Justin Koh
250837810
CS4481A
Professor El-Sakka

Assignment 3 Written Report

Peppers.pgm

Buffer size: 256

Figure 1

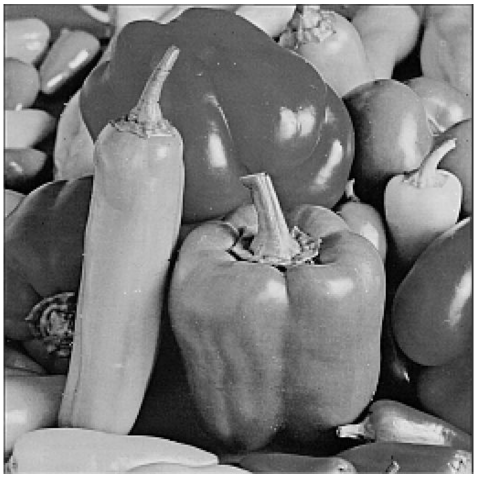


Figure 1 shows the peppers.pgm file after encoding and decoding using my LZ77 implementation using a buffer size of 256. It is similar to the original peppers.pgm file with very minor differences such as there being 2 pixels off color on the big pepper in the middle. The mean average error for this image is 0.000000.

Note: My MAE for each subsequent image in this assignment was calculated as 0.000000. There are two explanations that I have for this value: either I had improper implementation of the function which always returns a value of 0, or I am speculating that since it is a net sum of pixel values, any shifts in the pixels across the image would result in a MAE even though there are “visual” differences in the images.

Additional note: I could not get Google Sheets to show the y-axis (frequency) for each chart to start at 0, as a result some values seem to not be there, but this is because those match lengths/offsets frequencies are 1, and thus do not render on the graph; however, they are there.

Figure 1.1

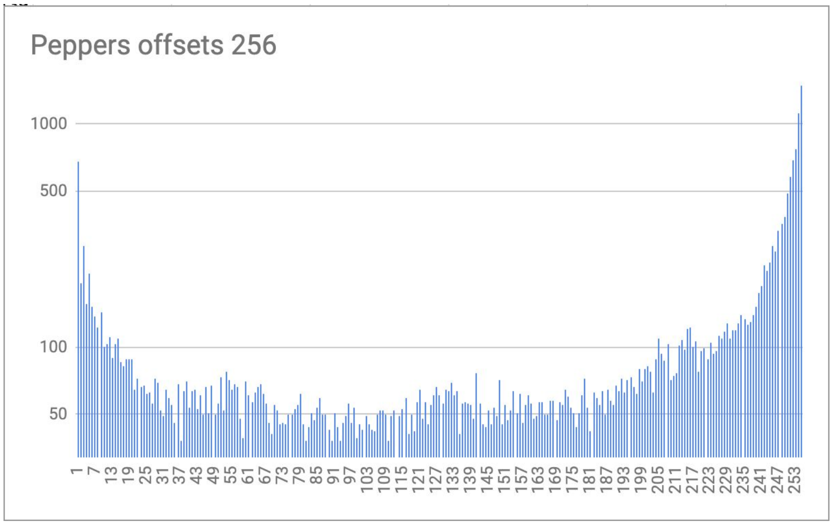
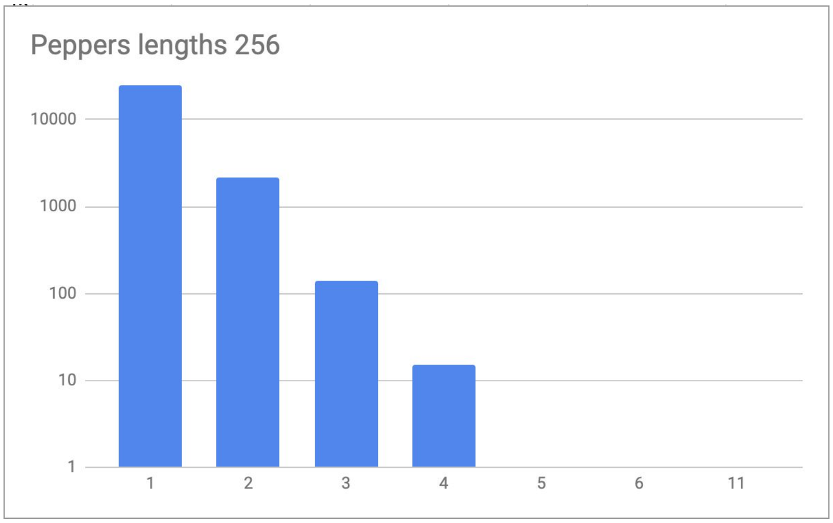


Figure 1.2



The above graphs show the frequency of token offset values (Figure 1.1), and the frequency of token match length values (Figure 1.2) for peppers.pgm using a buffer size of 256. There are noticeable spikes in Figure 1.1 at the start and end of the buffer. My speculation is that there is a pattern in the image that matches along the beginning values and end values of the buffer as it slides along during encoding. Figure 1.2 shows a downward trend in length frequency, probably due to there being a higher probability of there being a small match, rather than a match of long length.

Buffer size: 1024

Figure 2

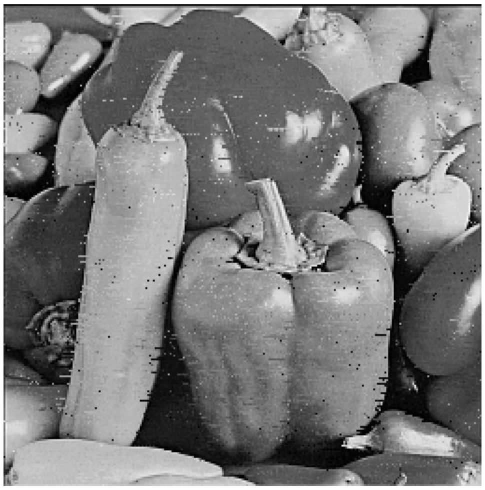


Figure 2 shows the peppers.pgm file after encoding and decoding using my LZ77 implementation using a buffer size of 1024. It is similar to the original peppers.pgm file with more significant differences such as there being “smeared” artifacts causing a tv static-like effect on the produced image. The mean average error for this image is 0.000000.

Figure 2.1

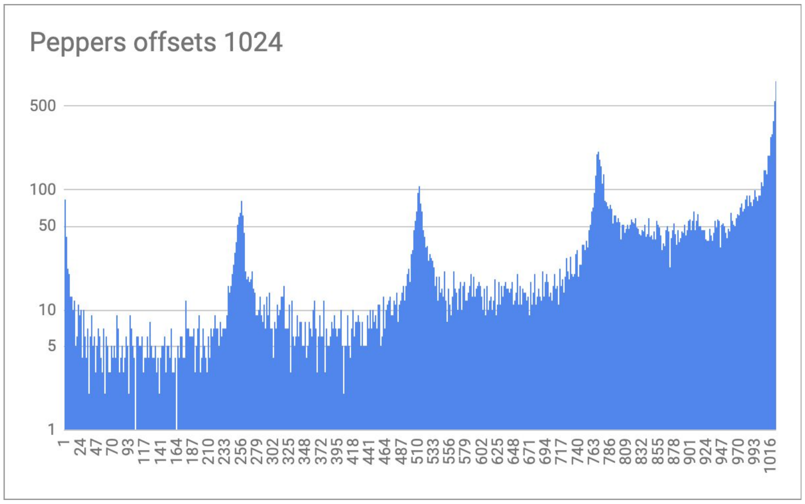
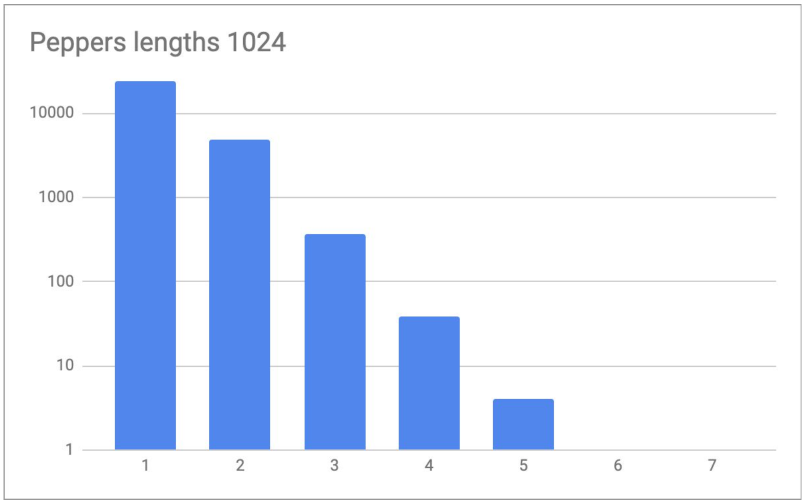


Figure 2.2



The above graphs show the frequency of token offset values (Figure 2.1), and the frequency of token match length values (Figure 2.2) for peppers.pgm using a buffer size of 1024. There are noticeable spikes in Figure 2.1 at distinct values in the buffer. My speculation is that the pattern in the previous Figure 1.1 is being repeated in a larger buffer size. This is because 1024 is a multiple of 256, causing the same image pattern to be repeated 4 times as frequently as the buffer slides along the image. Figure 2.2 also shows a downward trend in length frequency, for the same reasons as described earlier.

Buffer size: 5120

Figure 3

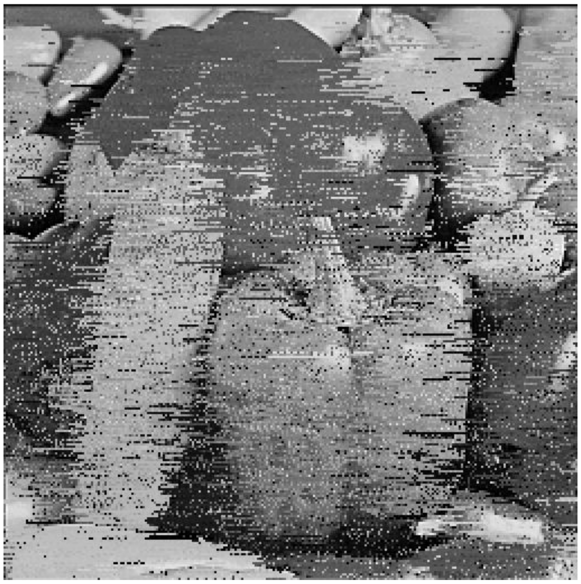


Figure 3 shows the peppers.pgm file after encoding and decoding using my LZ77 implementation using a buffer size of 5120. Comparing it to the original peppers.pgm file, it contains significant differences having “smeared” artifacts causing a more prominent tv static-like effect on the produced image. Although this is not the desired result, it still has structure found in the original image. The mean average error for this image is 0.000000.

Figure 3.1

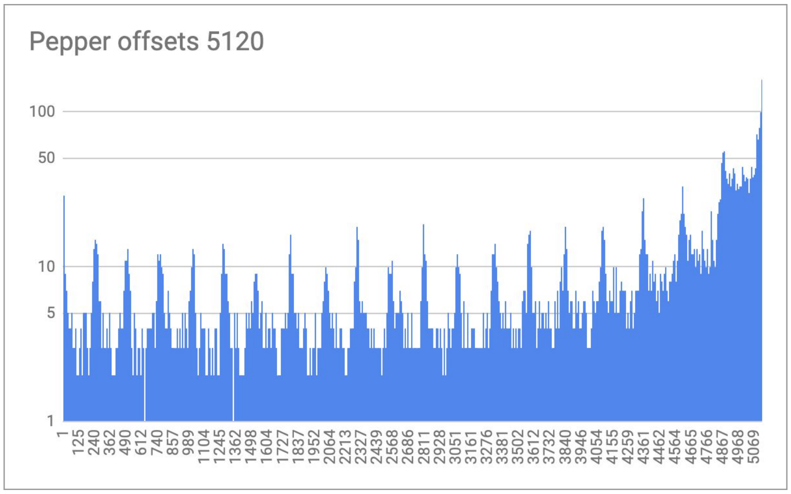
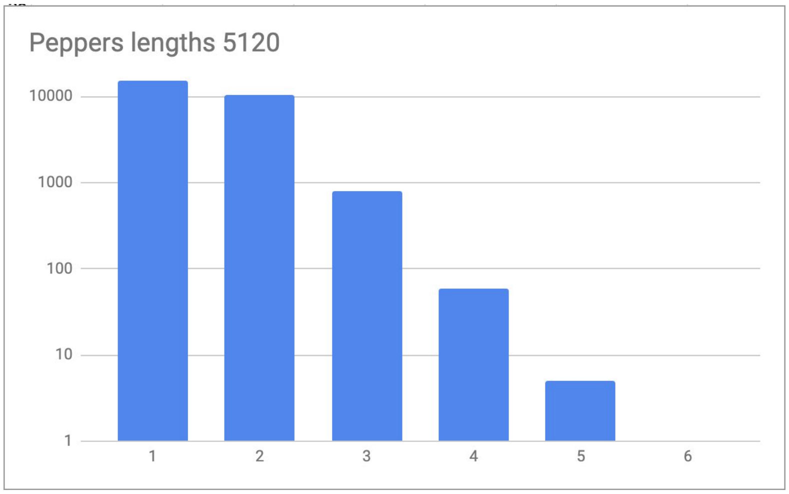


Figure 3.2



The above graphs show the frequency of token offset values (Figure 3.1), and the frequency of token match length values (Figure 3.2) for peppers.pgm using a buffer size of 5120. There are noticeable spikes in Figure 2.1 at distinct values in the buffer. My speculation is that the pattern in the previous Figure 2.1 is being repeated in a larger buffer size. This is because 5120 is a multiple of 256, causing the same image pattern to be repeated 20 times as frequently as the buffer slides along the image. Figure 3.2 also shows a downward trend in length frequency, for the same reasons as described earlier.

Goldhill.pgm

Buffer size 256

Figure 4

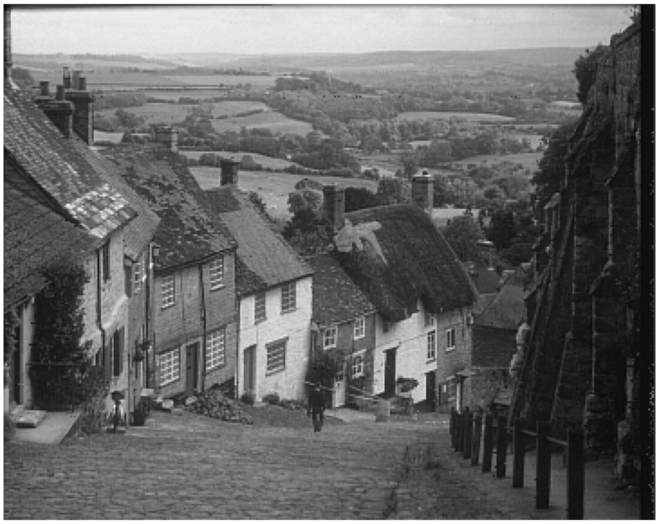


Figure 4 shows the goldhill.pgm file after encoding and decoding using my LZ77 implementation using a buffer size of 256. It is similar to the original goldhill.pgm file with very minor differences such as there being pixels off color on the bottom left side of the image. The mean average error for this image is 0.000000.

Figure 4.1

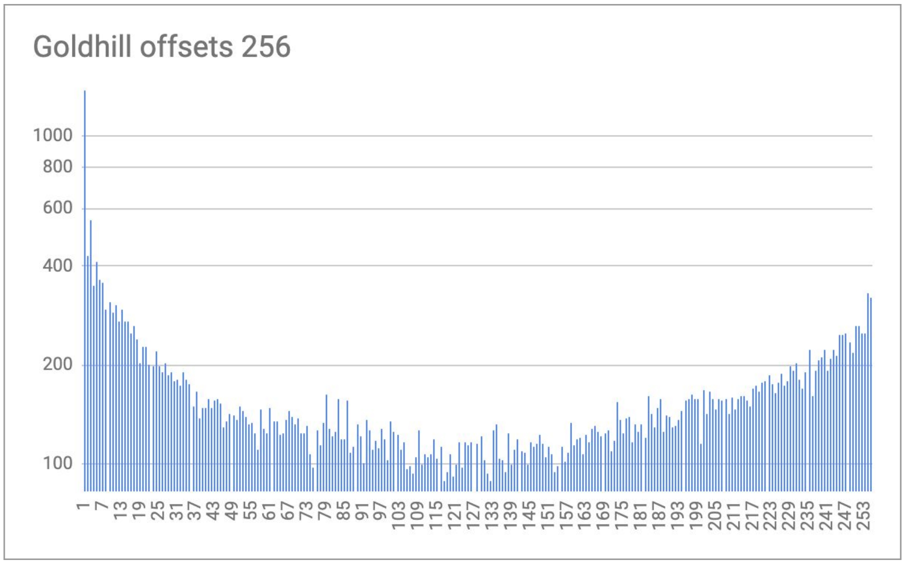
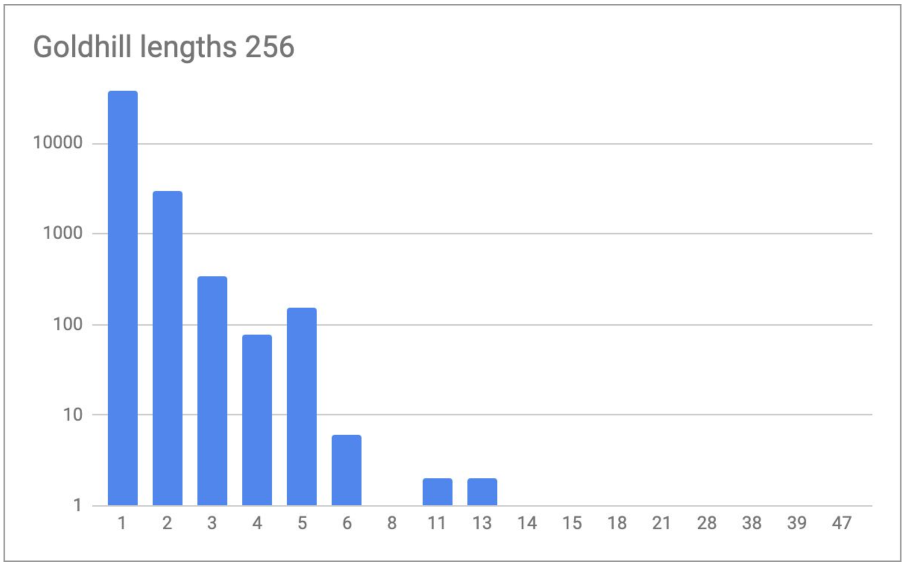


Figure 4.2



The above graphs show the frequency of token offset values (Figure 4.1), and the frequency of token match length values (Figure 4.2) for goldhill.pgm using a buffer size of 256. There are noticeable spikes in Figure 4.1 at distinct start and end values in the buffer. This is probably due to a pattern being repeated as the buffer slides along the image. Figure 4.2 also shows a downward trend in length frequency, for the same reasons as described earlier.

Buffer size 1024

Figure 5

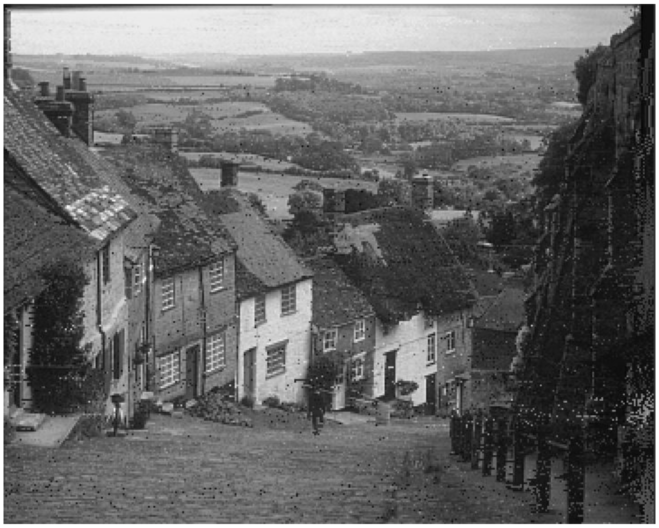


Figure 5 shows the goldhill.pgm file after encoding and decoding using my LZ77 implementation using a buffer size of 1024. It is similar to the original goldhill.pgm file with minor differences such as there being visible inconsistencies in certain parts of the image. The mean average error for this image is 0.000000.

Figure 5.1

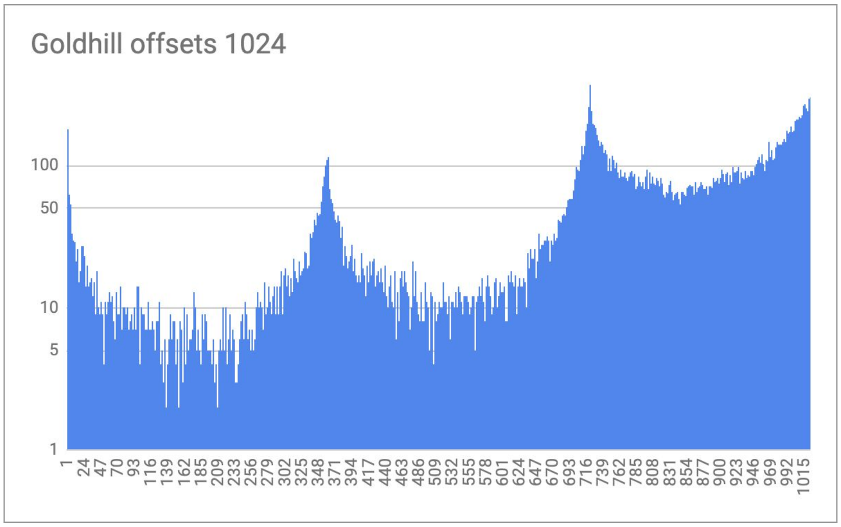
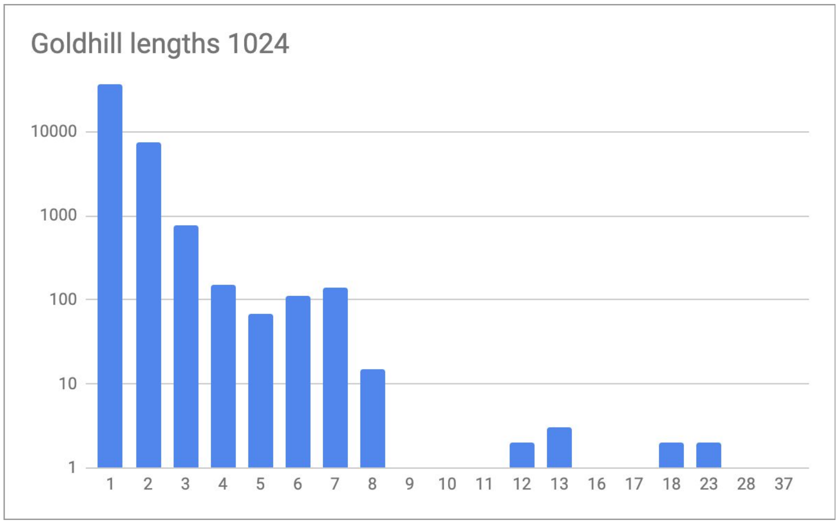


Figure 5.2



The above graphs show the frequency of token offset values (Figure 5.1), and the frequency of token match length values (Figure 5.2) for goldhill.pgm using a buffer size of 1024. There are noticeable spikes in Figure 5.1 at various offsets. This is probably due to a pattern being repeated 4x more frequently as the buffer slides along the image. Figure 4.2 also shows a downward trend in length frequency, for the same reasons as described earlier.

Buffer size 5120

Figure 6

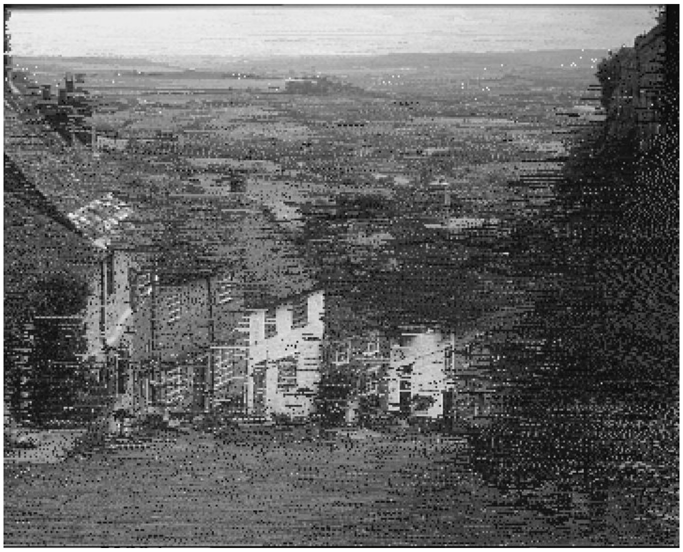


Figure 6 shows the goldhill.pgm file after encoding and decoding using my LZ77 implementation using a buffer size of 5120. It has the structure of the original goldhill.pgm file with “smearing” static-like artifacts throughout the image. The mean average error for this image is 0.000000.

Figure 6.1

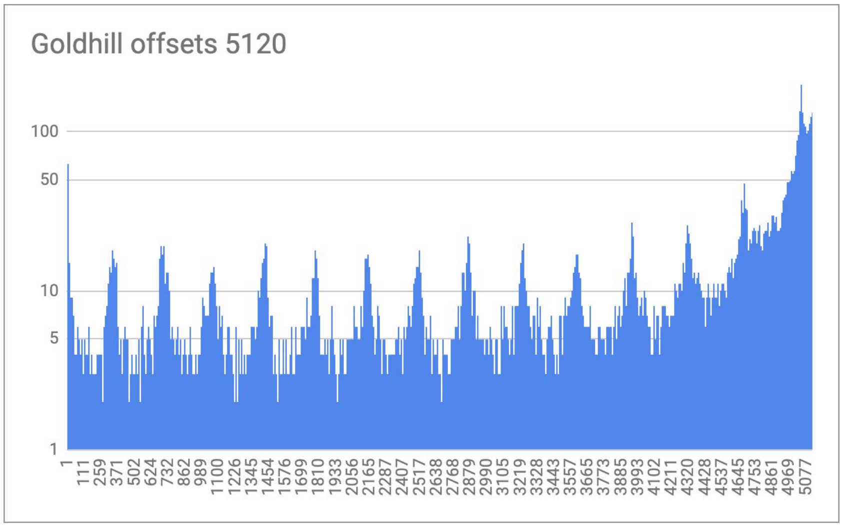
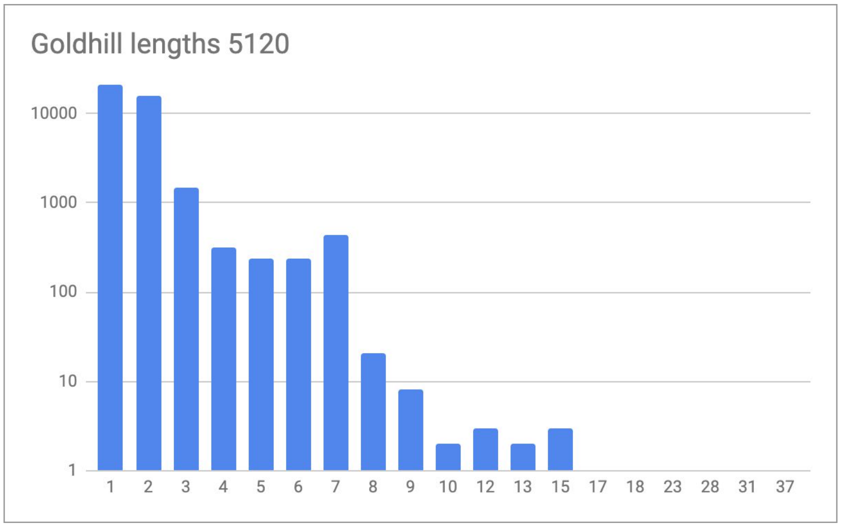


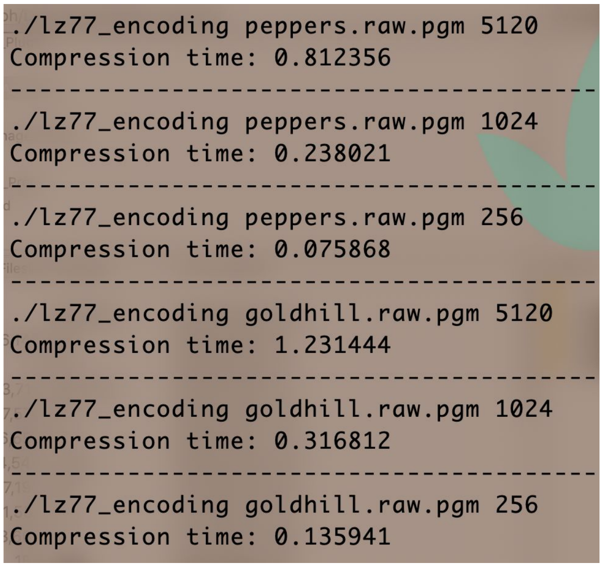
Figure 6.2



The above graphs show the frequency of token offset values (Figure 6.1), and the frequency of token match length values (Figure 6.2) for goldhill.pgm using a buffer size of 5120. There are noticeable spikes in Figure 6.1 at various offsets throughout the image that are 20 times as frequent as Figure 4.1. This is probably due to the same pattern being repeated 20x more frequently as the buffer slides along the image. Figure 6.2 also shows a downward trend in length frequency, for the same reasons as described earlier; however with higher frequency values than the previous cases.

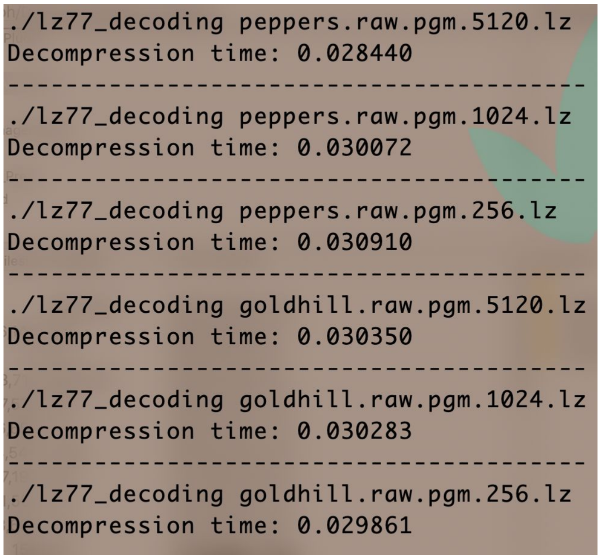
From my results, I believe it would be best to use a buffer size that is a multiple of the image width.

Compression times:



```
./lz77_encoding peppers.raw.pgm 5120
Compression time: 0.812356
----------------------------------------
./lz77_encoding peppers.raw.pgm 1024
Compression time: 0.238021
----------------------------------------
./lz77_encoding peppers.raw.pgm 256
Compression time: 0.075868
----------------------------------------
./lz77_encoding goldhill.raw.pgm 5120
Compression time: 1.231444
----------------------------------------
./lz77_encoding goldhill.raw.pgm 1024
Compression time: 0.316812
----------------------------------------
./lz77_encoding goldhill.raw.pgm 256
Compression time: 0.135941
```

Decompression times:



```
./lz77_decoding peppers.raw.pgm.5120.lz
Decompression time: 0.028440
---------------------------------------
./lz77_decoding peppers.raw.pgm.1024.lz
Decompression time: 0.030072
---------------------------------------
./lz77_decoding peppers.raw.pgm.256.lz
Decompression time: 0.030910
---------------------------------------
./lz77_decoding goldhill.raw.pgm.5120.lz
Decompression time: 0.030350
---------------------------------------
./lz77_decoding goldhill.raw.pgm.1024.lz
Decompression time: 0.030283
---------------------------------------
./lz77_decoding goldhill.raw.pgm.256.lz
Decompression time: 0.029861
```

Averages and Standard Deviations: MacOSX

```
./lz77_encoding peppers.raw.pgm 5120
Offset average: 3749.968506
Offset standard deviation: 1569.399292
Match length average: 1.446780
Match length standard deviation: 0.584772
----------------------------------------
./lz77_encoding peppers.raw.pgm 1024
Offset average: 744.899231
Offset standard deviation: 295.999969
Match length average: 1.144465
Match length standard deviation: 0.494514
----------------------------------------
./lz77_encoding peppers.raw.pgm 256
Offset average: 126.870903
Offset standard deviation: 106.727806
Match length average: 0.827704
Match length standard deviation: 0.542163
----------------------------------------
./lz77_encoding goldhill.raw.pgm 5120
Offset average: 3778.482178
Offset standard deviation: 1569.199829
Match length average: 1.611127
Match length standard deviation: 1.026083
----------------------------------------
./lz77_encoding goldhill.raw.pgm 1024
Offset average: 733.092407
Offset standard deviation: 280.672760
Match length average: 1.205723
Match length standard deviation: 0.744318
----------------------------------------
./lz77_encoding goldhill.raw.pgm 256
Offset average: 88.324402
Offset standard deviation: 91.176445
Match length average: 0.809189
Match length standard deviation: 0.706543
```

Final notes: My decompressed images are extremely similar to the original images for buffer sizes using 256; however, as buffer size increases, the resulting images become more distorted. Although this is not the desired result, I have spent time trying to resolve the issue and have a couple ideas as to why this is happening:

1. My encoder is not quite right and is failing to select the token with the smaller offset when choosing between tokens of equal match length.
2. My decoder is not quite right and is causing a skewing effect when writing to the image array.

Because of this, the produced images, although skewed, are only offset by a certain number of pixels causing a patterned distortion throughout the image. I hope that my reflections and thoughts can earn me credits, as I spent a lot of time trying to get the encoding and decoding to work properly.