**Program Discussion**

Van Joseph P. Cabuga, 2019-08453

Ferdinand Gador, 2019-08264

Given a folder containing a sequence of images, the problem is to compress all these images into a single file. Then, the program extracts and shows the compressed images from the file. The method for the program is an altered version of a Set Redundancy Compression method. The program checks through each pixel of each image in the folder and takes note of each pixel's highest and lowest value. With the minimum and maximum values in hand, the program proceeds to compress each image.

**Rationale**

The initial idea for this problem is to merely use the RLE method to compress each image in the folder. However, the Set Redundancy Compression method’s advantage is that it can exploit repeating pixel values throughout the folder. There are three known methods for Set Redundancy Compression: the differential, the predictive, and the centroid method. The next idea is to compress the image via the predictive method. With the predictive method, The program manually calculated the levels for the first row and column of the pixel and proceeded to use the MMP3 prediction with the rest. Then, the program uses these levels to create the predicted pixel. The program only needs to take the difference between the predicted and the original to create the compressed version of the image. The problem with the second idea is decompressing the image when the only value available is the difference between the original and predicted value. The third idea then is to alter the MMP method. Instead of calculating the levels of the pixel, the program proceeds to only shift the range of the maximum and minimum values. The idea now is to reduce the variety of colors in the image, culling values higher than the altered max and min.

**Background**

Set redundancy is a concept established by Karadimitriou in 1996. The existing compression algorithms at the time only recognize the coding, spatial, and psychovisual redundancies. Coding redundancy refers to the repeated information in the coding of an image. An example of this is reducing an image to the least amount of coding necessary to represent all values. Spatial or interpixel redundancy refers to the relationship between the current pixel and its neighbors. Predicting the value of the pixel given its neighbor is a method of image compression. Psychovisual redundancy refers to the minute details of the image not necessary for a person to perceive the entirety. As human vision cannot be as accurate as the computer’s quantitative analysis of an image, simply reducing details can be a form of image compression.

Karadimitrou offers a 4th type of redundancy, the Set Redundancy.Set redundancy is the redundancy of information present in similar images. Similar images are those that have: (a) similar pixel intensities in similar areas, (b) similar histograms, (c) similar edge distributions, and (d) similar distributions of features. Set redundancy can be removed via different methods like the Min-Max Differential, the Min-Max Predictive, The Centroid, and other methods.

The advantage of the Set Redundancy Compression is its ability to take advantage of similar images where previous methods compress individual images. This method is best applied to sequences of images or in medical imaging where large amounts of relatively similar images are produced every day thus there is a need to compress them for storage. Like other compression methods, Karadimitriou’s concept only resolves the set redundancy inherent in similar images so it requires more relatively similar images to be effective.

**Methodology**

The program initially requires the folder where the uncompressed images are located. The program then prepares two arrays to hold the maximum and minimum values each pixel has managed to reach throughout all images. With the maximum and minimum values noted, the program now has a range of values a pixel can be. The program then proceeds to an altered version of the set redundancy compression where it will shift the value of a pixel down. This method is needed to prevent the compressed image from reaching its higher values and becoming heavier. An image with dimmer and darker values turns out to be lighter than an image with brighter values.

When a compressed version of an image is ready, the program proceeds to append it to an array that collects all compressed images. Once the contents of the folder are exhausted, the program proceeds to create a .cmp file of the variable via the pickle method. This .cmp file now contains all the compressed images from the folder. The next step is for the program to read the .cmp file, convert each array of compressed images into an actual readable version, and then present them to the user. The program also showcases the original and the compressed file sizes and their compression ratio as well as the length of time it took for the program to compress the images.

**Results and Discussion**

There are two outputs created in the process of compression and extraction of an image. First is the compiled .cmp file containing all compressed images. The compiled file is much heavier than the sum of its original images. In a sample run, 20 800x450 pixel images with a total file size of 1.65 Mb can create a .cmp file reaching 150 Mb. This .cmp file will still undergo extraction where the final compressed images are lighter than the original.

Interestingly, since the program intends to cull pixels with values higher than its altered max value, images should become dimmer and darker. However, if a sequence of images contains consistent white pixels throughout, its compressed version may become to be heavier. This situation only occurs in sequences of simple images as compression is successful when applied to more complex samples. The hypothesized explanation regarding this situation is that there is not much change between images thus when run through the program’s altered set redundancy compression algorithm, there is not much to compress, thus the image only became a heavier version than the expected result.

**Conclusion**

Successfully compiling several images in 1 go can be a challenge without the set redundancy method. While the program did not follow any of the set redundancy's established methods, the program still managed to compress several sequences of pictures successfully. Merely deducting the pixel values of the image to create a compressed version of it is a simple way of compression. It would be better if the actual methods of compression, that is the MMP, MMD, and centroid methods are emulated. However, because there is a lack of relevant resources for decompressing this data, any hopes of compressing these images are difficult. Using the algorithm employed in this program is not recommended, instead, it would be best to follow the more established methods of Set redundancy compression. Furthermore, the current algorithm does not take full advantage of its maximum and minimum array of values.

**Distribution of Tasks:**

Ferdinand Gador is responsible for the program UI and image decompression.

Van Joseph P. Cabuga is responsible for image compression.

**References**

Ait-Aoudia, S., & Gabis, A. (2006). A comparison of set redundancy compression techniques. *EURASIP Journal on Advances in Signal Processing*, *2006*, 1-13.

Karadimitriou, K. (1996). *Set redundancy, the enhanced compression model, and methods for compressing sets of similar images*. Louisiana State University and Agricultural & Mechanical College.

*What are different types of redundancies in digital image? Explain in detail.* (n.d.). https://www.ques10.com/p/7293/what-are-different-types-of-redundancies-in-digi-1/