MTHE 493

Wireless Image Communication Between Trail Camera and Receiver

Group T2
Patrick Wilson 20136475
William Jackson 20102189
Mitchell Fillmore 20101212
Timothy Liu 20144307

Submitted On: 10/04/2023

Contents

List of Figures

1 Introduction

1.1 Problem Definition

The team has been contracted to develop the encoding algorithm for a wildlife camera company. The wildlife camera must be able to send its images through densely wooded areas up to a distance of 1 km away using a radio signal generated by the camera. The transmitted image will have to be 720p, and there will be a 90% chance that the image will suffer no worse than 30% pixel change over transmission. The diminishing population of many species of wildlife is a major concern for conservationists, so the main goal is to ensure the wildlife can be successfully monitored. Being able to constantly monitor the whereabouts and frequency of the wildlife is essential to combat the diminishing wildlife population.

In addition to meeting the primary performance goals, reducing energy consumption and data usage are secondary goals. The conservationist company wants to reduce costs, and minimizing energy consumption and data usage will help achieve this. As such, the team is committed to developing an encoding algorithm that optimizes the use of resources without sacrificing image quality or reliability.

It's worth noting that the team has assumed that the hardware level radio transmission protocols will be capable of sending and receiving the radio waves sent to them. Therefore, the team's focus is solely on developing the encoding algorithm that will allow the camera to transmit high-quality images over long distances. With this in mind, the team is working diligently to create an algorithm that meets the company's performance goals while also ensuring the long-term sustainability of wildlife populations.

2 Stakeholders

A primary stakeholder interested in a remote wildlife camera would be wildlife conservationists. Their job is to monitor the whereabouts of endangered species, ensuring that the animals are safe. They cannot physically be there because that would frighten the endangered species away. As a result, they will need cameras to transmit the pictures of the animals' whereabouts. It must be implemented through wireless communication because the surrounding wildlife would destroy wires, and the cameras would not work.

In addition to wildlife conservationists, there are other primary stakeholders who would be interested in a remote wildlife camera. One such group is researchers who study animal behavior in their natural habitats. These researchers could use the camera to track and analyze the behavior of different animal species, which could lead to a better understanding of how they interact with their environments and with each other. This information could then be used to inform conservation efforts and improve our understanding of animal behavior in the wild. For example, conservationists could monitor the movements of critically endangered species to ensure their survival and to identify patterns in their behavior that could help inform conservation efforts.

A secondary stakeholder interested in a remote wildlife camera would be hunters who are looking to track the movement of certain animals for sport. While this group may have a vested interest in using the camera for hunting purposes, ethical considerations must also be taken into account. The use of a remote wildlife camera could eliminate the time wasted by patiently waiting

for an animal to come, allowing hunters to determine the optimal time to hunt and potentially get a higher yield in less time. However, it's important to ensure that hunting practices are sustainable and do not harm the environment or endanger animal populations. Conservationists and wildlife management agencies can use data gathered from remote wildlife cameras to monitor and manage animal populations to ensure that hunting practices are ethical and sustainable. This could include setting hunting quotas, restricting hunting in certain areas, or even implementing new conservation efforts to support animal populations. Ultimately, the use of remote wildlife cameras in hunting practices must be approached with a balance between achieving hunting goals and maintaining the long-term health and sustainability of the environment and animal populations.

Furthermore, remote wildlife cameras could be used by wildlife management agencies to monitor and track the populations of game animals. This information could be used to set hunting quotas and inform management decisions that promote healthy and sustainable wildlife populations.

Another secondary stakeholder is homeowners in remote locations who want the ability to check if any unwanted visitors are trespassing on their property ranging from bears to lost hikers. The wireless camera would appeal to remote homeowners because in remote areas the reception is too weak to transmit images back to the homeowner, so this camera would add a layer of security ensuring the safety of the homeowner's valuables when they are not home. Additionally, remote wildlife cameras could be used by farmers who have livestock in remote areas. These cameras could be used to monitor the movements of predators such as wolves or coyotes, which can threaten the safety of livestock. By installing a remote wildlife camera, farmers could be alerted to the presence of predators and take steps to protect their livestock, ultimately leading to better animal welfare and higher yields for farmers.

Another group of secondary stakeholders would be tourists or adventurers who want to observe wildlife in remote locations. Remote wildlife cameras could be installed in popular wildlife viewing areas, allowing tourists to see animals in their natural habitats without disrupting their behaviors. This would allow for a more authentic wildlife viewing experience and contribute to the growth of eco-tourism in these areas. Wildlife cameras could also be used by environmentalists to monitor the impacts of climate change on different animal species, including changes in migration patterns and habitat use. This information could then be used to advocate for policy changes to protect endangered species and their habitats.

3 Background

3.1 System Description

The system is a peer-to-peer network in which stationary trail cameras will transmit images to a mobile receiver using electromagnetic (EM) waves. Channel complexity arises largely from natural obstacles such as vegetation, elevation changes, and weather. These obstacles make traditional line-of-sight transmission over radio frequencies difficult (typically rely on unobstructed space). Remote cameras also have a limited power supply and need to be light for mobility purposes, adding complexity to transmission and encoding techniques. The focus of the solution will be on the mathematics of channel modelling and encoder/decoder design.

3.2 Assumptions

We assumed that we are not responsible for any other aspect of design of the camera systems other than the image codecs. Our main goal was to create the lowest distortion possible quantizers given the different possible compression rates and BSC switching probabilities. Our channel optimized quantizers assume that the channel we are sending through is a discrete memoryless channel with a fixed probability of error. We assume that our image samples are a good representation of the true source distribution the wildlife camera would be capturing images from when training the quantizer. We assume that our test images are a good general representation of the performance of the various quantizers with respect to their end to end tests.

We also assume that the camera will have sufficient memory and power to perform the image compression/transmission in a timely manner. With respect to the image sources, we assume that the camera will have a similar resolution to that of our source images, so that reconstructed images will have a similar appeal to the human eye. When creating the lookup tables for speedier quantization (rather than calculating the optimal centroid for each new point based on the switching probability) we assumed that nearby points quantize to the same values optimally. We assumed that fidelity of different image transmission will be affected similarly by dropping the zero encoded values from the DCT.

3.3 Discrete Cosine Transform

The discrete cosine transform is an orthogonal transformation, which can be represented in matrix form. The transformation can be done on a $N \times N$ block, but for the purposes of this thesis, it will be performed on an 8×8 block. The DCT can be represented as follows,

$$X = Cx$$

and the inverse DCT can be represented as follows,

$$x = C^{-1}X$$

where C is an 8×8 matrix with elements defined as follows: When k=0

$$C_{k,n} = \sqrt{\frac{1}{8}}$$

When $k > 0$

$$C_{k,n} = \frac{1}{2}\cos(\frac{\pi}{2}(n + \frac{1}{2})k$$

The 2-dimensional DCT can be expressed as

$$X = CxC^T$$

and the inverse DCT can be expressed as

$$x = C^T XC$$

3.4 Channels

A channel is a noisy medium used to transmit information from the source encoder to the source decoder, in which outputs are not deterministic of inputs. The probability of the channel output, y, given an input, x, is the *transition probability* and is denoted by $p(y|x)$

There are two types of channels that were taken into considerations during the design process: Discrete Memoryless Channels (DMC) and Channels with Memory.

3.4.1 Channels with Memory

The channel under consideration is a channel with memory where the memory occurs in an additive noise process. Specifically we have a binary channel described by the following,

$$Y_i = X_i \oplus Z_i$$

Where X_i is the channel input, Z_i is the noise source, and Y_i is the output. $\{Z_i\}_{i=1}^{\infty}$ is a stationary ergodic Markov process of order M with the following property

$$Pr\{Z_i = e_i | Z_{i-1} = e_{i-1}, \cdots, Z_1 = e_1\} = Pr\{Z_i = e_i | Z_{i-1} = e_{i-1}, \cdots, Z_{i-M} = e_{i-M}\}$$

where $e_i \in \{0, 1\}$. The probability of Z_i is as follows,

$$Pr\{Z_i = 1 | Z_{i-M} = e_{i-M}, \cdots, Z_{i-1} = e_{i-1}\} = \frac{\epsilon + (\sum_{j=i-M}^{i-1} e_j)\delta}{1 + M\delta}$$

For order $M = 1$, the probability of Z_i becomes

$$Pr\{Z_i = 1 | Z_{i-1} = e\} = \frac{\epsilon + e\delta}{1 + \delta}$$

and the state transition probability becomes

$$\begin{bmatrix} Q(0|0) & Q(1|0) \\ Q(0|1) & Q(1|1) \end{bmatrix} = \begin{bmatrix} \frac{1-\epsilon+\delta}{1+\delta} & \frac{\epsilon}{1+\delta} \\ \frac{1-\epsilon}{1+\delta} & \frac{\epsilon+\delta}{1+\delta} \end{bmatrix}$$

A channel with memory is complex to implement; however, this model more accurately simulates the behaviour of noisy mediums.

3.4.2 Discrete Memoryless Channel

Under a discrete memoryless channel (DMC) the value of the additive noise is independent of all previous values. The DMC $\{Z_i\}_{i=1}^{\infty}$ exhibits the following property.

$$Pr\{Z_i = e_i | Z_{i-1} = e_{i-1}, \ldots, Z_1 = e_1\} = Pr\{Z_i = e_i\} \quad \forall i \geq 1$$

Let $e_i = \epsilon$, the state transition probabilities will then be as follows,

$$\begin{bmatrix} Q(0|0) & Q(1|0) \\ Q(0|1) & Q(1|1) \end{bmatrix} = \begin{bmatrix} 1-\epsilon & \epsilon \\ \epsilon & 1-\epsilon \end{bmatrix}$$

3.5 Source Coding

To reduce and meet the bit rate requirements for transmission or storage source coding is used to remove the redundant information in the source. There are two main types of source redundancy: statistical redundancy, which is due to source memory, and/or the source non-uniform distribution. Lossless data compression is used to remove the statistical redundancy, so that there is zero distortion when comparing the recovered source with the original source. The second type of source redundancy is non-statistical redundancy, which can be in the form of psycho-visual redundancy. Human vision cannot see every pixel in an image, and this can be exploited in lossy compression to minimize the bit rate while still maintaining a certain fidelity.

3.6 Quantization

Quantization is the process of mapping values to another (usually smaller) set of values. In digital signal processing, this usually involves sampling a continuous signal and mapping it to a countable set with finite order, that way it can be represented in binary with a maximum bit length.

3.6.1 Scalar Quantization

We will be considering quantizers $q(x)$ which are functions $q : \mathbb{R} \to C$, where C is the set of valid codewords $C \subset \mathbb{R} = \{y_1, ..., y_n\}$.

A scalar quantizer is a quantizer that considers each signal sample individually. A scalar quantizer partitions the real line into disjoint sets, and maps all real values within the partition to the same output codeword.

Define S_i to be the ith partition on $\mathbb{R}$

$$q_{scalar}(x) = \{\ y_i, \quad \text{if } x \in S_i \quad \}$$

Note the union of all $S_i = \mathbb{R}$

A performance metric of concern is the mean square error of the scalar quantizer over a random variable. For a given input random variable, X, with pdf $p(X)$ we define the mean square error to be

$$E[d(x, q(x))]$$

where

$$d(x, y) = (x - y)^2$$

Minimizing the mean square error will result in a "closer" average distance from the real values of X.

A quantizer is said to be a Lloyd Max Quantizer if it satisfies the conditions:

- The Nearest Neighbour Condition: Values are mapped to the centroid that minimizes the values distortion. $q(x) = y$ s.t. $y = argmin_{y_i}(d(y_i - x))$

- The Centroid Condition: Centroids of a partition are chosen such that they minimize expected distortion over the region. $y_i = argmin_{y \in S_i}(E_{x \in S_i}[d(x, y_i)])$

3.6.2 Vector Quantization

Vector quantization is very similar to scalar quantization, with the exception that we are now dealing with sets of samples as the input to our quantization function.

Consider $x \in \mathbb{R}^n$. This is our set of samples. A vector quantization is a complete disjoint partition on $\mathbb{R}^n$ composed of partition sets S_i such that

$$q_{scalar}(x) = \{ \ y_i, \quad \text{if } x \in S_i$$

where y_i is the output vector for that particular partition.

An important performance metric of the quantization function is again the mean square error. This time, we define

$$d(x, y) = \sum_{i=1}^{n} (x_i - y_i)^2$$

where n is the magnitude of the vectors x and y.

Intuitively, we can infer that performance limits of a vector quantization will exceed those of scalar quantization, by exploiting dependencies between samples.

3.7 Peak Signal to Noise Ratio

Peak signal to noise ratio (PSNR) is a measurement used to calculate the differences of quality between the original and compressed image. For grey scaled images, PSNR is calculated as shown

$$PSNR = 10 \log_{10} \frac{255^2}{MSE}$$

where 255 is the peak signal/pixel value of an image and MSE represents the mean squared error between both images. This measurement is preferred over Signal to Noise Ratio (SNR) since SNR performs poorly for homogeneous images as it is defined relative to a signal range, whereas PSNR is defined relative to the peak dynamic range.

3.8 Channel Optimized Scalar Quantization

The problem the channel optimized scalar quantization solves is how to construct a quantizer that minimizes expected distortion, given a channel that will change the indexed value received by the inverse quantizer. When the channel is without noise, then we have the centroid and nearest neighbor conditions as outlined in section 3.6.1. When noise is introduced into the channel, then a more generalized form of the centroid and nearest neighbor condition must be used.
Nearest Neighbor Condition

$$S_i = \{X : \sum_{j=1}^{N} p(b(j)|b(i))(x - y_j)^2 \leq \sum_{j=1}^{N} p(b(j)|b(i))(x - y_j)^2, \quad \forall j \neq i\} \tag{1}$$

Centroid Condition

$$y_j = \frac{\sum_{i=1}^{N} p(b(j)|b(i)) \int_{S_i} xp(x)dx}{\sum_{i=1}^{N} p(b(j)|b(i)) \int_{S_i} p(x)dx} \tag{2}$$

Additionally, the generalized distortion for a channel with noise will become:

$$D = \sum_{i=1}^{N} \int_{S_i} \sum_{j=1}^{N} p(b(j)|b(i))(x - y_i)^2 p(x)dx \tag{3}$$

Where $b(i)$ and $b(j)$ are the received and quantized bits.
In each of the generalized equations, noise is introduced in the term $p(b(j)|b(i))$ and the distortion becomes a weighted average of the mean squared value.

3.9 Code Rate and Bit Allocation

The bit allocation problem is the minimize distortion for a set of quantizers, with a limited number of bits, as follows for random variables $X_1, \ldots, X_k$ with corresponding distribution functions $f_1, \ldots, f_k$,

$$D(b) = \sum_{i=1}^{k} W_i(b_i) \quad s.t. \sum_{i=1}^{k} b_i \leq B$$

The optimal bit allocation is given by the following,

$$b_i = \bar{b} + \frac{1}{2} \log_2 \frac{\sigma_i^2}{\rho^2}$$

where,

$$\bar{b} = B/k$$

$$\rho^2 = (\prod_{i=1}^{k} \sigma_i^2)^{\frac{1}{k}}$$

$$||f_i||_{\frac{1}{3}} = ||\tilde{f}_i||_{\frac{1}{3}} \sigma_i^2$$

$$h_i = \frac{1}{12} ||\tilde{f}_i||_{\frac{1}{3}}$$

and B denotes the total number of allocated bits, k denotes the total number of quantizers, b_i denotes the bits allocated for X_i, and σ_i is the standard deviation for X_i.
and the minimum distortion achieved is

$$D = kh\rho^2 2^{-2\bar{b}}$$

[2]. The encoding rate for binary strings is given by,

$$R = \frac{1}{k} \log_2 N \text{ (bits per sample)}$$

These equations are used to derive the optimal bit allocation for a set of quantizers. The bit allocations used in Cheng's thesis will be used to train models in the proposed solution [3]. It is assumed that the hardware will be provided and limited in computing power. In general, high bits will yield lower distortion, but hardware assumptions will limit the number of bits that will be computationally feasible. By using the principles of optimal bit allocation, optimal parameters can be calculated for varying total bits.

4 Simulation Design

4.1 Memoryless Channel Implementation

The memoryless channel was simulated in python. When an value in the DCT matrix was quantized, the index is passed into the simulating function. The function would convert the number into binary and then generate a random floating point between 0 and 1 (with uniform probability). If the number is in $[0, \epsilon)$ the bit would be flipped and kept the same otherwise. The resulting binary string would then be converted back to an integer, then passed to the inverse quantizer.

5 Design Process

5.1 Tools Used

In the assumptions, it is assumed that the client will provide the hardware required to complete the project. Therefore, this project will primarily focus on developing the software that will be implemented on the camera. The primary tools that were used include GitHub, OpenCV, C, and Python.

5.1.1 Third Party Software

GitHub is a service, which was used to host the code base. This provided the collaborative tools necessary for team members to simultaneously edit program files. In order to import images as a matrix with RGB values, OpenCV was use in conjunction with Python.

5.1.2 Programming Languages

This project uses both Python and C to develop the simulator and quantizer. Python is a dynamically typed language with multiple imports, allowing for faster development time. However, because of this flexibility, the run times in Python can be slow. Contrarily, C is a statically typed language. This causes development time to be longer, since variables have to be assigned a type in code. However, because types are known at compile time, C has a faster run time than Python. Given these benefits and disadvantages, the main program will be developed in Python and computationally expensive functions will be developed in C and ported into Python.

5.2 Evaluation Criteria

From the assumptions, hardware will be provided for the project. Due to the portability and small size of the cameras, it is assumed that the computational power and storage on the camera will be limited. Therefore, the solutions/iterations will be evaluated primarily based on the computational power required and speed the quantizer can be trained and compress and image.
PSNR will be used to measure end to end performance of the channel optimized quantizer. However, images that retain the animal and its features will be sufficient for the context of this project.

5.3 Joint vs Tandem Coding

A joint source-channel encoder will be employed using combined source-channel coding. Using a joint source-channel encoding technique will allow the solution to avoid limitations of source and channel coding theorems which effect separately designed (tandem) source and channel encoders. These theorems impose that arbitrarily long block codes will need to be used to reach optimally,

increasing delay and complexity [3]. Given the increased complexity of utilizing interleaving to render the channel locally memoryless, a tandem source-channel encoder must be avoided.

Also, we chose a joint encoding method for algorithmic efficiency. Very broadly speaking, one quantization step should require less computational work than two separate encoding steps for similar performance.

It is also worth noting that Shannon's source channel separation theorem informs us that joint and tandem encoding/decoding schemes have the ability to perform similarly.

5.4 Iterative Process

The following section details the improvements and changes made in the quantization program, as well as a background on how the program in each iteration is structured. The program runs the following stages when training a channel optimized quantizer:

- Import images and training Set
- Divide images to 8x8 blocks
- Apply DCT to 8x8 blocks
- Apply generalized Lloyd Max Algorithm to training set, until centroids converge

After the quantizers are trained, the code runs the following steps to compress and send the image through a noisy channel:

- Breaks image into 8x8 blocks
- Applies DCT
- Quantizes values in DCT transformed Matrix
- Applies inverse DCT
- Reconstructs image from 8x8 blocks

5.4.1 First Iteration

Training Time Results
In the first iteration, the quantizers were implemented purely in Python, including the recursive training function and image compressor. Results are summarized as follows,

Quantizer	Training Time (100,000 Samples)	Image Compressing HD Image
24 Bit	2+ Hours	5 Minutes
58 Bit	Unknown	Unknown
76 Bit	Unknown	Unknown

Table 1: First Iteration Run Time Results for Varying Quantizers

Due to the slow run time of the program and limited computation power available, the 58 bit and 76 bit quantizer could not converge in a reasonable amount of time. Because of this, many results were not collected in this iteration.

Improvements for next iteration
Key results from this iteration include that a majority of the run time was spent on the applying the generalized Lloyd max quantizer. This was due to the recursive function being developed in Python. The inefficiencies in the dynamically typed language were reflected more significantly in the training step. Many times functions such as finding the centroid with the minimum distortion and calculating transition probabilities between bits, would increase exponentially with the bit rate and training samples. Therefore, the efficiency of training needed to be improved in the next iteration, in order to train the 58 bit and 76 bit quantizer and attain more results.

5.4.2 Second Iteration

Training Time Results
In order to reduce the time spent training, the training function was developed in C and ported to the main python file. This allowed for most of the existing code to be able to be reused, while reducing the time spent training the quantizer. Results are shown as follows,

Quantizer	Training Time (100,000 Samples)	Image Compressing HD Image
24 Bit	15 Minutes	30 Minutes
58 Bit	1 Hour	2 Hours
76 Bit	6 Hours	10+ Hours

Table 2: Second Iteration Run Time Results for Varying Quantizers

As seen, the training time for the 24 Bit Quantizer was reduced significantly and the 58 Bit and 76 Bit Quantizers could be trained in a reasonable about of time. However, the image compression time was significantly larger for the 76 bit quantizer. A 10 hour compression time would not be a feasible amount of time given the assumptions of the problem and limited hardware in the cameras. Therefore, improvements would be need to be made in compression time.

Quantization Results
Despite these improvements that needed to be made in the next iteration, the quantizers were able to be trained and yielded the following results.

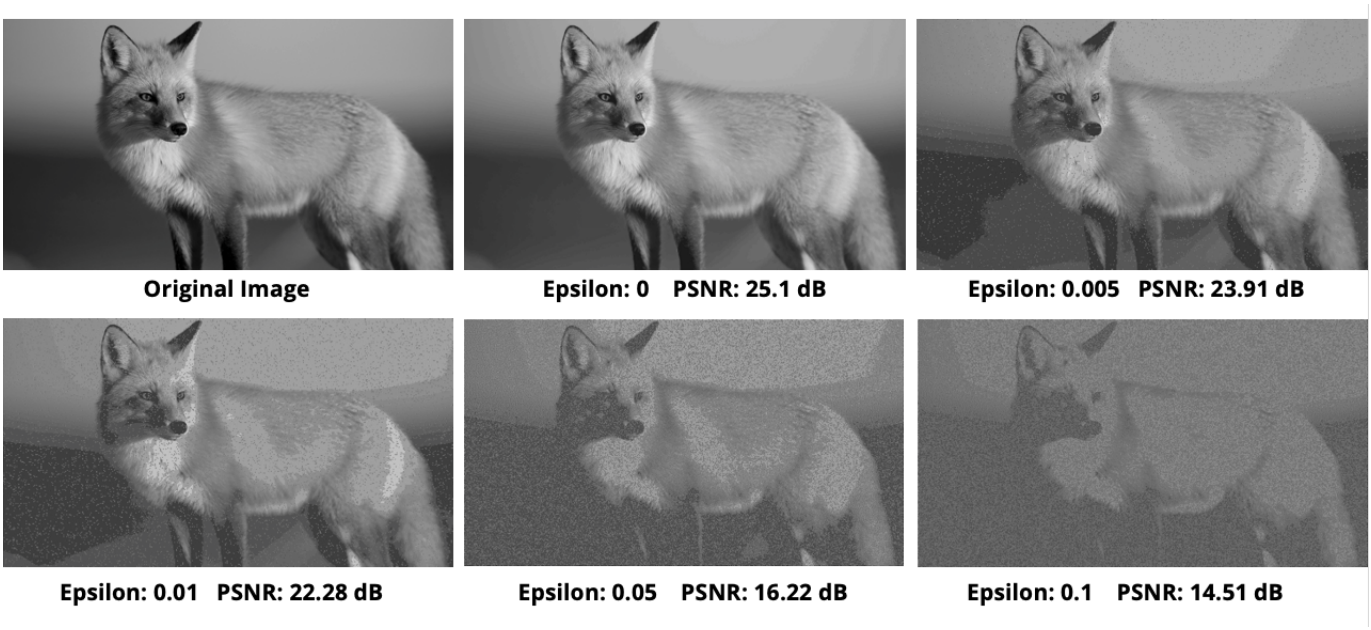


Figure 1: Images and PSNR of 24 Bit Quantizer with Varying Channel Noise

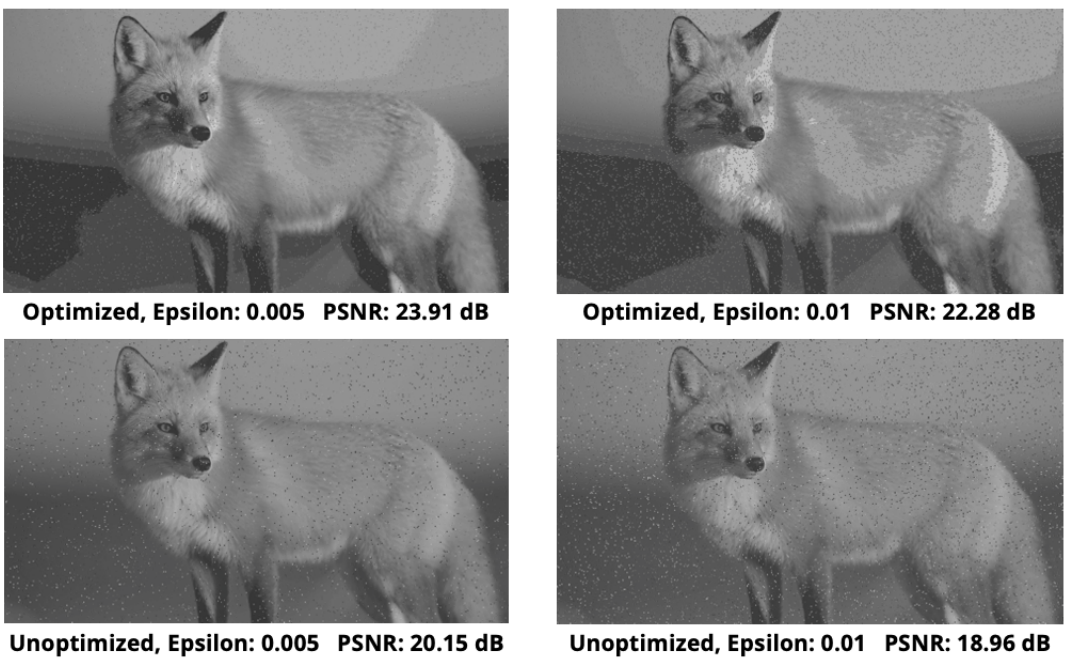


Figure 2: Channel Optimized vs Channel Unoptimized Quantizer (24 Bit)

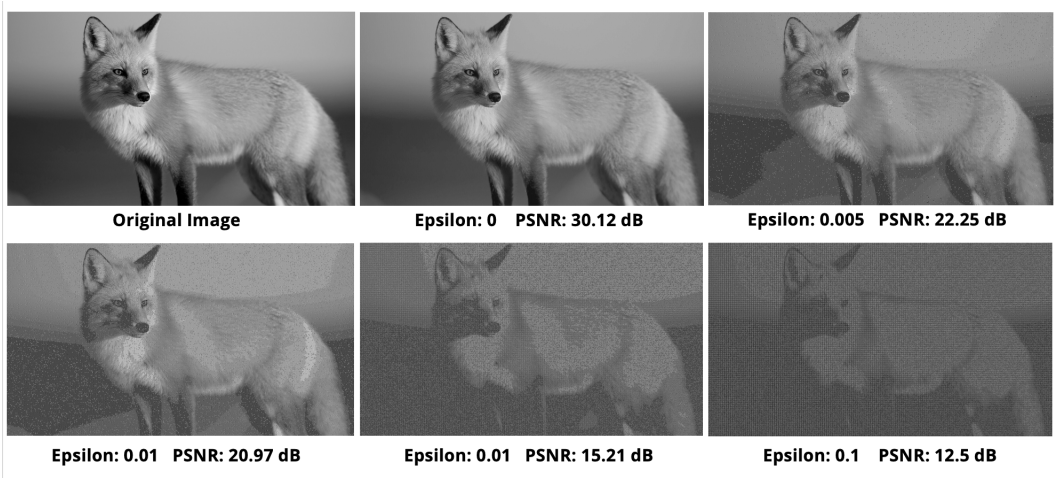


Figure 3: Images and PSNR of 58 Bit Quantizer with Varying Channel Noise

From Figure 1, the graphic shows the PSNR for varying values of channel noise. The relationship between ϵ and PSNR are as expected: higher values of ϵ lead to lower values of PSNR. Additionally, from Figure 2, it can be seen that the channel optimized quantizer is improving image quality compared to its unoptimized counter part. These results are as expected and demonstrates that the channel optimized scalar quantizer does improve image quality in practice. However, the PSNR values were lower than expected for higher bit quantizers. In fact, higher bit quantizers performed worse than their lower bit counterparts when noise is present in the system. The distortions for each bit quantizer were then investigated, as shown, to analyze the potential issue.

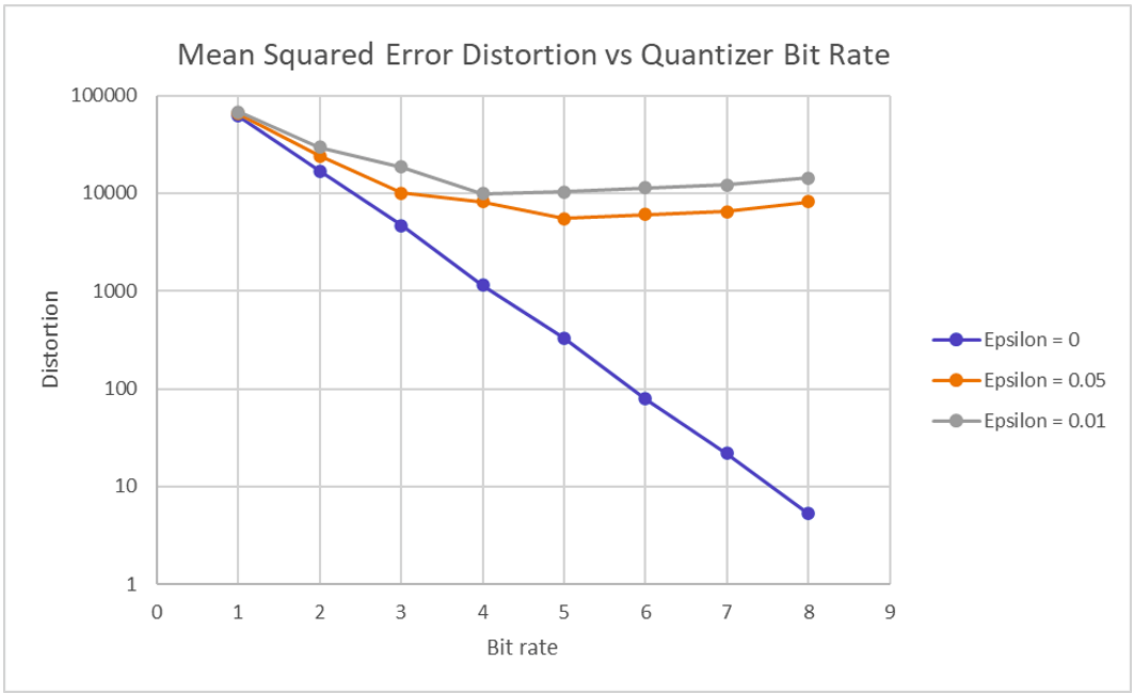


Figure 4: Distortion Graph

Results that were unexpected include the distortion curves in Figure 4. It is expected that dis-

tortion will decrease as bit rate increases. However, this is only true for distortions with $\epsilon = 0$. When channel error is introduced, the quantizer distortion increases after 5 bits. This result is unanticipated and this may be caused by the channel optimized quantizers not being fully trained. Therefore, in future iterations, the quantizers should have a stricter training requirement and should be run for more iterations.

5.4.3 Third Iteration

When noise is introduced into the channel, quantizing a value in the DCT block is a $n^2 \log n$ time complexity computation. This goal of this iteration was to reduce the computation power required to compress and quantize values of an image. In order to achieve this, a *quantization map* was created, which is a hash-map of pre-computed values and their quantized values. These pre-computed values would range from -2000 to 2000 with increments of 0.1. An incoming value would be rounded to the nearest 0.1 decimal place and then be quantized with the corresponding value in the hash-map. This would effectively reduce the computation time from $n^2 \log n$ to constant time for each quantization.

Run Time Results

Quantizer	Training Time (100,000 Samples)	Image Compressing HD Image
24 Bit	15 Minutes	<5 Minutes
58 Bit	1 Hour	<5 Minutes
76 Bit	6 Hours	<5 Minutes

Table 3: Third Iteration Run Time Results for Varying Quantization Values

Sample Images

Figure 5: Sample Images of Optimized and Unoptimized Compression for 58 Bit Quantizer $\epsilon = 0.1$

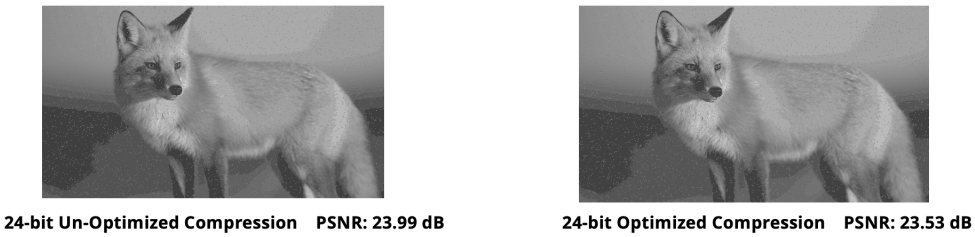


Figure 6: Sample Images of Optimized and Unoptimized Compression for 24 Bit Quantizer $\epsilon = 0.005$

From Figures 6 and 5 we can see that there is a difference in the PSNRs when using optimized and unoptimized compression. The run time is significantly reduced across all quantizers for the optimized compression system. However, the trade off is an approximate 3% decrease in PSNR. However, this reduction in quality is visually negligible. Therefore, the trade off for reduced computation complexity is worth the small reduction in image quality.
Additionally, the training function was changed to incorporate multi-threading when calculating partitions. It can be seen in equation 1 that the issue of partitioning a training set of size N is a $N(2^{2R})$ computing problem. The search requires checking distortion against each centroid which in turn requires looking at each centroid again (hence why the 2^R is squared). Multithreading would allow multiple cores in a CPU to be used when training which divides the total time taken for the partition step by the amount of CPU cores available. This allows iterations to be run for less time and decreases the distortion of the quantizers.

6 Proposed Solution and Final Results

The following section described the designed and modelled end-to-end system along with final results.

6.1 Description

The final solution is a set of communication systems which have been optimized for multiple levels of channel noise. The systems have been designed for binary symmetric channels (BSC) with transition probabilities ranging from 0 to 0.1. The appropriate system can be deployed given the reliability of the channel.

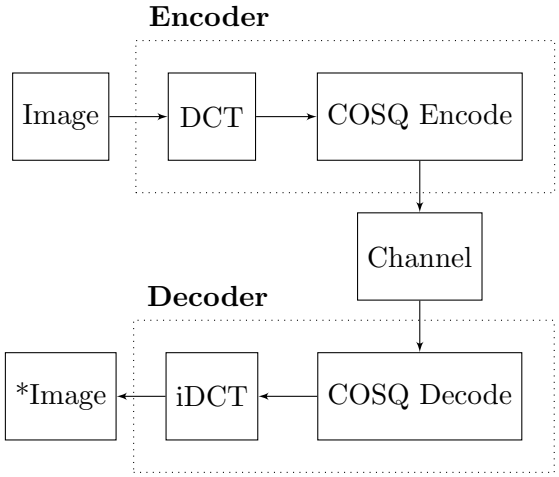


Figure 7: Block diagram of communication system

A high-level block diagram for the proposed communication system is shown above in Figure 7. The designed system is split into three parts, the encoder, channel and decoder. The encoder is implemented on the transmission (Tx) side and involves DCT followed by COSQ encoding to encode the image. Conversely, the decoder is implemented on the receiving (Rx) side and involves COSQ decoding followed by inverse DCT to recover the image. A gray scale image (1 byte per pixel) is split into 8x8 pixel blocks and passed to the encoder, then the image is transmitted wirelessly over the noisy channel, finally the receiving side decodes and reconstructs the image.

The properties of the channel arise from physical phenomenon and cannot be modified, thus the channel has been simplified as a memoryless BSC (even though most natural physical channels do exhibit some form of memory). This simplification allowed design focus to be placed on optimizing the encoder/decoder design. Within the encoder and decoder, most design decisions and optimizations are centered around the COSQ design. The modifiable parts of the design are split into two sub categories, training methods and coding parameters.

6.2 Training Methods

6.2.1 Training Data

In order to use the Generalized Lloyd-Max Algorithm to train the COSQs, training data must be acquired. Images put through DCT omit innate distributions according their index within the resulting matrix. The upper left (or Direct Current (DC)) entry omits a Gaussian distribution while all other entries (Alternating Current (AC)) omit Laplacian distributions [3]. Utilizing this fact, with proper normalization, training data can be generated according to these distributions. This results in COSQs that result in acceptable distortion for generalized training sets.

The second method involves using a training set of images for training data. This results in targeted optimization, with the system performing better on images which are similar to those in the training set. Training sets with high diversity and a large size will closely approximate the distribution generation described above.

To train the COSQs the training set method was chosen since the wildlife cameras will only be used in certain applications thus the subset of possible images will be limited to outdoor images of wilderness and animals. To closely model this, a dataset of 200 portable gray map images of animals was chosen. This data set features many different types of animals outdoors and has good feature diversity, such as different terrains, number of animals, brightness etc. (see Figure 8).The feature diversity results in a robust training set and will allow the communication system to perform well when deployed in varying locations. The Portable Gray Map (PGM) is simple and allows for easy parsing of images since the data is stored bytewise with each byte representing a pixel value (source).

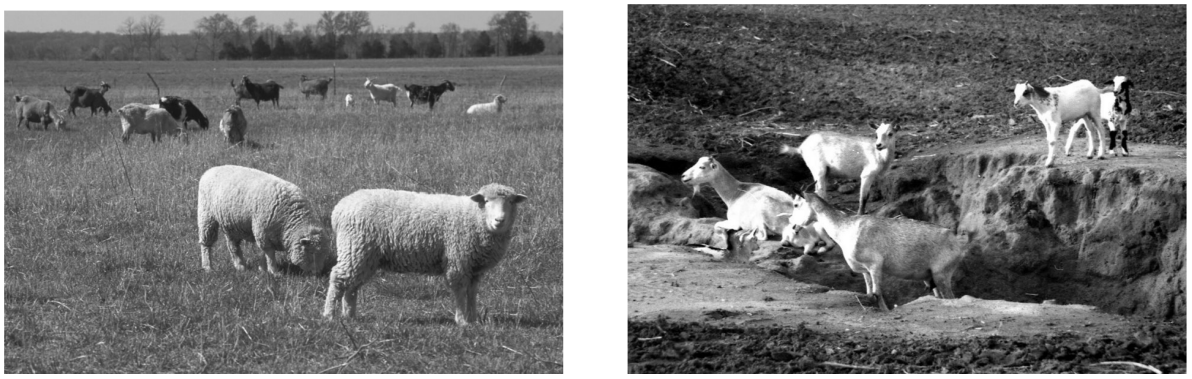


Figure 8: Sample images from Oregan State University Hoofed Animal Dataset [1]

6.2.2 Training and Stopping Criteria

Training the quantizers using the Generalized Lloyd-Max algorithm is a computationally expensive problem, which effects the feasibility of certain stopping criteria when utilizing conventional personal computing equipment. In order to have sufficient stopping criteria to approximate locally optimal solutions, efficient computing algorithms are essential to limit training time. Multi-threaded parralelization and cycle optimization were used extensively to limit computation time (see Section 5.4 on iterations for in-depth overview of optimizations). The stopping criteria was based on relative distortion changes between iterations. Let d_i represent the distortion at iteration i, and let R represent the rate of the quantizer, then the stopping criteria is given by,

$$\frac{|d_{i-1} - d_i|}{d_i} \leq 0.0005(R) \tag{4}$$

The stopping criteria changes based on the rate of the code, since the computation problem is in the order of 2^{2R}, the stopping criteria is softer for higher rate quantizers, to limit training time without overly effecting quantizer optimality.

6.3 Coding Parameters

There are two principle coding parameters that control the performance of the communication system, code rate and bit allocation. Code rate refers to the amount of bits allocated per 8x8 block of pixels. Bit allocation is how these bits are allocated to different quantizers within the 8x8 block. Models were trained for 24, 58 and 76 Bit systems with the following bit allocation matrices shown in Table 4. The bit allocation matrices were optimized for the given bit rate and memoryless BSC transition probability in [3].

ϵ	**24 bit**	**58 bit**	**76 bit**
0	6 4 3 1 0 0 0 0 3 2 2 0 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	7 5 4 3 2 1 0 0 4 4 3 3 2 1 0 0 3 3 3 2 1 0 0 0 2 2 2 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	7 6 4 4 3 2 1 0 5 4 4 3 2 2 0 0 3 3 3 3 2 1 0 0 2 2 2 2 2 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0.005	8 4 2 1 0 0 0 0 3 2 2 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	8 7 4 3 2 1 0 0 8 7 4 3 2 1 0 0 3 3 2 2 1 0 0 0 2 2 2 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	8 8 5 3 3 2 0 0 6 5 3 3 2 1 0 0 3 3 3 2 2 1 0 0 2 2 2 2 1 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0.01	8 5 2 1 0 0 0 0 3 2 1 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	8 8 5 3 2 1 0 0 8 8 5 3 2 1 0 0 3 3 2 2 1 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	8 8 5 3 2 2 0 0 7 5 3 3 2 1 0 0 3 3 3 2 2 1 0 0 2 2 2 2 1 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0.005	8 8 0 0 0 0 0 0 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	8 7 6 4 0 0 0 0 7 6 5 0 0 0 0 0 6 5 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	8 7 6 4 3 0 0 0 7 6 5 4 0 0 0 0 6 5 4 0 0 0 0 0 4 4 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0.1	8 8 2 0 0 0 0 0 4 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	8 8 8 3 1 0 0 0 8 8 8 3 1 0 0 0 3 3 2 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	8 8 8 4 2 1 0 0 8 8 6 4 1 0 0 0 4 4 4 1 1 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

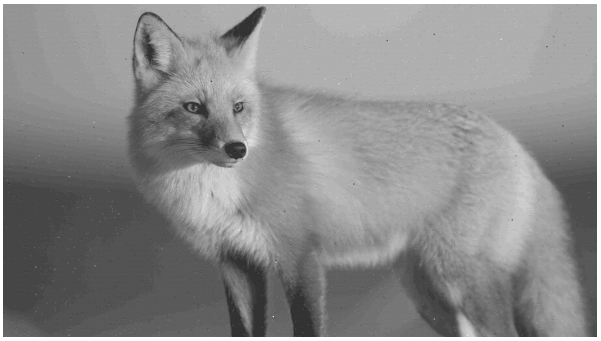
Table 4: Table of 8x8 bit allocation matrices used to train different quantizers [3]

6.4 Results

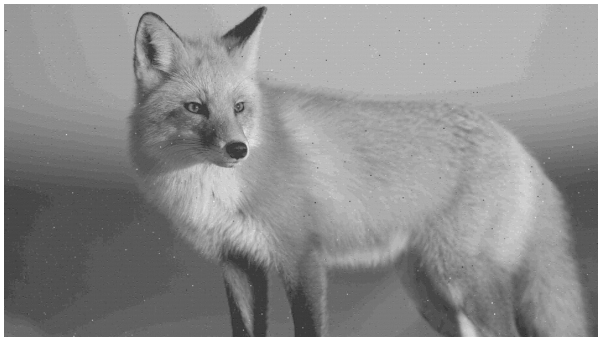
The following are end-to-end using the baseline fox image for multiple different epsilon and bit rates.

(a) Original fox image

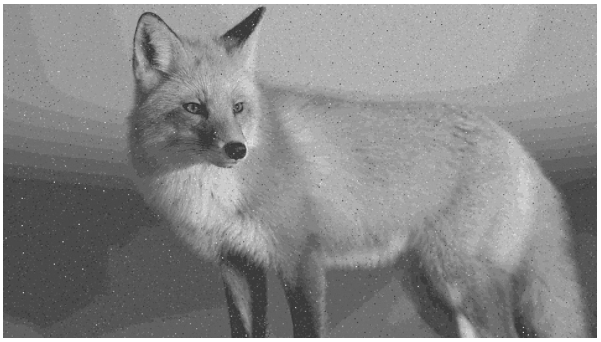
(b) $\epsilon = 0$, $PSNR = 27.31$ dB



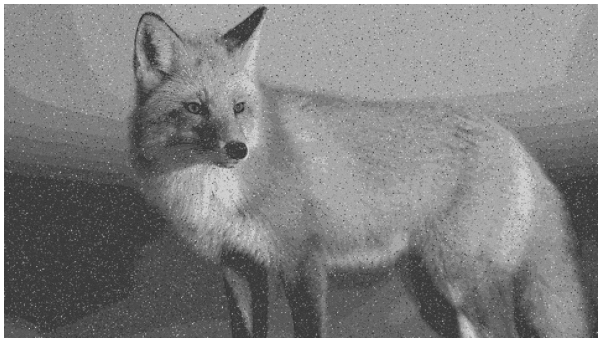
(c) $\epsilon = 0.005$, $PSNR = 26.20$ dB



(d) $\epsilon = 0.01$, $PSNR = 25.20$ dB



(e) $\epsilon = 0.05$, $PSNR = 24.92$ dB



(f) $\epsilon = 0.1$, $PSNR = 23.10$ dB

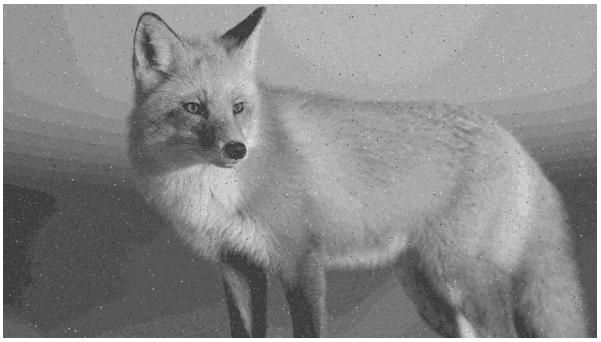
Figure 9: Sample images for 0.375 BPP (24 bit allocation matrix)

(a) Original fox image

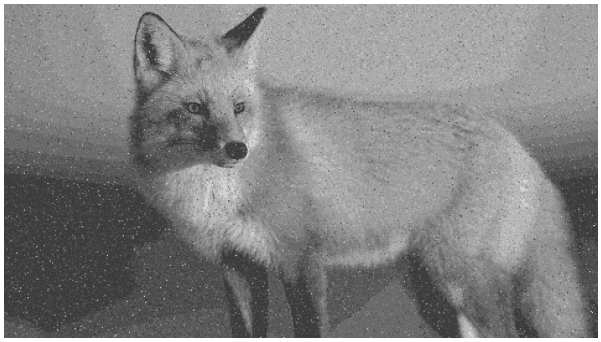
(b) $\epsilon = 0$, $PSNR = 28.72$ dB

(c) $\epsilon = 0.005$, $PSNR = 26.71$ dB

(d) $\epsilon = 0.01$, $PSNR = 25.1$ dB



(e) $\epsilon = 0.05$, $PSNR = 24.32$ dB



(f) $\epsilon = 0.1$, $PSNR = 23.03$ dB

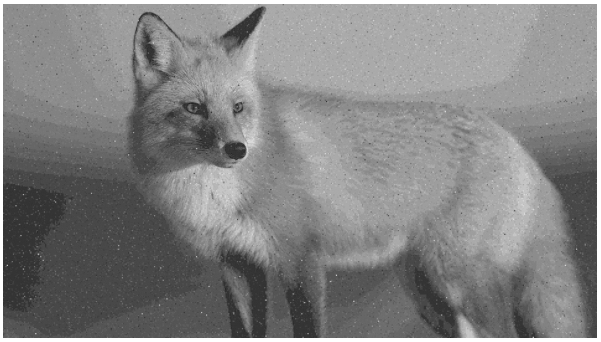
Figure 10: Sample images for 0.9 BPP (58 bit allocation matrix)

(a) Original fox image

(b) $\epsilon = 0$, $PSNR = 29.61$ dB

(c) $\epsilon = 0.005$, $PSNR = 27.06$ dB

(d) $\epsilon = 0.01$, $PSNR = 25.60$ dB

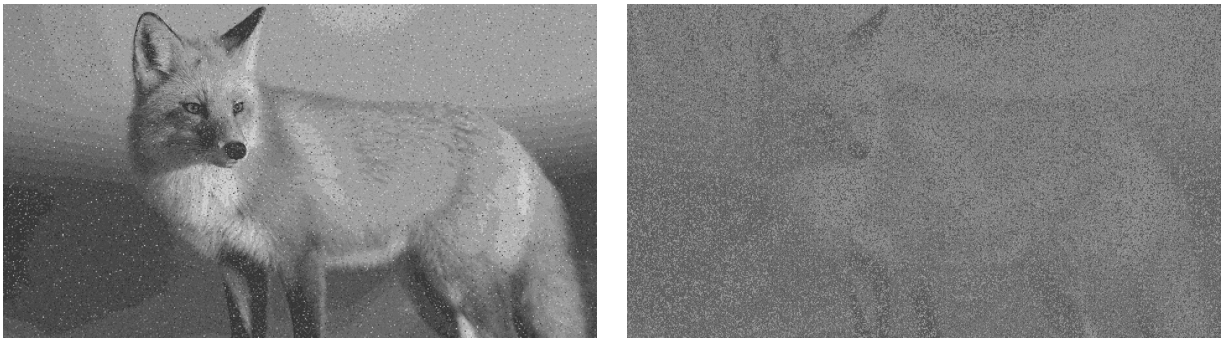


(e) $\epsilon = 0.05$, $PSNR = 24.71$ dB

(f) $\epsilon = 0.1$, $PSNR = 22.48$ dB

Figure 11: Sample images for 1.19 BPP (76 bit allocation matrix)

For all three bit rates it is clear that quality decreases as the BSC transition probability increases. Mathematically, this is seen through the decreasing PSNR but can also be seen visually as the images appear noisier. Even at high ϵ, the image remains clear enough to recognize the animal and distinct features, and the PSNR remains sufficiently high. For the purposes of the wildlife camera, the main criteria for image quality is that the animal features remain identifyable to human perception after being distorted through the channel. Given this criteria, the communication system remains robust enough to handle very noisy channels (up to $\epsilon = 0.1$) while still maintaining a high PSNR and image features.



(a) 76-Bit, trained with $\epsilon = 0.1$, PSNR = 22.48 dB (b) 76-Bit, Trained with $\epsilon = 0$, PSNR = 15.62 dB

Figure 12: Comparison of images sent through channel optimized vs unoptimized scalar quantizer systems (using BSC with $\epsilon = 0.1$)

The results shown in Figure 12 serve as a sanity check and demonstrate the benefits gained from taking the channel characteristics into account when training the quantizers. The 76-bit channel optimized system has a 6.86 dB larger PSNR than its unoptimized counterpart. Visually, the right image in Figure 12 is almost unrecognizable which is unacceptable for the application which requires clear feature recognition.

6.4.1 End-to-End PSNR Discussion

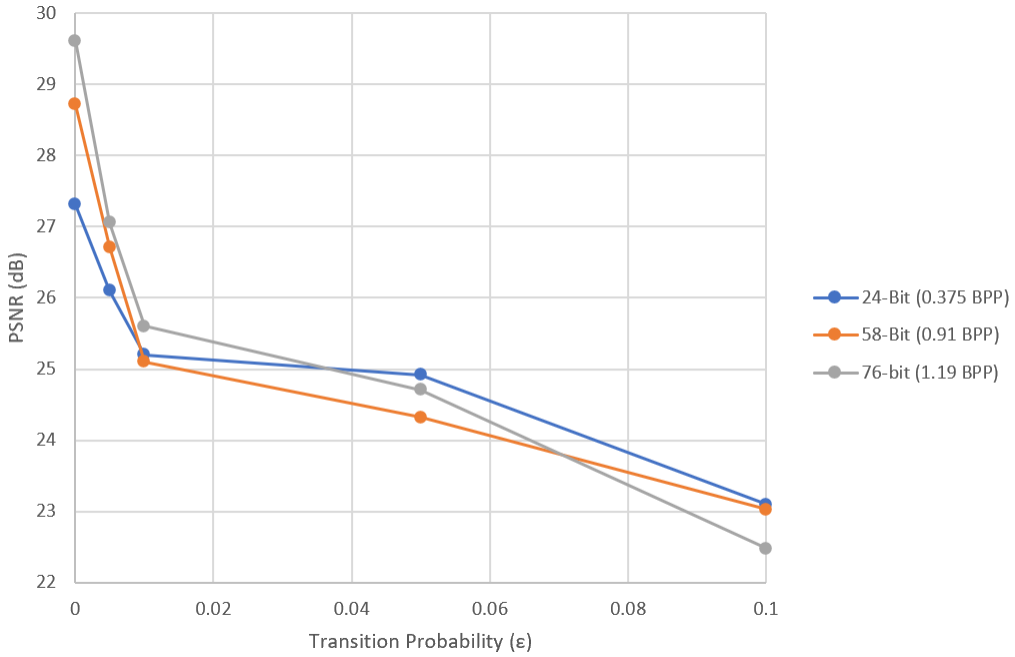


Figure 13: PSNR at different BSC Transition Probabilities

The results shown in Figure 13 show the PSNR of the End-to-End systems as the transition probability increases. As expected, as the channel becomes noisier, PSNR decreases. The fact that the 24-bit system starts to outperform both higher bit systems is unexpected and concerning. In

sources such as [3], increasing bit rate per 8x8 block always increases PSNR when the transition probability of the channel the remains constant. This most likely indicates that there is either an issue with the model training program or that there is an issue in the communication system code. An investigation in needs to be conducted in future work to discover the root cause of this issue.

6.4.2 Quantizer Distortion Discussion

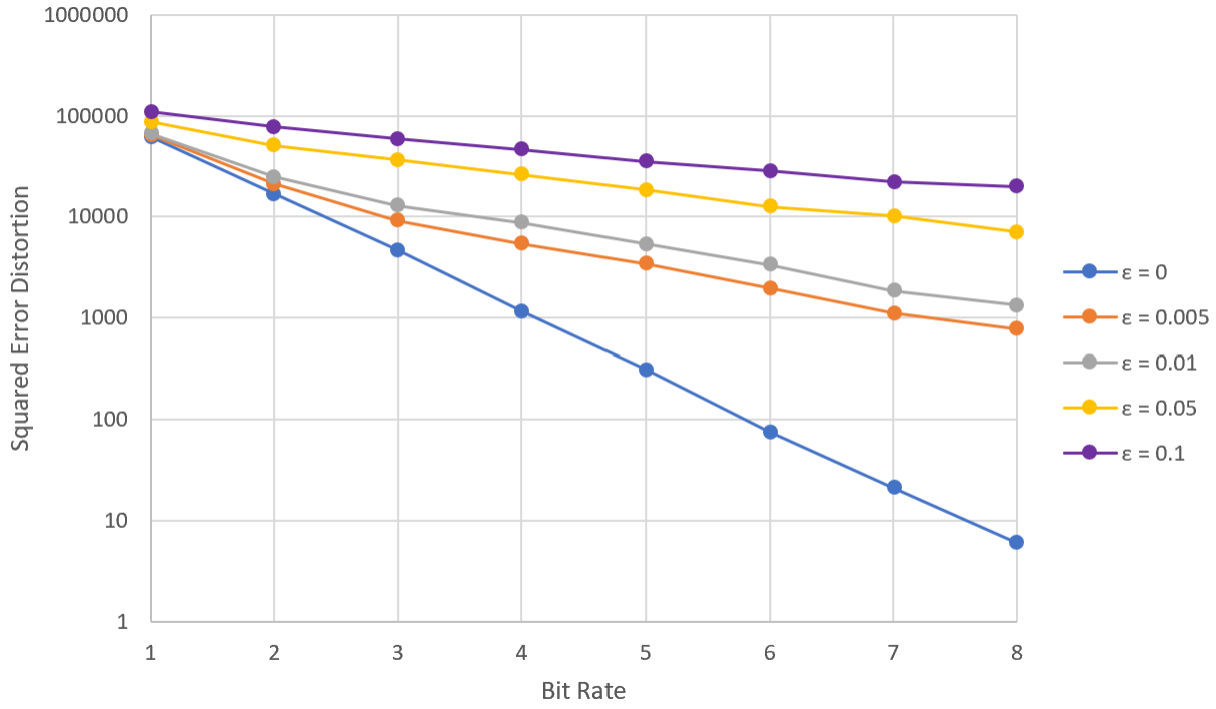


Figure 14: Distortion-rate curves for multiple BSC transition probabilities (ϵ)

The results shows in Figure 14 show the distortion-rate curves for COSQs trained on DC (upper left corner) data from the resulting matrix data after DCT. The curves are all monotonically decreasing as expected from rate-distortion theory. This graph demonstrated improvement over the under-training issues seen in earlier iterations of the design (see Figure 4). The regular, monotonically decreasing nature of the curves in Figure 14 demonstrate that the stopping criteria was sufficient to allow the quantizers to be approximately optimal.

6.4.3 Performance Comparison to Baseline

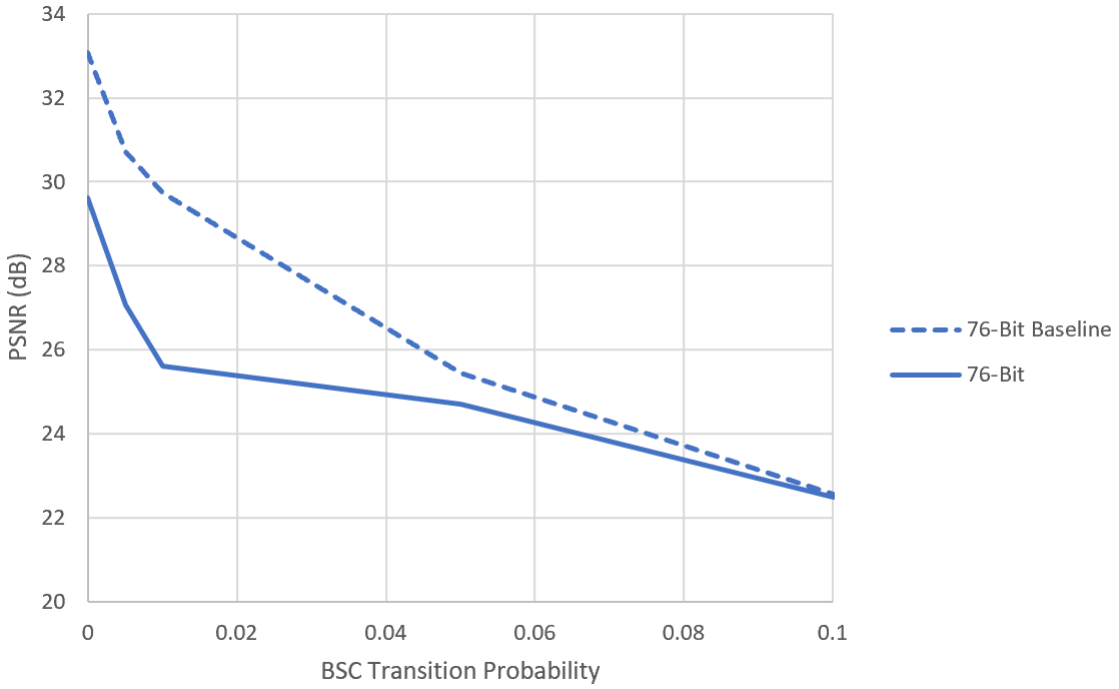


Figure 15: PSNR comparison to baseline for different BSC transition probabilities

As discussed in Section 5.2, the performance of the system was measured against the implementation of a similar system in [3]. Figure 15 shows the performance of the system against the baseline. It is clear that the designed system performs worse than the baseline when the transition probability of the channel, but as noise increases the system approaches the performance of baseline. The discrepancies at lower transition probabilities are most likely due to similar issues that are causing the strange behavior of end-to-end PSNR as seen in Figure 13 and discussed in Section 6.4.1. It should also be noted that the discrepancies are partially due to different images being used to measure performance.

6.5 Recommended Solution

The trade off in communication system is typically between transmission rate and PSNR, where high PSNR comes at the cost of low transmission rates. In the case of the proposed designs, the 0.375 bit per pixel (BPP) system outperforms the higher BPP systems in channels with transition probability $\geq$ 0.05. Although this is most likely due to design errors, as discussed in Section 6.4.1, the high compression ratio of $8/0.375 = 21.3$ makes the choice of the 24-bit systems obvious for now since the system offers high PSNR with higher transmission rates than the higher BPP systems.

Once further investigation occurs regarding the errors, the recommended system will most likely depend on user needs. Noisier channels will benefit from higher BPP systems to maintain high PSNR and feature recognition. Less noisy channels may have acceptable PSNR and feature recognition with lower BPP systems. The simplicity of the models after being trained will allow different options to be selected depending the deployment scenario.

6.6 Hardware Integration

The generated partition mapping could be loaded into whatever hardware deemed appropriate by the client. Currently, our quantizer software is implemented in python. Python has the advantage of being a widely implemented language, so unless the customer were looking to to load the quantizer into custom PCB's we could integrate the code into most microprocessors on the market.

Implementing the encoder and decoder in a hardware specific programming language would be straight forward now that the quantizer models have been trained. On the encoder side all that is needed is DCT and a lookup in the stored hashed partition map. DCT is matrix multiplication and the lookup requires minimal code. On the decoder side, only inverse DCT and index matching is necessary.

6.7 Economics of Solution

Encoding/decoding time takes a few minutes on full computer hardware as it currently stands. Depending on the hardware deemed necessary to complete the encoding on the camera, this could significantly increase the end cost per unit. The customer would always be welcome to accept latency in encoding/transmission time as a trade off to cheaper hardware. The encoding algorithm is not particularly complex, so should see a peak memory load of about 40kB (size of the partition hash map) + a negligible amount for source code. The bulk of the space in main memory would be taken up by the image which could be upwards of 10 MB in size. This would mean the software would be capable of running on a machine with 10 MB of RAM minimum, which would be a fairly basic and cheap system.

6.8 Triple Bottom Line

Implementing a camera in a remote area needs to be non-invasive because the goal is to blend in, so the animals do not recognize the change in the environment and get scared off. Also, the primary stakeholder is wildlife conservationists, so it is paramount that the installation and maintenance of the camera have minimized negative environmental effects. To ensure this, careful consideration needs to be given to the materials used in the camera's construction, such as biodegradable or recycled materials, to minimize the impact on the environment. Additionally, the camera's power source needs to be sustainable, such as using solar panels, to avoid leaving behind any environmental footprints.

One of the most critical aspects of developing a wildlife camera is to make sure that it is non-invasive, meaning that it does not negatively impact the environment or disrupt the animals' natural habitat. As such, the camera's design must take into account the materials used in its construction. For instance, the use of biodegradable or recycled materials could help reduce the environmental impact of the camera, as these materials are less likely to pollute the environment or harm the wildlife that interacts with them.

Moreover, the camera's power source must be sustainable, as traditional sources of energy such as fossil fuels can have severe environmental impacts. A feasible alternative is the use of solar panels, which are both eco-friendly and reliable. With advances in solar technology, a camera powered by solar panels can be used for extended periods without worrying about battery life, thus reducing the need for frequent maintenance trips that may disturb the animals in the area.

Wildlife cameras could give hunters a competitive edge, making them able to track and hunt wildlife much easier. To combat this, the camera must be advertised and sold to environmentalists to limit the chances of misuse for the opposite reason the camera is being designed for. Another approach to prevent misuse could be to implement software that detects the presence of firearms in the image and automatically alerts the appropriate authorities to prevent illegal hunting activities.

One of the most significant risks associated with wildlife cameras is their potential for misuse. For instance, they could be used by hunters to gain an unfair advantage in their hunting activities. To prevent this, cameras must be marketed and sold exclusively to environmentalists and other groups that have a legitimate need for them. Additionally, software that can detect the presence of firearms in the image could be incorporated into the camera to alert the appropriate authorities in case of illegal hunting activities. This would help protect the animals from undue harm and prevent illegal hunting activities from taking place.

The conservation company is nonprofit, so keeping costs down is a priority. As a result, trade-offs will have to be made mainly between product cost and efficiency. If the team were to increase the bit allocation rate more, the image would be improved, but the cost to do so would not be worth it. The money could be better spent on increasing battery life, so the camera can last longer resulting in fewer maintenance trips and interruptions. Furthermore, reducing the camera's size and weight would also help minimize costs by reducing the need for expensive transportation and installation equipment.

Developing a wildlife camera can be an expensive venture, especially when considering the costs associated with design, production, and distribution. As such, conservation companies must balance the cost of production with the product's efficiency to keep costs down. This might involve making trade-offs between features, such as image quality and battery life, to achieve the desired balance between cost and efficiency. In this regard, reducing the camera's size and weight can help minimize costs by reducing the need for expensive transportation and installation equipment.

Additionally, a longer battery life can reduce the need for frequent maintenance trips, which can be expensive and time-consuming. Therefore, conservation companies must prioritize cost-effectiveness in their product design to maximize the benefits of their wildlife camera technology.

7 Future Work

7.1 Increased Reconstruction Probability of First DCT Entry

One potential avenue for improvement to our codec would be to increase the accuracy of the transmitted first entry in the DCT matrix over the channel. The first entry in the DCT matrix tends to carry the most important information in recreating the original image matrix, while the subsequent entries hold information about smaller details. If the first entry of the DCT matrix could be reconstructed without error this could add a degree of reliability that the reconstructed image will be recognizable by the human eye. The use of variable length lossless codes is not feasible since errors within the channel could result in error propagation on the receiving side [3].

Some options for increasing the transmission accuracy of the first DCT entry could be:

- Apply a fixed length error correcting code to the entry, such as Reed Solomon

- Use an even higher resolution quantizer on the first entry

Of course the tradeoff of this route would be a lower compression rate, but also a lower distortion.

7.1.1 Vector Quantization

Our current codecs use channel optimized scalar quantization on the DCT values of each block of the image. One potential improvement on this setup could be to instead use vector quantization on several values within the DCT blocks or the DCT block as a whole. Given the entries in the DCT blocks should have some statistical relationship (as they are a transform on image blocks which do), this redundancy is not being taken advantage of in our current quantizer setup.

7.1.2 Adding Channel Memory

Another potential improvement to our encoding/decoding mechanism would be to use a channel model that incorporates memory. As channel errors tend to occur in bursts, and subsequent transmissions of image data tend to be correlated, modelling the channel we optimize our quantizer off of using memory could significantly increase the fidelity of the reconstructed images. A potential model to represent a radio channel includes the Gilbert-Elliot channel (GEC) [4].

7.2 Larger Training Set

This improvement is fairly straightforward - a larger training set would expect a closer to true source distribution training sample. By this method, our channel optimized quantizers will be better tailored to our source distribution and we should see a lower overall expected distortion.

7.3 Experiment with Gray Coding

The index mapping currently uses natural ordering of binary numbers. It has been shown in sources such as [5] show that Gray ordering of the indexes, which limits Hamming distance of adjacent indices, can perform better than natural ordering in very noisy channels.

References

[1]

[2] Allen Gersho and Robert M. Gray. Vector quantization and signal compression. In *The Kluwer International Series in Engineering and Computer Science*, 1991.

[3] Julian Zhishen Cheng. *Channel optimized quantization of images over binary channels with memory*. PhD thesis, National Library of Canada = Bibliotheque nationale du Canada, 1998.

[4] Libo Zhong, Fady Alajaji, and Glen Takahara. A binary communication channel with memory based on a finite queue. *IEEE Transactions on Information Theory*, 53(8):2815–2840, 2007. doi: 10.1109/TIT.2007.901184.

[5] A. Kurtenbach and P. Wintz. Quantizing for noisy channels. *IEEE Transactions Communication Technology*, 17, 1969.