Comparative Study on Lossless Data Compression Techniques for Video files

[1]Ratanpara Pratik, [2]Limbasiya Kalpesh

Computer Engineering Department
VVP Engineering College, Gujarat Technological University, Rajkot, India
[1]pratik08ce@gmail.com , [2]limbasiya_kalpesh@yahoo.com

***Abstract*— The digital video application has become more and more popular in mobile terminals such as smart phones, laptop and tablet. But original video data intensity has become impractical to store and transmit. This data must be compressed to proper size at source and decompressed at destination. For this purpose, compression is useful to reduce the data size without excessively reducing quality of data. There are number of compression and decompression algorithm used for this purpose i.e. Arithmetic coding (AC), Run Length Coding (RLE), Huffman coding, Shannon-Fano coding. In this paper we try to explain use of different encoding techniques more suitable for particular data compression algorithm based on compression ratio and performance from previous work. We find that Arithmetic coding is better than all other based on compression ratio but performance of Huffman coding is better than all other encoding techniques.**

***Index Terms*—Arithmetic Encoding, Run Length Encoding, Huffman Encoding, Shannon-Fano Encoding, Video Compression.**

I. INTRODUCTION

Digital video can be found at various places such as video telephony, DVD, Internet video streaming, cellular media, digital television security/surveillance system and etc. Videos are made up from sequence of images also known as frames. Eye has a property that can sense image for 1/30th of a second [4].

Compression is an art of representing information in a compact form rather than its original form. Because of requiring large capacity of storing and transmitting of video files are not easy and it also need high bandwidth [2]. In order to overcome this problem various compression techniques have been used to minimize storage space and reducing the amount of required bandwidth to transmit video files.

The bandwidth of channel and memory of devices are limited that need to encode data contain fewer bits than original data to allow small storage and increase the speed of transmission. The process of reducing bits from original message is known as Encoding or Compression. In other words, we have some data and we decrease its size that requires few bits to store and transmit [2]. At receiver side exactly reverse of encoding is performed that is known as Decoding or Decompression.

Compression can be classified as either lossy or lossless. Lossless compression techniques reconstruct the original data from compressed file without loss of any data that is also known as reversible compression because original data are reconstructed by decompression process without any loss. Lossless techniques are used mainly in medical image and for executable files. Lossy compression techniques generate data with some discarded information so also known as irreversible. Lossy compression techniques are used in multimedia image/ video to achieve more compression [3].

This paper provides overview of different encoding techniques and highlight where they are best suited. We are going to study more popular encoding techniques like Arithmetic Coding (AC), Run Length Coding (RLC), Huffman Coding, Shannon-Fano Coding. From these techniques Arithmetic Coding and Run Length Coding provide high compression ratio but performance of Huffman Coding is best among them.

This paper organized as follows. In section 2, we take overview of video Compression and Decompression process. In section 3, we discuss in brief some compression techniques. In section 4, we introduce measuring parameters for quality matrices used in video compression. Finally, we conclude and give some future direction.

II. OVERVIEW OF VIDEO COMPRESSION

Our eye can sense image for 1/30th of a second, but if more number of frames are passed per second in front of our eye, it seen as video [4]. The video is made up from group of frames/images. Video Compression is actually image compression. For Video Compression process video is divided into frames. There are three types of frames can be generated in Video Compression technique. A frame that is spatially compressed and is independent of surrounding frames is referred as “intraframe” or I-frame. An image that is predicted from one intraframe and stores only differences is referred as “interframe” or P-frame. The image that is bidirectional predicted from two reference images is called B-frame [1]. Figure 1 shows the method of Compression and Decompression. The frames are divided into small blocks mostly of size 8x8 pixels. Discrete Cosine Transform (DCT) is performed on each block to replace each pixel value into frequency domain. This transformation is reversible. Quantization discards decimal digits and convert real coefficient to integer by rounding it. After quantization most of coefficient are zeros, they are encoded into zig-zag manner to group them together from top-left corner to bottom-right corner. Then after one of different encoding procedures is perform to compress bits [6].

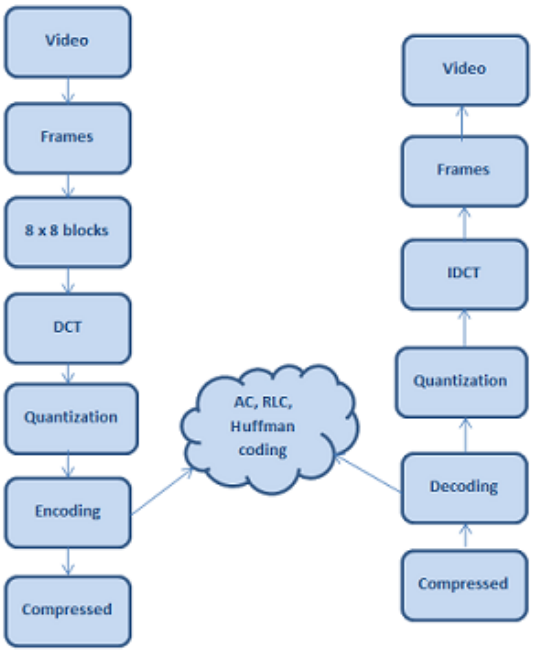

Fig. 1. Steps of Compression and Decompression

III. Technique of Compression

In digital world a lossless compression data when decompressed, it should match the original data bit by bit. In lossy compression, reconstructed data are not exactly same as original data. Lossless compression provides very high quality digital video, but requires a lot of data. Some of well-known lossless/lossy compression algorithms are [1 and 6]:

- Arithmetic Coding (AC)
- Run Length Coding (RLC)
- Shannon Fano Coding
- Huffman Coding
- Discrete Cosine Transform (DCT)
- Burrows Wheeler Transform

Using these techniques an 8-bit character or string could be represented with few bits with removing large amount of redundant data.

A. Shannon Fano Coding

Shannon Fano coding technique is one of the first algorithms which purpose is to create code word with minimum redundancy [8]. Shannon Fano coding provides a similar result compared with Huffman Coding. It will never exceed Huffman Coding [9].

Shannon Fano code does not offer the best code efficiency compare to Huffman Code. This technique is based on variable length code words. This means some particular symbol of message can be represented with shorter code compare to other code. In Shannon Fano coding, symbols are rearranged in order from most probable to least probable and then divided into two sets those total probabilities are closest to being equal.

All symbols have first digit of their codes, symbols in first set receive 0 and in second set receive 1. If two similar sets produced by partitioning of probability, only one bit is used to distinguish that is most efficient. Shannon Fano coding technique does not always produce optimal codes. For this reason Shannon Fano coding is almost never used [9].

B. Huffman Coding

Huffman Code belongs from a family of codes with a variable code word length. That means individual symbols which makes a message are represented with bit sequences that have distinct length. This characteristic of code word helps to decrease the amount of redundancy in message data. Symbols with higher probabilities are coded with shorter code words while symbols with lower probabilities are coded with larger code words [3].

This encoding technique always generates optimal code working in tree structure from leaves to root. Huffman Encoding have two types: Static Huffman Algorithm and Adaptive Huffman Algorithm. Static Huffman Algorithm first calculates frequencies and then build common tree for compression and decompression process. This tree is transferred with compressed file. The Adaptive Huffman Algorithm builds tree by calculating frequencies and also build two different tree, with flag symbol that are updated when next symbol is read [2].

The Huffman technique gives best performance but it is not used for larger files because it is very time consuming.

C. Run Length Coding

Run Length Encoding (RLE) is a very simple form of data compression encoding based on simple principle that every stream which is form of the same data values i.e. sequence of repeated data values is replaced with count number and single value [8]. RLE is usually applied to the files that contain large number of same byte pattern.

RLE is a lossless type of compression and cannot achieve great compression ratio, but a good point of that compression can be easily implemented and quickly executed [1]. This algorithm works with small redundancy. It checks for repeating symbol. The repeating string, called Run, is typically encoded with two bytes. The first byte represents number of character in run and second byte is value of character in run.

RLE is suited for compressing any type of data regardless of its information but information will affect the compression ratio. For example, "ABCBBB" is considered as a source to compress then first 3 letters are considering as non-run with length 3. The next 3 letters are considering as run with length 3 because there is a repetition of symbol B [3].

D. Arithmetic Coding

Arithmetic Coding (AC) assigns a sequence of bits to a message. Unlike Huffman Coding, Arithmetic Coding does not use discrete number of bits for each symbol [4]. The main idea behind AC is to assign each symbol an interval. Each interval is divided into subinterval, which size is equal to the current probabilities of symbols [7].

AC assigns one probability line to every symbol based on probabilities. The higher probability symbols have higher range and lower probability symbols have lower range of line [1]. After defining range and probability line, start to encode the symbols, every symbol defines where the output floating point number lands and then start to code the symbol.

Run Length coding cannot support 3GP, MP4, MKV video files that can be compressed by Arithmetic Coding. But Arithmetic Coding is not better in DAT file that can be compressed using Run Length Coding [1].

IV. Measuring Parameters

There are various criteria to measure the performance of a compression algorithm. Space efficiency is main performance measure. The time efficiency is another factor. It is difficult to measure performance of compression algorithm because it depend on redundancy of symbols [3]. The compression also depends on type of compression algorithm: lossy or lossless. If lossy compression is used to compress a file, space and time efficiency would be higher than lossless compression. Following are some measures used to evaluate performance of lossless compression.

A. Compression Ratio

Compression Ratio is the ratio between size of file after compress and size of file before compress. For any algorithm compression Ratio should be higher.

$$Compression\ Ratio = \frac{size\ after\ compression}{size\ before\ compression} \quad (1)$$

B. Compression Factor

Compression Factor is the inverse of Compression Ratio. It is the ratio between size of file before compress and size of file after compress [3].

$$Compression\ Factor = \frac{size\ before\ compression}{size\ after\ compression} \quad (2)$$

C. Compression Time

Time taken for the compression and decompression should be considered separately. In some applications the decompression time is more important, while some other applications both compression and decompression time are equally important. If time required for compression and decompression is small or acceptable than algorithm is acceptable [3]. The time for compression or decompression is mostly depended on computing device.

V. Comparing Results

According to Kodituwakku and Amaresinghe, results are shown in Table 1 [3]. By comparing Huffman Encoding and Shannon Fano Encoding in term of Compression Ratio, Compression Time and Compressed Size, we can conclude that Huffman Encoding technique is more suitable than Shannon Fano Encoding.

According to Shahbahrami, Bahrampour and Mobin Sabbaghi, results are shown in Table 2 [4]. We compare Huffman Encoding and Arithmetic Encoding for compression Ratio and Compression Time. As shows in Table 2, the compression Ratio of Arithmetic Coding for different image sizes is higher than the Huffman Coding. Arithmetic Coding needs more execution time than Huffman Coding. It needs more resources than Huffman Coding.

TABLE I. Compare Huffman and Shannon Fano encoding

Original file	**Huffman Encoding**			**Shannon Fano Encoding**		
Size	***Comp Size***	***Comp Ratio***	***Comp Time***	***Comp Size***	***Comp Ratio***	***Comp Time***
22,094	13,826	62.57	16141	14,127	63.57	14219
44,355	27,357	61.67	54719	27,585	62.19	55078

11,252	7,584	67.40	3766	7652	68.00	3766

TABLE II. COMPARE HUFFMAN AND ARITHMETIC ENCODING

Original File	**Huffman Encoding**		**Arithmetic Encoding**	
Size	***Comp Ratio***	***Comp Time***	***Comp Ratio***	***Comp Time***
2048 x 2048	6.37	32.67	12.02	63.22
1024 x 1024	5.64	8.42	7.73	20.37
512 x 512	5.27	2.13	6.55	5.67
256 x 256	4.78	0.55	5.4	5.63
128 x 128	4.38	0.14	4.65	0.43

VI. CONCLUSION

Compression techniques are improved the efficiency of compressed data. Several existing lossless compression methods are compared for their effectiveness based on implementation, compression Ratio and Compression Time. Implementation of Huffman coding is easier than all other technique. Arithmetic Coding provides much more Compression Ratio but needs large execution time. Run Length coding is faster than Arithmetic Coding but its Compression Ratio is less than Arithmetic Coding. That means in some applications where time is not important, we can use Arithmetic Coding while for some applications we have to use Huffman Coding.

ACKNOWLEDGMENT

There are lots of people who inspired and helped us making research successful. We thank to our staff member of VVP Engineering College, Rajkot, to give us valuable guidance about research and publication. We are also thankful to Sweety Maniar, Assistant Professor, VVP Engineering College, Rajkot, to encourage, advice and support us. Last, but not least our special thanks to our institute, VVP Engineering College, Rajkot, to giving this opportunity in great environment.

REFERENCES

[1] Mozammil, S.M. Zakariya and Inamullah, "Analysis of video compression algorithms on different video files," 4th international conference on computational Intelligence and Communication networks, 2012, pp. 257-262.

[2] Dalvir Kaur and Kamaljeet Kaur, "Analysis of lossless data compression techniques," Internation Journal of Computational Engineering Research. Vol. 03 April, 2013, pp.123-127.

[3] S.R. Kodituwakku and U.S. Amarasinghe, "Comparison of lossless data compression algorithms for text data," Indian Journal of Computer Science and Engineering vol. 1, pp. 416-425.

[4] A. Shahbahrami, Ramin and M. Subbaghi, "Evaluation of huffman and arithmetic algorithms for multimidea compression standards,"

[5] J.G. Apostolopoulos and S.J. Wee, "Video compression standards," wiley Encyclopedia of Electrical and Electronics Engineering, Jonh Wiley & sons. Inc. New York, 1999.

[6] Ken Cabeen and Peter Gent, "Image compression and the descrete cosine transform," Math 45, college of the Redwoods, pp.1-11, 1998.

[7] Kavitha, V. and Easwarkumar, K.S., "Enhancing privacy in arithmetic coding," ICGST-AIML journal, vol. 8, June 2008.

[8] Fundamental Compression Algorithms, Compression Team, http://oldwww.rasip.fer.hr/research/compress/algorithm/fund/.

[9] Shannon Fano coding, http://en.wikipedia.org/wiki/shannon-fano.