT coefficients for one 8X8 block

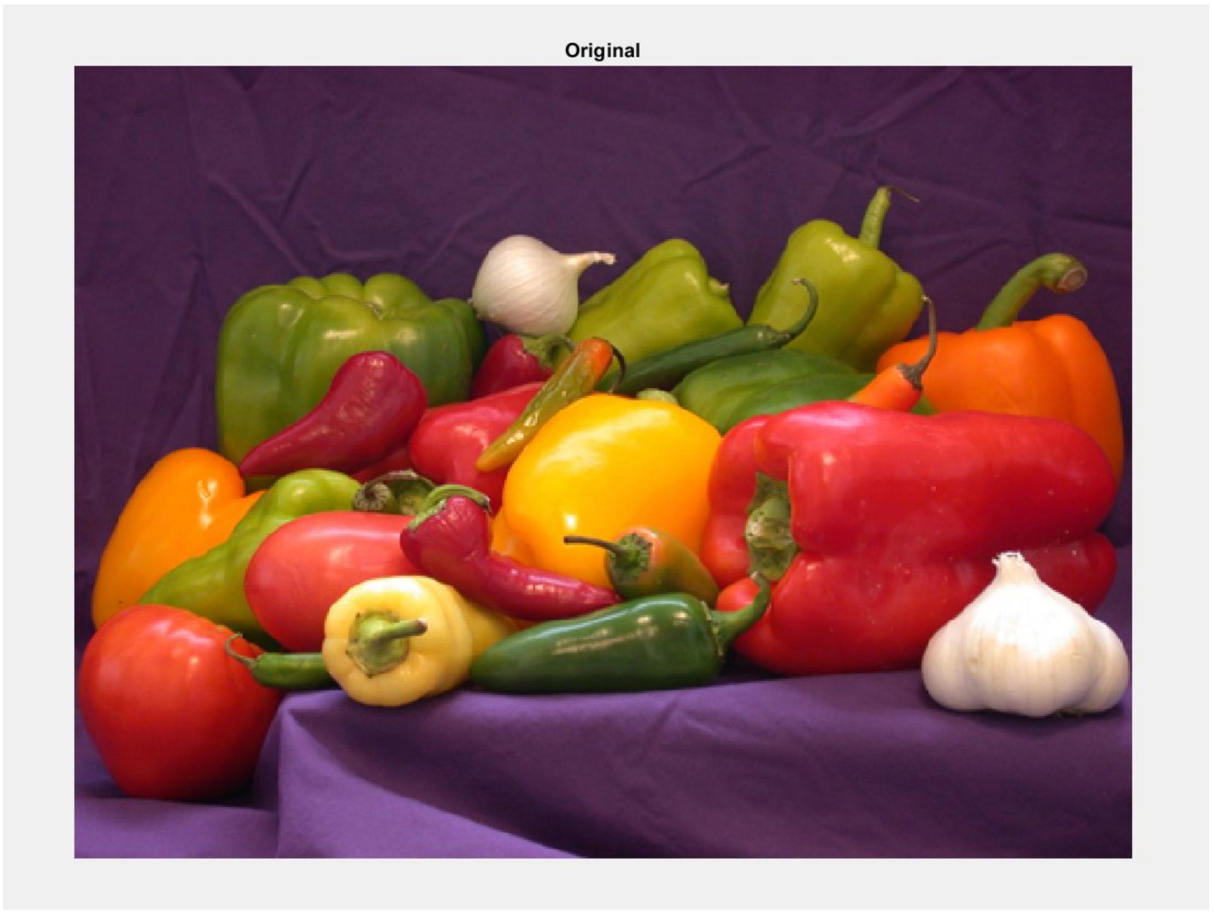


Figure 4: Original input image before compression

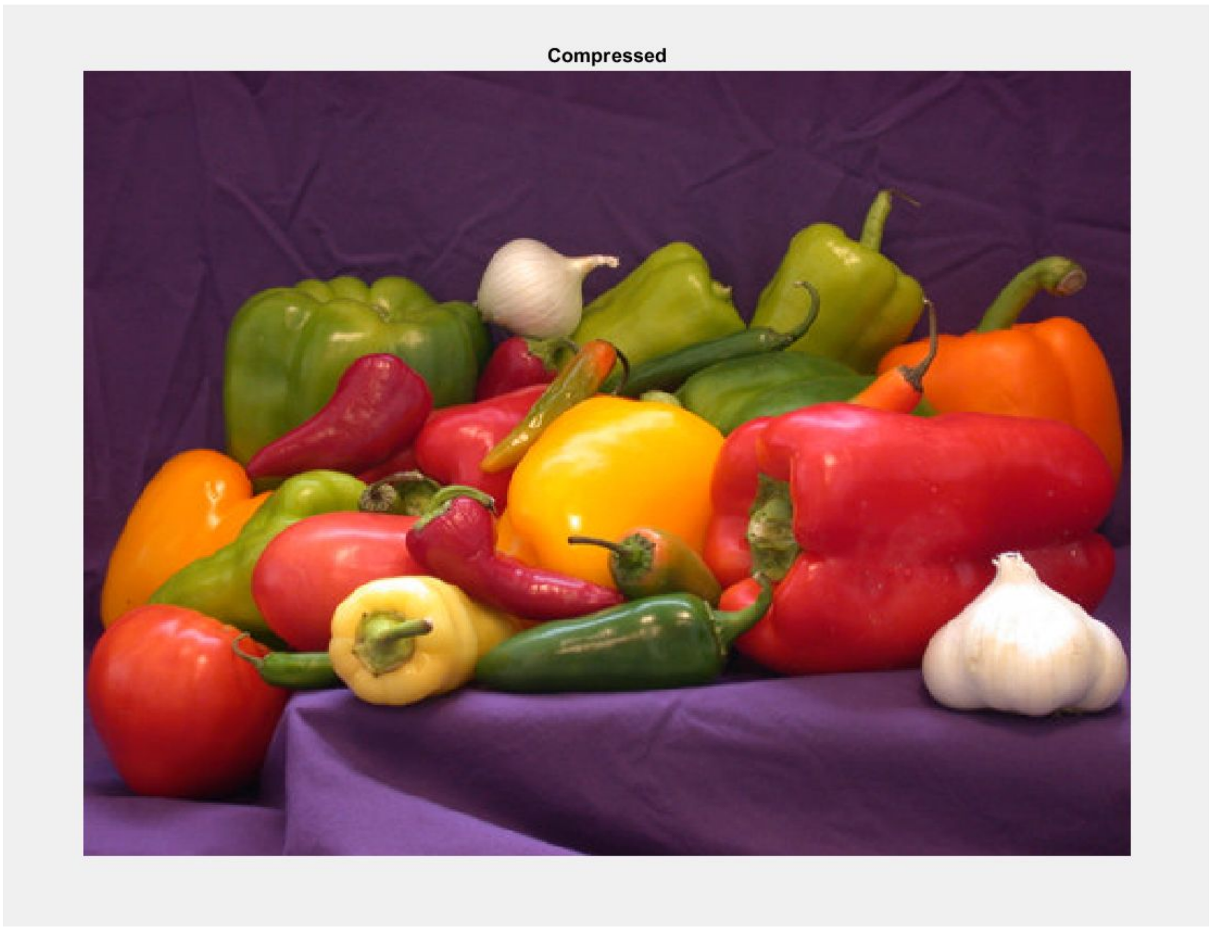
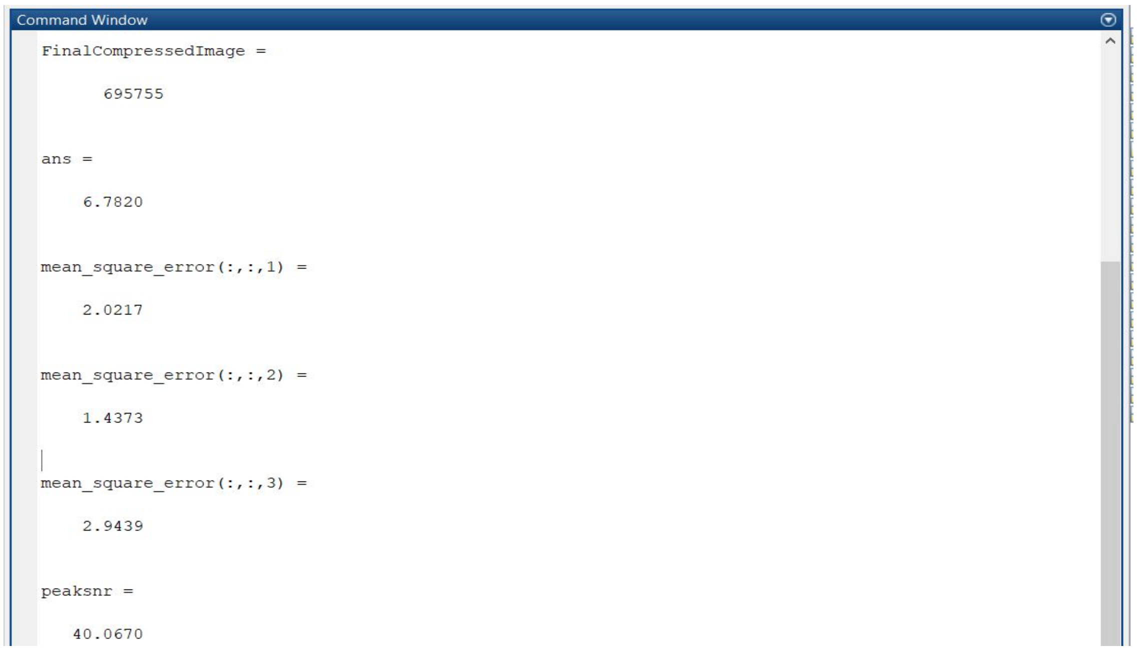
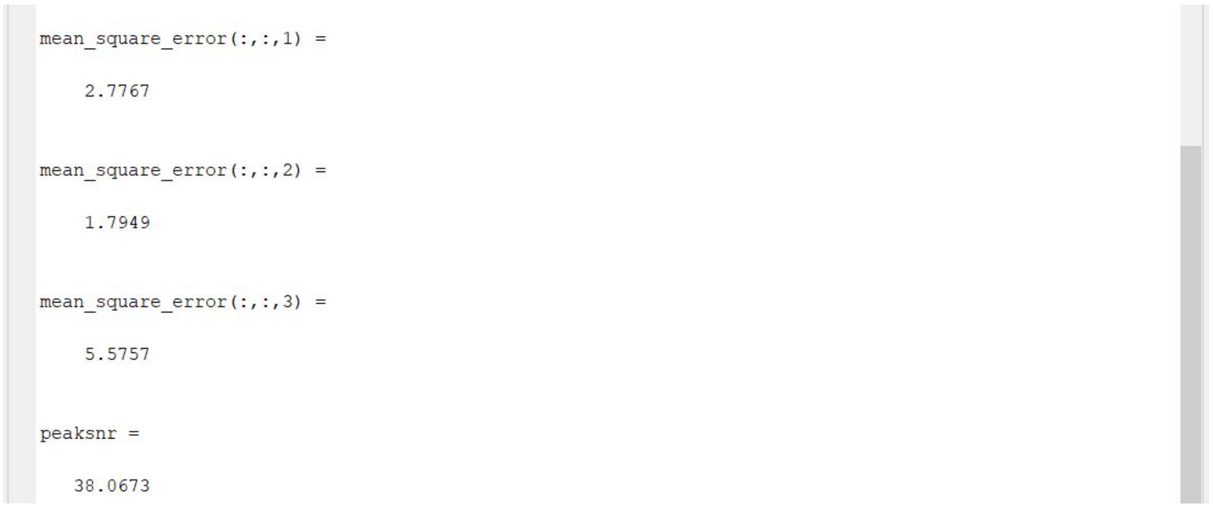


Figure 5: Final output compressed image



```
FinalCompressedImage =

      695755


ans =

    6.7820


mean_square_error(:,:,1) =

    2.0217


mean_square_error(:,:,2) =

    1.4373


mean_square_error(:,:,3) =

    2.9439


peaksnr =

   40.0670
```

Figure 6: Compression ratio, MSE and peakSNR obtained on downsampling by two



```
mean_square_error(:,:,1) =

    2.7767


mean_square_error(:,:,2) =

    1.7949


mean_square_error(:,:,3) =

    5.5757


peaksnr =

   38.0673
```

Figure 7: MSE and PSNR on downsampling by a factor of 10

OBSERVATIONS AND INFERENCES:

Compression Ratio:

Initially the colour components are downsampled by a factor of 2. A compression ratio of a 6.782 is obtained.
Upon downsampling by 5, the compression ratio decreased to 6.2515.
Further increasing downsampling rate to 10, the compression ratio decreased further down to 5.7844.
So an inverse relation between compression ratio and downsampling rate can be inferred from the above observations

Mean Square Error:

At a downsampling rate of 2, a MSE of 2.9439 is observed for YCr colour component. When increased to 10, MSE increases to 5.5757. Similar increase in MSE is observed for all colour components as given in figure.

PSNR:

PSNR value of 40.067 is obtained when downsampled by a factor of 2 and 38.067 when downsampled by a factor of 5. PSNR decreases by increasing downsampling rate.