PROJECT PLAN
Image Compression based on Non-Parametric Sampling in Noisy Environments

Group 19G01: Kishan Narotam (717 931) & Nitesh Nana (720 064)

School of Electrical & Information Engineering, University of the Witwatersrand, Private Bag 3, 2050, Johannesburg, South Africa

Abstract: This report covers the project plan for the project based on image compression. The project plan looks at the specifications that will define a successful project implementation and a brief look at existing models of image compression. The proposed strategy involves an encoder and decoder side that will be implemented in MATLAB. The encoder side of the strategy involves reading the image, converting to grayscale, dividing the image into a domain pool, creating holes in the image, compressing the image using known technique. The image is encoded and transmitted through a channel and random errors introduced. The decoder involves decompressing the image, filling the holes and reconstructing the final image. A performance index is proposed as well as a look into cost management. Time management is discussed in detail with the aid of a Gantt chart and how the documentation of the project will be done.1

Key words: Image compression, DCT, Wavelets, MATLAB, Project Plan

1. INTRODUCTION

A project plan is a formal document that forms a guideline of the project that will ensure the overall success and completion of the project [1]. Planning the project involves looking at three fundamental aspects of the project:

- Scope: what are the objectives of the project and how it will be completed
- Cost: how much money is allocated, budgeted and spent over the course of the project
- Time: the time taken to execute the project to a point of completion [2].

In the report that follows, a project plan is created for the telecommunications project of *image compression based on non-parametric sampling in noisy environments*. Firstly, the project specifications are defined, followed by a brief look into existing models. Subsequently, the proposed strategy for the project is done with a cost and time management which is discussed in order for the project to be completed on time within a specified budget.

2. PROJECT SPECIFICATIONS

The aim of the project is to create a robust scheme for determining multiple holes that may be received in an image and appropriately fill these holes. in addition, the image may be subject to random burst errors which need to be identified and corrected.

An image will be chosen for compression utilizing a compression technique and creating holes in the image. After transmission, filling of those holes will result in the original image. The holes will be created manually with a proposed algorithm and the additional compression layer will make use of current compression algorithms (DCT or fractal compression). Once the holes are created, the image will be transmitted via a simple channel that will introduce errors randomly. The image will be received at the receiver, and knowing where the holes are present through the encoding scheme will fill them accordingly using a different algorithm. The receiver will also have to deal with random errors that may have occurred from the channel. The received image with the filled holes will be reconstructed and presented as the final image which should coincide with the original image.

3. EXISTING MODELS

Image compression is an application of data compression where an image file is encoded with a few bits with the overall goal of reducing the size of the image file compared to that of the original [3]. There are two main techniques of image compression:

- Lossless
- Lossy

3.1 Lossless

Lossless compression allows the original form of data to be reproduced, thus meaning that the original data from the file before compression is preserved [4]. Some of the current lossless compression techniques include:

- Run-Length encoding
- Huffman encoding
- Shannon-Fano encoding
- Arithmetic
- Dictionary based [5].

3.2 Lossy

Lossy compression removes some of the data from the original file, resulting in an overall reduction of the size of the file [4]. The non-useful parts of the data that is not noticeable is removed, thus reducing the overall quality of the data and file. Some of the current lossy compression techniques include:

- Lossy predictive
- Vector Quantization
- Transform Coding
- Block transform
- DCT/DWT
- JPEG [5].

4. PROPOSED STRATEGY

The proposed solution will be implemented in MATLAB and can be broken down into two main facets:

- Encoder side
- Decoder side.

In conjunction with this, a simple channel will be created and random errors will be introduced after the image is encoded. Figure 1 shows the basic framework and block diagram of the proposed solution that will be implemented.

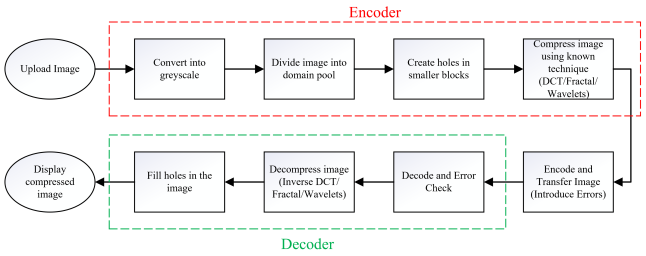

Figure 1 : Block diagram of the proposed strategy

4.1 *Step 1: Loading/Reading an Image into MATLAB*

Firstly, an image is loaded into MATLAB, and a PNG or BMP image is specifically chosen with the dimensions $M \times N$, where M and N are divisible by 8. A PNG or BMP image is used as a JPEG image, is already a compressed image via the DCT image compression technique. Figure 2 shows an example of the image that can be used.

4.2 *Step 2: Convert the image to greyscale*

The loaded image is converted to greyscale. The reason for this is because an image that has been imported into MATLAB creates a 3-dimensional array where the first two elements of the array represent the dimensions of the image and the third dimension is the colour map. Converting the image to a greyscale one, creates a 2-dimensional array, which are simply the dimensions of the image, where each array value correlates to the specific integer value of that specific pixel. Figure 3 shows an example of the outcome of converting the image to greyscale.

Figure 2 : 128×128 image that will be loaded

Figure 3 : Image after it has been greyscaled in MATLAB

4.3 *Step 3: Divide the image into the domain pool*

The image is now divided into smaller 8×8 blocks, creating the domain pool. Once the image has been divided into smaller blocks, we index each block from the top left starting from 1 and increment each index by one, moving left-to-right, top-to-bottom. The reason for creating such an index is so that the receiver will be able to determine which smaller block in the domain pool contains holes so that the receiver can fill these holes. Figure 4 shows an example of the loaded image having a domain pool of 256 blocks.

4.3.1 Indexing the blocks Since the images that will be chosen will have dimensions divisible by 8, the first step in determining how many blocks in the domain pool will be created, would be to divide the width

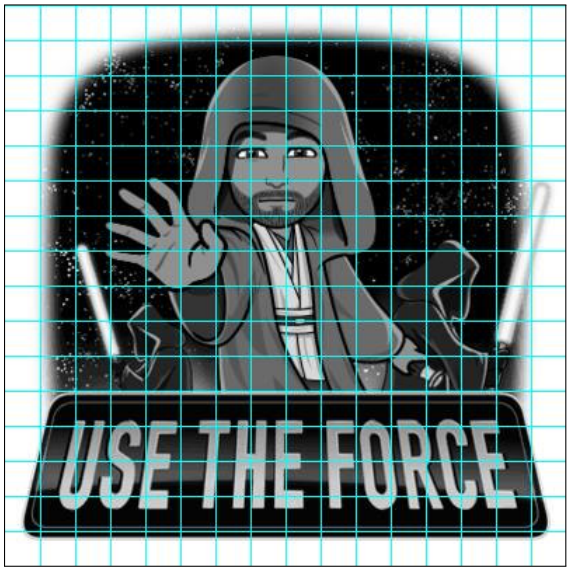

Figure 4 : Smaller 8×8 blocks creating the domain pool

and height of the image by 8 to determine how many blocks there will be going across and down the image. Algorithm 1 shows a very high-level approach to finding the top left pixel of a block in the domain pool

Algorithm 1 High level algorithm of finding the `x,y` coordinates of the respective blocks in the domain pool

```
xBlocks ← widthOfImage/8
yBlocks ← heightOfImage/8
Set initial (x, y) values to be (1, 1) for block 1
To move onto next block:
  set x to 8(i) − 7, where i is the block number
if BlockNumber > xBlocks then
   Set y to 8(i + 1) − 7
   Reset x to be 1 and move to next block
end if
```

4.4 *Step 4: Create holes in the smaller blocks*

With the domain pool created, holes are created by starting at the center square within the 8×8 block. Starting off with the center 2×2 square, the average value of those 4 pixels are calculated. Each pixel in the 2×2 square is compared to the average value of the smaller square using the Chebyshev distance. The Chebyshev distance between each pixel and the average forms the basis of the similarity index that will be used in the project and its implementation. If the value of the similarity index is less than 5 for all pixels, a hole can be created in those 4 pixels. A larger square in the same 8×8 block is checked with a size of 4×4. The average of these pixels are calculated and as before compared to each value of the center square. This can continue until we reach a center square size of 6×6, which we will define as the largest hole that can be inserted. Algorithm 2 shows a high-level algorithm for creating the holes in the blocks in the domain pool.

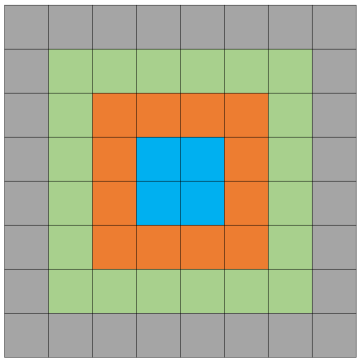

Figure 5 : How a 8×8 block (gray square) will be checked for hole creation starting with the blue square.

Figure 5 shows how each of the 8×8 squares in the domain pool, will be checked, starting with the blue squares, moving up to the orange squares to the largest defined hole which are the green squares.

4.5 *Step 5: Compress image using a known technique (DCT/Fractal/Wavelets)*

The DCT compression technique will be utilized for the compression of the image. DCT will be performed on each block in the domain pool. DCT is implemented on a 8×8 matrix, which is the reason the domain pool is made up of multiple 8×8 blocks and that an image where the dimensions are divisible by 8 is chosen. The image is now compressed and can be transmitted via the channel.

4.6 *Step 6: Encode and Transfer image*

The image is encoded as a binary message, and when transmitted through the channel, errors are randomly introduced. The data that will be sent through the channel will be a bit stream and the errors introduced will be based off of a byte with a probability of 10^{-3} where the byte can then be randomized. Alternatively each bit can be randomly selected, using the same probability above, to be prone to an error and the bit value flipped.

4.7 *Step 7: Decode and Error Check*

Identification of the location of the errors is required. First, identifying the errors are required, and if detected must be corrected and then the image can be decoded. The image will be converted from a bit stream into its 2-dimensional matrix.

4.8 *Step 8: Decompress (Inverse DCT/Fractal/Wavelets)*

Since DCT compression was used, we now perform the inverse DCT to obtain the compressed image and its respective values in the 2-dimensional matrix.

Algorithm 2 High level algorithm of method of creating holes

```
x coordinate ← x+3
y coordinate ← y+3
Calculate average of squares: (x,y; x+1,y; x, y+1; x+1, y+1)
for x till x+1 do
    for y till y+1 do
        Check Chebyshev distance between (x,y) and average
    end for
end for
if Chebyshev distance between average & each pixel ≤ 5 then
    Set value in pixels (x,y; x+1,y; x, y+1; x+1, y+1) to 0
else
    Compress image
end if

x coordinate ← x+2
y coordinate ← y+2
Calculate average of squares (now 4 × 4 square)
for x till x+3 do
    for y till y+3 do
        Check Chebyshev distance between (x,y) and average
    end for
end for
if Chebyshev distance between average & each pixel ≤ 5 then
    Set value in pixels to 0
else
    Compress image
end if

x coordinate ← x+1
y coordinate ← y+1
Calculate average of squares (now 6 × 6 square)
for x till x+3 do
    for y till y+3 do
        Check Chebyshev distance between (x,y) and average
    end for
end for
if Chebyshev distance between average & each pixel ≤ 5 then
    Set value in pixels to 0
else
    Compress image
end if
Compress image
```

4.9 Step 9: Fill holes in the image

With the one-dimensional array sent across the channel, the blocks within the domain pool that contain the holes can be identified. Starting at outer pixels, the average pixel value of the square is calculated, and that will fill the inner pixels where the hole was created.

5. PERFORMANCE INDEX

The nature of this project is based around current methodologies and is more research-based. Results are required to present the overall performance of the proposed technique. MATLAB allows for relative simplicity in creating and performing many image compression techniques such as DCT. For that reason, other compression techniques such as fractal compression or wavelets compression will be implemented in addition to DCT compression. The DCT compression will be set as the benchmark and a comparison with the fractal or wavelets compression will be completed. The values of signal-to-quantization-noise ratio (SQNR), peak-signal-to-noise-ratio (PSNR) and mean-squared-error (MSE) will be the main focus of comparison.

6. COST MANAGEMENT

The nature of this project is software based, and thus the total costs are minimal to none. Licences for the MATLAB software is already provided in the form of a student licence by the university, and personal computers or laptops will be used.

7. TIME MANAGEMENT

The Project window runs from 01 July 2019 until 11 September 2019. The proposed timeline of events and duration of these events can be seen in Figure 6, in conjunction with this, Figure 7 shows the . The project begins with the planning phase including a literature review and the development of the project plan. The overall idea and method was continuously discussed between the team of engineers and the associated supervisor. Once the project plan is approved, the engineers can begin with the execution of the plan.

The encoder side is tackled first while the channel can be created in parallel with that. Once the encoder and channel have been completed and tested, the decoder side of the proposed solution can be implemented. The initial implementation of the encoder will utilize DCT, and once an entire solution is created and tested using DCT, other compression techniques can be implemented in conjunction with the proposed solution. The results of the DCT compression is compared to the other implemented compression technique.

Based on the planned approximated time that is allocated to each task, this indicates that keeping to a tight schedule is required. Since the supervisor will not be available during the first two weeks, the engineers will have to work together and communicate efficiently in order for them to encounter minimal problems and build the proposed solution.

8. DOCUMENTATION

Throughout the project window, constant meetings and discussions will be held between the engineers and the supervisor. As seen in Figure 6 and 7 the *Monitoring and Control* task is run from the day after the project plan is submitted till the project is presented. All meetings have and will be documented as meeting minutes, discussing the meeting and the agenda of the project at that period of time. A GitHub repository is created so that the engineers can constantly work on the proposed strategy while keeping all documents and helpful materials available to them at all times.

9. CONCLUSION

The proposed project plan was given approval by the supervisor. This document discussed existing models of image compression, and gives a detailed description of the proposed strategy of image compression for this project. Block diagrams, example images and high-level algorithms are given as a way of explaining the proposed strategy in more detail. Multiple image compression techniques will be utilized, and the results will be discussed as a performance index. The cost management is discussed, however since the project is research and software-based, no costs will be incurred in the proposed solution. The time management of the project is discussed in detail with a Gantt chart presented as supporting material.

REFERENCES

[1] Techopedia; *What is a Project Plan? - Definition from Techopedia*; `https://www.techopedia.com/definition/24775/project-plan`; Last Accessed: 02/07/2019

[2] Heerkins, G R; *Project Management*; McGraw-Hill; New York, NY, United States; 1st Edition, 2001

[3] Wei-Yi Wei; *An Introduction to Image Compression*; Graduate Institute of Communication Engineering; National Taiwan University; Taipei, Taiwan, ROC

[4] Chapman, C; *Everything You Need to Know About Image Compression — The JotForm Blog*; `https://www.jotform.com/blog/everything-you-need-to-know-about-image-compression/`; Last Accessed: 02/07/2019

[5] Singh, M; Kumar, S; Chouhan, SS; Shrivastava, M (PhD); *Various Image Compression Techniques: Lossy and Lossless*; Journal of International Journal of Computer Applications; Volume 142; Issue No. 6; pp 23 - 26; May 2016

	Task Name	Duration	Start	Finish	Predecessors
1	**Lab Project: Image Compression**	**53 days**	**Mon 01/07/19**	**Wed 11/09/19**	
2	**Planning**	**11 days**	**Mon 01/07/19**	**Mon 15/07/19**	
3	**Research**	**11 days**	**Mon 01/07/19**	**Mon 15/07/19**	
4	Literature Review	11 days	Mon 01/07/19	Mon 15/07/19	
5	Project Plan	11 days	Mon 01/07/19	Mon 15/07/19	
6	Project Plan submission	0 days	Mon 15/07/19	Mon 15/07/19	
7	**Project Execution**	**34 days**	**Tue 16/07/19**	**Fri 30/08/19**	
8	**Encoder Implementation**	**12.5 days**	**Tue 16/07/19**	**Thu 01/08/19**	
9	Upload Image	0.5 days	Tue 16/07/19	Tue 16/07/19	5
10	Convert Image to grayscale	0.5 days	Tue 16/07/19	Tue 16/07/19	9
11	Divide image into domain pool	2 days	Tue 16/07/19	Thu 18/07/19	10
12	Develop hole-creation algorithm	7 days	Thu 18/07/19	Mon 29/07/19	11
13	Compress image using DCT	3 days	Mon 29/07/19	Thu 01/08/19	12
14	Complete Encoder Side	0 days	Thu 01/08/19	Thu 01/08/19	
15	**Channel Implementation**	**5 days**	**Tue 23/07/19**	**Mon 29/07/19**	**5**
16	Set up channel	3 days	Tue 23/07/19	Thu 25/07/19	5
17	Create error probability	2 days	Fri 26/07/19	Mon 29/07/19	16
18	**Decoder Implementation**	**12 days**	**Thu 01/08/19**	**Mon 19/08/19**	
19	Decode and introduce errors	3 days	Thu 01/08/19	Tue 06/08/19	13,17
20	Decompress using inverse DCT	5 days	Thu 01/08/19	Thu 08/08/19	13
21	Fill holes in image	7 days	Thu 08/08/19	Mon 19/08/19	20
22	Complete Decoder side	0 days	Mon 19/08/19	Mon 19/08/19	
23	**Performance Index**	**6 days**	**Mon 19/08/19**	**Tue 27/08/19**	
24	Generate results for DCT compression	2 days	Mon 19/08/19	Wed 21/08/19	21
25	Implement second compression technique	3 days	Mon 19/08/19	Thu 22/08/19	21
26	Final results comparison	3 days	Thu 22/08/19	Tue 27/08/19	24,25
27	Monitoring and Control	28 days	Tue 16/07/19	Thu 22/08/19	5
28	**Project Close**	**11.5 days**	**Tue 27/08/19**	**Wed 11/09/19**	
29	Open Day (Build Demonstration)	1 day	Tue 27/08/19	Wed 28/08/19	26
30	Open Day	0 days	Thu 29/08/19	Thu 29/08/19	
31	Final report	7.5 days	Wed 28/08/19	Fri 06/09/19	29
32	Final report submission	0 days	Fri 06/09/19	Fri 06/09/19	
33	Final presentation	3 days	Mon 09/09/19	Wed 11/09/19	31
34	Final presentation submission	0 days	Wed 11/09/19	Wed 11/09/19	

Figure 6 : Table showing the tasks and time allocated in completing the project

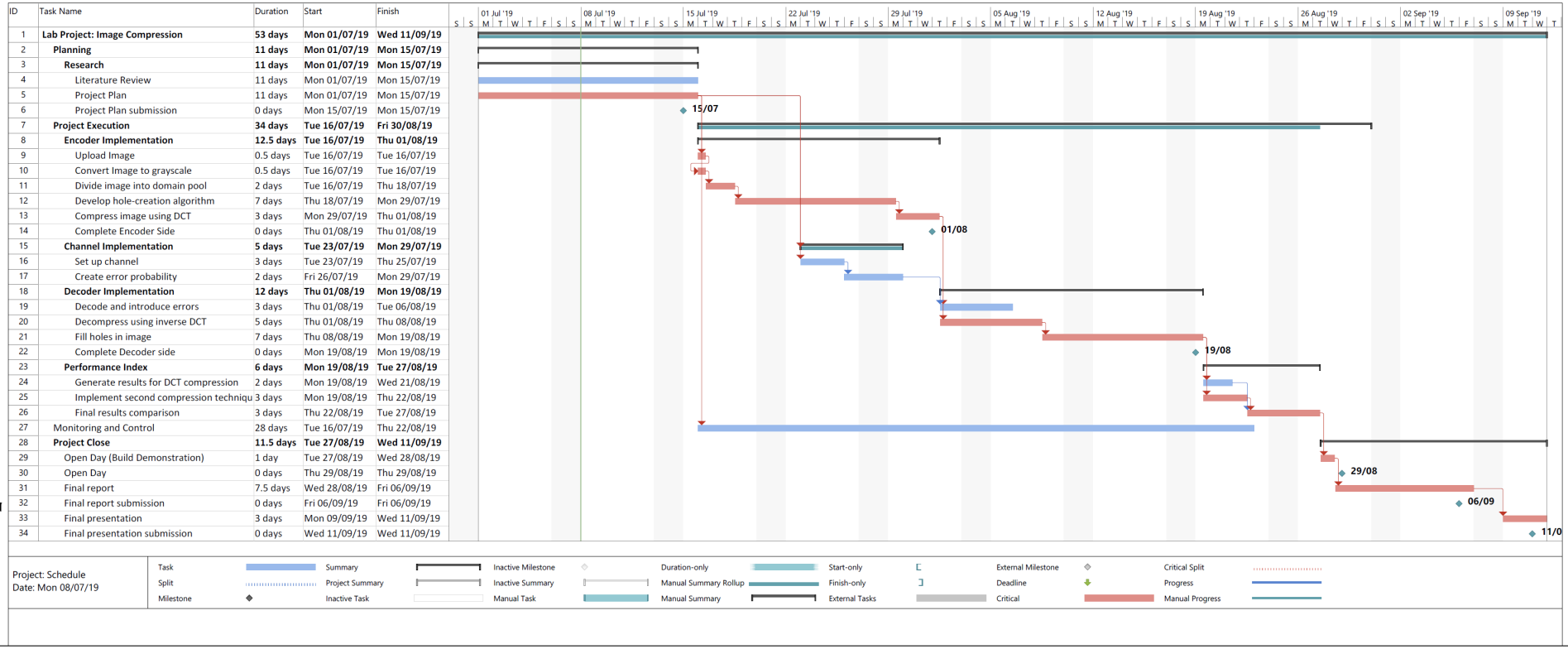

Figure 7 : Gantt chart showing the tasks and time allocated in completing the project with the critical path highlighted