ISSN:0975-9646

Implementation of H.264/MPEG-4 AVC for Compound Image Compression Using Histogram based Block Classification Scheme

Dr.PS Jagadeesh kumar[1]
Professor/CSE
Don Bosco Institute of Technology,
Kumbalagodu, Bangalore - 570064

Mr.Arif Aziz[2]
ESL Lab,
Saltlake City,
Kolkata - 700064
West Bengal

Mr.Sudipta Sahana[3]
Assistant Professor/CSE
JIS College of Engineering,
Kalyani, Nadia- 741235
West Bengal

Mr.Prasanth Thapliyal[4]
M.Tech Scholar /EE
DIT University, Makkawala,
Dehradun - 248009,
Uttarakhand

Abstract-Most of the internet sites involve buffering of compound images and people expect them to be of enhanced quality and stature. In this paper, H.264/MPEG-4 AVC for compound image compression using histogram based block classification scheme is implemented to evaluate the performance of compound images to better know the importance of this scheme. In this method, conventional histogram is adopted and improves the block classification between background, text, hybrid and picture image areas. Different compression scheme is implemented for different types of classified blocks to improve the compression ratio to an optimal value. H.264 AVC using CABAC entropy coding is used to compress the hybrid blocks, on the other hand standard conventional algorithm such as run-length, wavelet coding and JPEG coding for background, text and picture blocks respectively. Experimental results show that the proposed scheme improves the compression ratio of compound image to larger extent but increases the coding complexity.

***General Terms*-Digital Image Compression, H.264 Advanced Video Coding, CABAC Entropy Coding.**

***Keywords*-Compound Image, Discrete Wavelet Transform, Run-length coding, JPEG Coding, Histogram.**

1. INTRODUCTION

With the widespread of digital devices such as digital cameras, personal computers, more and more compound images, containing text, graphics and natural images, are available in digital forms such as screen images, web pages. A compound image is an image that contains data of various types such as photographic images, text and graphics. Each of these data types has different statistical properties, and is characterized by different level of distortion that a human observer can notice. The sensitivity of human eyes for natural image and text is different. The quality requirement of compound image coding is different from general image coding because users cannot accept the quality if text is not clear enough to recognize. Most of the efforts in image compression until now have been in developing new algorithms that achieve better compression at the cost of considerable increase in complexity. The block classification algorithm and compression algorithm have low calculation complexity, which makes them very suitable for real time application. With the fast development of internet and widespread rich media applications, compound images such as web pages, slides and posters are most commonly used. For natural images, the existing image and video coding standards (e.g., JPEG2000 and H.264/AVC) have shown good coding performance. However, they are not good at compressing the compound images [1]. Thus, it is necessary to have more research work on compound image compression.

2. COMPOUND IMAGE

Electronic document images often contain mixed content types like, text, background, graphics in both gray scale and in color form because of this mixed content, and they are termed as “Compound Images”. It contains graphics, text and natural images. A compound image occurs in many important applications like document imaging and printing. Sample compound image is shown in Fig 1. Segmentation subdivides an image into its constituent regions or objects. Image segmentation is typically used to locate objects and boundaries (lines, curves, etc.) in images [1]. A general approach to compress the compound image includes the image segmentation into the regions of similar data types. Bandwidth is a very important limiting factor in application of image segmentation. Several segmentation schemes require morphological analysis of the different regions, and multiple passes over the image being segmented. However, each pass normally requires loading memory data from slow to fast memory (L1 cache, etc.), which is a slow process.

Segmentation solutions based on multiple passes are much slower (or costly) than what can be expected by, for instance, counting the number of operations. Thus, an ideal solution would use a single pass to decide on the type of image region. Such solution would be very difficult with arbitrary shapes of segmentation regions, but it is feasible if considered only a predefined shape.

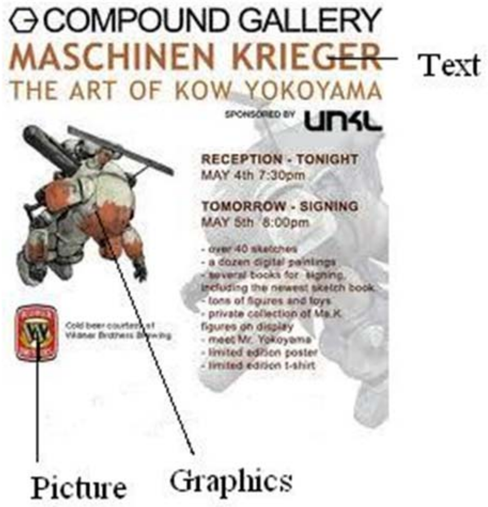

Fig 1: Sample Compound Image

3. BLOCK BASED COMPRESSION

The framework of the block-based compression scheme is shown in Fig.2. The compound image is first divided into 8x8 blocks. Then blocks are classified into four types: background, text, mixed and picture according to their different statistical characteristics. Blocks of different type will be compressed with different algorithms. The proposed scheme can effectively compress the mixed blocks, which are not well handled by some block-based algorithms. The proposed scheme achieves good coding performance on text images, picture images and compound images. It also outperforms DjVu on compound images at high bit rate. The block type map is compressed using an arithmetic coder [2].

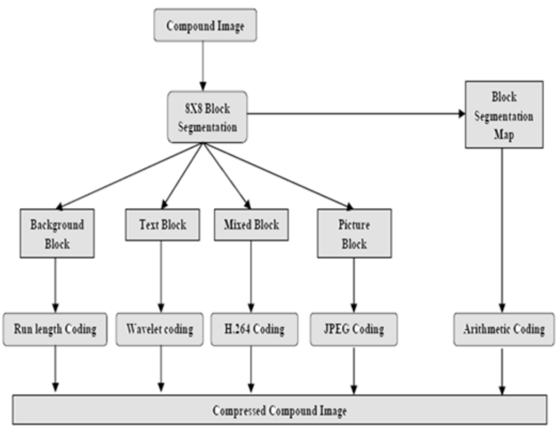

Fig 2: Block Based Compression Scheme

4. BLOCK CLASSIFICATION SCHEME

Block classification is defined as to classify the blocks into individual blocks. Classification is performed by using the histogram values. A histogram is a graphical representation, showing a visual impression of distribution of experimental data. Based upon the histogram, we have to set the threshold values. The threshold values classify the blocks as text, picture, background and mixed blocks. A fast and effective classification algorithm based on three features: histogram, gradient of the block and the anisotropic values. The entire block classification flow is shown in Fig.3. Blocks are classified into four types: background, text, mixed and picture. Blocks of different type are distinct in nature and have different statistics properties.

The background blocks contain only the low histogram pixels and show one peak at the low histogram pixels. The text blocks always shows several peaks in low histogram value (LHV). Only a few mid-histogram values (MHV) are observed in text blocks. If the block contains large numbers of high histogram and mid histogram values, it will be identified as mixed blocks [2]. The block mainly consisting of mid histogram values are declared as picture blocks. Here thresholds T1-T4 is chosen to determine the block type.

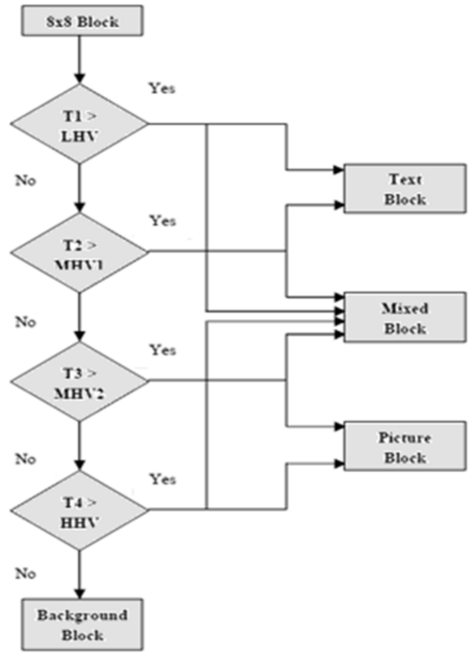

Fig 3: Block Classification Scheme

5. RUN LENGTH ENCODING

Run-length encoding (**RLE**) is a very simple form of data compression in which *runs* of data (that is, sequences in which the same data value occurs in many consecutive data elements) are stored as a single data value and count, rather than as the original run. This is most useful on data that contains many such runs. Consider, for example, simple graphic images such as icons, line drawings, and animations. It is not useful with files that don't have many runs as it could greatly increase the file size.

RLE may also be used to refer to an early graphics file format supported by CompuServe for compressing black and white images, but was widely supplanted by their later Graphics Interchange Format. RLE also refers to a little-used image format in Windows 3.x, with the extension **rle**, which is a Run Length Encoded Bitmap, used to compress the Windows 3.x startup screen. Typical applications of this encoding are when the source information comprises long substrings of the same character/binary digit. Run-length encoding performs lossless data compression and well suited to palette-based bitmapped images such as computer. It does not work well at all on continuous-tone images such as photographs; JPEG uses it quite effectively on the coefficients that remain after transforming and quantizing image blocks. Common formats for run-length encoded data include Truevision TGA, Pack Bits, PCX and ILBM. ITU also describes a standard to encode run-length-colour for fax machines, known as T.45. Run-length encoding is used in fax machines (combined with other techniques into Modified Huffman coding). It is relatively efficient because most faxed documents are generally white space, with occasional interruptions of black.

For example, consider a screen containing plain black text on a solid white background. There will be many long runs of white pixels in the blank space, and many short runs of black pixels within the text. A hypothetical scan line, with B representing a black pixel and W representing white, might read as follows:

WWWWWWWWWWWWBWWWWWWWWWWWW
WWBBBWWWWWWWWWWWWWWWWWWWWWW
WWWWWBWWWWWWWWWWWWWW

With a run-length encoding (RLE) data compression algorithm applied to the above hypothetical scan line; it can be rendered as follows:

12W1B12W3B24W1B14W

This can be interpreted as a sequence of twelve Ws, one B, twelve Ws, three Bs, etc. The run-length code represents the original 67 characters in only 18. While the actual format used for the storage of images is generally binary rather than ASCII characters like this, the principle remains the same. Even binary data files can be compressed with this method; file format specifications often dictate repeated bytes in files as padding space. However, newer compression methods such as DEFLATE often use LZ77-based algorithms, a generalization of run-length encoding that can take advantage of runs of strings of characters (such as BWWBWWBWWBWW).

Run-length encoding can be expressed in multiple ways to accommodate data properties as well as additional compression algorithms. For instance, one popular method encodes run lengths for runs of two or more characters only, using an "escape" symbol to identify runs, or using the character itself as the escape, so that any time a character appears twice it denotes a run [2]. On the previous example, this would give the following:

WW12BWW12BB3WW24BWW14

This would be interpreted as a run of twelve Ws, a B, a run of twelve Ws, a run of three Bs, etc. In data where runs are less frequent, this can significantly improve the compression rate.

One other matter is the application of additional compression algorithms. Even with the runs extracted, the frequencies of different characters may be large, allowing for further compression; however, if the run lengths are written in the file in the locations where the runs occurred, the presence of these numbers interrupts the normal flow and makes it harder to compress. To overcome this, some run-length encoders separate the data and escape symbols from the run lengths, so that the two can be handled independently. For the example data, this would result in two outputs, the string "WWBWWBBWWBWW" and thus 12, 12, 3, 24, 14.

6. WAVELET CODING

In wavelet image compression, after the quantization step entropy encoding, which is a lossless form of compression is performed on a particular image for more efficient storage. Either 8 bits or 16 bits are required to store a pixel on a digital image. Efficient entropy encoding, can use a smaller number of bits to represent a pixel in an image; this results in less memory usage to store or even transmit an image. Karhunen-Lo`eve theorem enables us to pick the best basis thus to minimize the entropy and error, to better represent an image for optimal storage or transmission. Also, Shannon-Fano entropy, Huffman coding, Kolmogorov entropy and arithmetic coding are ones that are used by engineers. Here, optimal means it uses least memory space to represent the data. i.e., instead of using 16 bits, use 11 bits. Thus, the best basis found would make it possible to represent the digital image with less storage memory. In addition, the choices made for entropy encoding varies; one might take into account the effectiveness of the coding and the degree of difficulty of implementation step into programming codes [3].

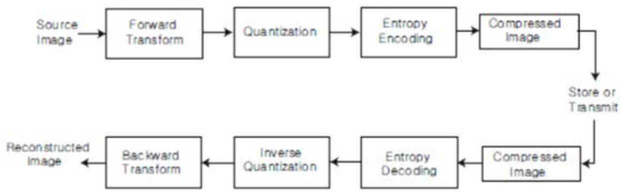

Fig 4: Wavelet Coding Scheme

7. JPEG CODING

JPEG Standard is the very well known ISO/ITU-T standard created in the late 1980s. JPEG standard is targeted for full-color still frame applications. One of the most common compression standards is the JPEG standard. Several modes are defined for JPEG including baseline, lossless, progressive and hierarchical. The most common mode uses the discrete cosine transform is the JPEG baseline coding system, also it is suitable for most compression applications. Despite being developed for low compressions JPEG it is very helpful for DCT quantization and compression. JPEG compression reduces file size with minimum image degradation by eliminating the least important information. But it is considered a lossy image compression technique because the final image and the original image are not completely the same and in lossy compression the information that may be lost and missed is affordable. JPEG compression is performed in sequential steps. JPEG compression is used in a number of image file formats. JPEG/Exif is the most common image format used by digital cameras and other photographic image capture devices; along with JPEG/JFIF, it is the most common format for storing and transmitting photographic images on the World Wide Web. These format variations are often not distinguished, and are simply called JPEG. The term "JPEG" is an acronym for the Joint Photographic Experts Group, which created the standard. The MIME media type for JPEG is *image/jpeg* (defined in RFC 1341), except in older Internet Explorer versions, which provides a MIME type of *image /pjpeg* when uploading JPEG images. JPEG/JFIF supports a maximum image size of 65535×65535 pixels, hence up to 4 gigapixels (for an aspect ratio of 1:1). JPEG uses a lossy form of compression based on the discrete cosine transform (DCT). This mathematical operation converts each frame/field of the video source from the spatial (2D) domain into the frequency domain (a.k.a. transform domain.) A perceptual model based loosely on the human psychovisual system discards high-frequency information, i.e. sharp transitions in intensity, and color hue. In the transform domain, the process of reducing information is called quantization. In simpler terms, quantization is a method for optimally reducing a large number scale (with different occurrences of each number) into a smaller one, and the transform-domain is a convenient representation of the image because the high-frequency coefficients, which contribute less to the overall picture than other coefficients, are characteristically small-values with high compressibility. The quantized coefficients are then sequenced and losslessly packed into the output bitstream. Nearly all software implementations of JPEG permit user control over the compression-ratio (as well as other optional parameters), allowing the user to trade off picture-quality for smaller file size. In embedded applications (such as miniDV, which uses a similar DCT-compression scheme), the parameters are pre-selected and fixed for the application.

The compression method is usually lossy, meaning that some original image information is lost and cannot be restored, possibly affecting image quality. There is an optional lossless mode defined in the JPEG standard. However, this mode is not widely supported in products. Widespread use of the format has stimulated the adoption of simulated high-dynamic-range imaging (HDR) modes in inexpensive cameras and smartphones, to correct the loss of shadow and highlight detail [4,8].

There is also an interlaced "Progressive JPEG" format, in which data is compressed in multiple passes of progressively higher detail [3]. This is ideal for large images that will be displayed while downloading over a slow connection, allowing a reasonable preview after receiving only a portion of the data. However, support for progressive JPEGs is not universal. When progressive JPEGs are received by programs that do not support them (such as versions of Internet Explorer before Windows7) the software displays the image only after it has been completely downloaded. There are also many medical imaging and traffic systems that create and process 12-bit JPEG images, normally grayscale images.

The 12-bit JPEG format has been part of the JPEG specification for some time, but this format is not as widely supported. JPEG compression artifacts blend well into photographs with detailed non-uniform textures, allowing higher compression ratios. Notice how a higher compression ratio first affects the high-frequency textures in the upper-left corner of the image, and how the contrasting lines become more fuzzy. The very high compression ratio severely affects the quality of the image, although the overall colors and image form are still recognizable. However, the precision of colors suffer less (for a human eye) than the precision of contours (based on luminance). This justifies the fact that images should be first transformed in a color model separating the luminance from the chromatic information, before sub sampling the chromatic planes (which may also use lower quality quantization) in order to preserve the precision of the luminance plane with more information bits.

8. H.264 ADVANCED VIDEO CODING

In 2001, with the aim of developing a more efficient compression system, the standardization bodies ISO/IEC (MPEG) and the ITU brought their efforts together in the Joint Video Team (JVT), a working group charged with developing a coding system called Advanced Video Coding (AVC). In 2003, the AVC system was integrated as part 10 in the MPEG-4 ISO/IEC 14496-10 standard and assumed the name H.264 in the ITU. In September 2004, the DVB Consortium modified ETSI standard TS 101 154 2 (Implementation guidelines for the use of Video and Audio Coding in Broadcasting Applications based on the MPEG-2 Transport Stream) to also include AVC/H.264. The goal behind the H.264 standard was to provide high quality video at considerably lower bit rates than previous standards. At the same time, the design needed to be not too complex or expensive to implement. A secondary goal was to make the standard flexible enough to be implemented across a variety of applications, networks and systems.

Video standards have hit a sweet-spot based on the compression and decompression technologies available currently [5]. Because technology is roughly doubling in performance every 18 months, every few years the industry is able to make a quantum leap in performance. H.264 vastly improves compression performance over standards such as MPEG-2 and MPEG-4 Visual. In the last few years, many promising proprietary technologies (Windows Media, Real Video, On2, Sorenson etc.) have been developed to distribute multimedia content over narrow-bandwidth transmission channels. These compression systems have been used primarily for low-bitrate video at smaller picture sizes (e.g. SQCIF, QCIF and CIF). However, their scope has more recently been expanded to provide coding efficiency across a wide range of bitrates – from a few kbit/s up to tens of Mbit/s that can cope with SDTV and HDTV.

9. CABAC ENTROPY CODING

Although Q-coder, MQ coder, and QM coder are also binary arithmetic coders with statistics adaptivity features, the *Context-based Adaptive Binary Arithmetic Coder* (CABAC) – proposed by Marpe et al. in 2001 – is adopted as the entropy coding tool for the Main profile and higher profiles of the H.264/AVC standard. Before CABAC, LUT-based VLC are generally utilized for the entropy coding stage in the hybrid block-based video coding standards including H.263, MPEG-2, MPEG-4 Part 2. The weakness of VLCs is that coding event with probability larger than 0.5 cannot be efficiently represented, and the coding procedure is not adaptive to the actual symbol statistics as the values of LUTs are fixed. The only arithmetic coder adopted in video standard is of Annex E of H.263 [ITU-T Rec. H.263], in which coding efficiency of entropy coding is not significantly improved, because it directly uses SEs of VLC for arithmetic coding without redefinition. Before CABAC proposals by H.264/AVC, similar arithmetic-coder-based implementations were first investigated and applied to the non-block-based video coding, such as DWT. CABAC is the first successful arithmetic coding scheme deployed in the video coding standard, with significant compression improvement compared to the previous entropy coding tools. CABAC encoding process consists of three elementary *steps*: binarization, context modeling, and binary arithmetic coding (BAC). The input SEs are binarized into bin strings, in which regular bins and bypass bins are encoded separately by the encoding engines of BAC. For regular bin coding, the context model (probability model) of the bin is prepared by the step of context modeling. Details of the three steps are discussed in the following sections [5, 6].

9.1 Binarization

Binarization maps the non-binary valued SE into bin string, which is a sequence of binary decision (bin). Three types of bins are generated in the binarization step: *regular bin*, *bypass bin*, and *terminate bin* for the bins with unequal (variable), equal, or dominant probabilities of value 1 and 0, respectively. Advantages of binarization include:

(a) The probability of non-binary SE can be represented by the probabilities of individual coding bins, while compression efficiency is not influenced;

(b) Low-complexity BAC can be utilized;

(c) Context modeling at sub-symbol (sub-SE) level provides more accurate probability estimation than context modeling at the symbol level, and the alphabet of the encoder is reduced. Five binarization schemes are used in CABAC: Unary (U), Truncated Unary (TU), kth order Exp-Golomb (EGk), concatenation of the first and third scheme (UEGk), and fixed length binarization (FL). Kth ordered Exp-Golomb binarization (EGk) – a derivative of Golomb coding is proved to be an optimal prefix-free coding for geometrically distributed sources. EGk codeword consists of prefix and suffix bin strings, with total length of $2l+k+1$ bits. EGk prefix is a Unary codeword, with l bits of 1 and one terminating bit 0. The length l of string of bit 1 is represented as:

$$l = \left\lceil \log_2\left(\frac{x}{2^k}+1\right)\right\rceil$$

The length of suffix binary string is equal to $l + k$, and the value of the suffix string is:

$$EGk_{suffix} = x + 2^k - 2^{k+l}$$

UEGk is combined binarization scheme of TU and EGk. It is utilized for binarization of SEs of absolute value of residual coefficient level and MVD. TU generates prefix of the bin string, and EGk is adopted to generate the suffix with k set to 0 and 3 for coefficient level and MVD, respectively. TU is simple and it permits fast adaptation of probability of coding symbol. However, it is only beneficial for small SE values. For large SE values, suffix bin string generated by EGk provides a good fit to the probability distribution, and bypass bin coding is also utilized to reduce computation complexity.

9.2 Context modeling

The context modeling implements two sub-functions: context model selection and context model access. The statistics of the coded SEs are utilized to update the probability models (context model) of regular bins. For regular bin coding, one context model is chosen, and fetched from a pre-defined set of context models to provide the probability of regular bin to be MPS or LPS, and the context model is updated after bin coding based on bin value. The context index (*CtxIdx*) is calculated to select context model, which is the sum of context offset (*CtxOffset*) and context index increment (*CtxIdxInc*). *CtxOffset* locates the context model set of processed SE, while *CtxIdxInc* selects one context model from the set based on the values of coded bins or coded SEs of neighboring coded blocks. The idea of multiplication-free arithmetic coding of H.264/AVC is based on the assumption that estimated probability of each context model can be represented by a sufficiently limited set of representative values. In CABAC, the number of the representative values is set to 64 to enable accurate estimation, which is larger than the 30 of Q-coder. Each context model contains a 1-bit tag of MPS value, and a 6-bit probability state index (*pStateIdx*) that addresses one of 64 representative probability values of LPS from $p0$ to $p63$ in the range of [0.01875, 0.5]. The ratio of two neighboring probability values is a constant value α, which is approximately equal to 0.949.

$$p_\sigma = \alpha \cdot p_{\sigma-1}$$
$$for: \sigma = 1,\ldots,63, \quad \alpha = \left(\frac{0.01875}{0.5}\right)^{1/63}, and \quad p_0 = 0.5$$

By mapping the update probability value of LPS of (2.4) to the closest value in the aforementioned set of representative values, multiplication of probability estimation of CABAC is replaced by simple table lookup for the *pStateIdx* of next probability state according to *pStateIdx* of the current bin, and based on whether it is MPS or LPS. This probability value estimation of context state is actually the function state transition of FSM with 64 predefined states. This type of probability FSM is first utilized in Q-coder, and adopted in QM coder and MQ coder. Compared to Q-coder, QM coder, and MQ coder, the representative LPS probability values need not to be stored in CABAC. Instead, the approximation of the products of coding interval Range and the LPS probability of (1.37) are stored. In order to be more adaptive to the coding context, the values of MPS and LPS can be exchanged when the probabilities of MPS and LPS are equal and the coding bin is LPS.

For particular regular bins of CABAC, multiple context models are allocated for single bin to more precisely represent probabilities of bin in different coding contexts. Four types of context model selection techniques are supported in CABAC, based on (a) neighboring coded SE values of the current SE, (b) values of prior coded bins of SE bin string, (c) position of the to-be-encoded residual coefficient in the scanning path of residual block coefficients, and (d) level values of encoded coefficients of residual block. If the coding bin is MPS, the probability of LPS decreases by simply multiplying the ratio α, while for the LPS bin, the update probability of MPS is calculated first, and then the probability of LPS are obtained.

$$p_{new} = \begin{cases} \max(\alpha \cdot p_{old}, p_{62}), & if \quad bin = MPS \\ 1 - \alpha \cdot (1 - p_{old}), & if \quad bin = LPS \end{cases}$$

9.3 Binary Arithmetic Coding (BAC)

BAC performs arithmetic coding of each bin based on bin value, type, and the corresponding context model of the bin. BAC is a recursive procedure of coding interval subdivision and selection. Coding interval subdivision mechanism of CABAC is different from that of QM and MQ coders. In QM and MQ coders, calculation of RangeLPS is simplified by using the approximated value 1 for Range, and multiplication is eliminated. In comparison, Range is also utilized for RangeLPS calculation of CABAC. The reference interval subdivision and selection of regular bin, in which the 2 higher-order bits of Range (bit 7 and bit 6) and the index of probability state (*pStateIdx*) of LPS are used to look up for the precalculated product of Range and *p*LPS in a 2-D LUT. Although the product of LUT is of limited precision, the precision of RangeLPS calculation, interval subdivision is improved, and computational complexity is reduced in CABAC, compared to those of QM and MQ coders. Because Range and Low of coding interval are represented by finite number of bits (9 bits for Range and 10 bits for Low), it is necessary to renormalize (scale up) the interval to prevent precision degradation, and the upper bits of Low are output as coded bits during renormalization.

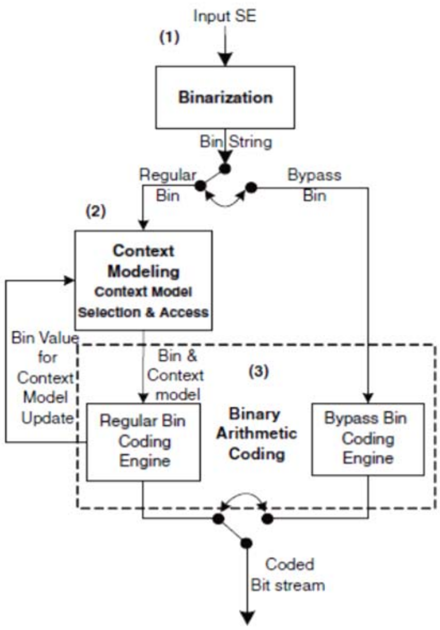

Fig 5: CABAC Entropy Coding

Coding interval renormalization and bit output of CABAC is based on Witten's algorithm. The coding interval of (Low, Low + Range) is renormalized when Range is smaller than the threshold value 256 (0x100), which is 1/4 of the maximum range of coding interval. Renormalization of Range and Low is an iterative procedure, and the maximum number of iterations is 6, as the smallest possible value of Range is 6. For the processing of carry propagation and output of coding bits, the coded bits of CABAC are not output until it is confirmed that further carry propagation will not influence bit values. Only when interval length (Range) is smaller than the threshold 0x100, one bit can be output if the interval is located within the top half [0x200, 0x400) or bottom half [0, 0x200) of the maximum coding range, or an outstanding (OS) bit is accumulated when the interval is within [0x100, 0x300). When a bit of value *X* is output in BAC, the accumulated OS bits are output with the value 1–*X*. Compared to the bit stuffing or byte stuffing schemes of Q-coder, QM coder, and MQ coder, carry propagation is completely solved during renormalization of BAC, and no additional processing of bit stream is needed at CABAC decoder. Moreover, as no bits or bytes are stuffed in the bit stream, the compression efficiency of CABAC is further improved. However, the renormalization illustrated in Fig. 5 is a highly sequential operation, and as the number of iterations is variable depending on the selected subinterval Range, it is challenging for SW or HW acceleration of renormalization and bit output of BAC. In some situations, long delay may be experienced, when a large number of OS bits are accumulated.

10. IMPLEMENTATION

In this compression scheme, the compound image is given as input and then converted to grayscale. The grayscale array image is further classified into background, text, hybrid and natural image using histogram based block classification method. The classified blocks are further compressed using different compressed algorithm. H.264 AVC using CABAC entropy coding is used to compress the hybrid blocks, on the other hand standard conventional algorithm such as run-length, wavelet coding and JPEG coding for background, text and picture blocks respectively.

10.1 Getting the input compound image and converting it to grayscale

The compound image is given as input. The input compound image is converted to grayscale. Calculates the size of the input compound image and returns the number of rows and columns of the input compound image pixel array.

10.2 Using Discrete Wavelet Transform to get the transformed values

Daubechies 9/7 which is used here is usually used for lossless compression. In general Daubechies wavelet has extremal phase and highest number of vanishing moments for defined support width. The wavelet is also easy to put into practice with minimum-phase filters. This wavelet is often also called the CDF 9/7 wavelet (where 9 and 7 denote the number of filter taps). There are several ways wavelet transforms can decompose a signal into various sub bands. These include uniform decomposition, octave-band decomposition, and adaptive/wavelet packet decomposition. Out of these, octave-band decomposition is the most widely used. The decomposition of the signal into different frequency bands is simply obtained by successive high pass and low pass filtering of the time domain signal. This filter pair is called the analysis filter pair. First, the low pass filter is applied for each row of data, thereby getting the low frequency components of the row. But since the low pass filter is a half band filter, the output data contains frequencies only in the first half of the original frequency range. They are down-sampled by two, so that the output data contains only half the original number of samples. Now, the high pass filter is applied for the same row of data, and similarly the high pass components are separated.

10.3 Encoding and Decoding Phase

The sample values are encoded and then decoded using Context adaptive base arithmetic coding (CABAC). When entropy coding mode is set to 1, an arithmetic coding system is used to encode and decode H.264 syntax elements. The arithmetic coding scheme selected for H.264, Context-based Adaptive Binary Arithmetic Coding or CABAC, achieves good compression performance through (a) Selecting probability models for each syntax element according to the element's context (b) Adapting probability estimates based on local statistics and (c) using arithmetic coding.

Coding a data symbol involves the following stages:

10.3.1. Binarization: CABAC uses Binary Arithmetic Coding which means that only binary decisions (1 or 0) are encoded. A non-binary-valued symbol (e.g. a transform coefficient or motion vector) is "binarized" or converted into a binary code prior to arithmetic coding. This process is similar to the process of converting a data symbol into a variable length code but the binary code is further encoded (by the arithmetic coder) prior to transmission. Stages 2, 3 and 4 are repeated for each bit (or "bin") of the binarized symbol.

10.3.2. Context model selection: A "context model" is a probability model for one or more bins of the binarized symbol. This model may be chosen from a selection of available models depending on the statistics of recently-coded data symbols. The context model stores the probability of each bin being "1" or "0".

10.3.3. Arithmetic encoding: An arithmetic coder encodes each bin according to the selected probability model. Note that there are just two sub-ranges for each bin (corresponding to "0" and "1").

10.3.4. Probability update: The selected context model is updated based on the actual coded value (e.g. if the bin value was "1", the frequency count of "1"s is increased).

10.3.5. Calculating compression ratios and evaluating performance:

The compressed image pixel array is obtained from Inverse DWT. Now from this signal to noise ratio is calculated after calculating MSE. In order to evaluate the performance of video compression coding, it is necessary to define a measure to compare the original video and the video after compressed. Most video compression systems are designed to minimize the *mean square error (MSE)* between two video sequences Ψ1 and Ψ2, which is defined as

11. Comparisons Of CABAC With Other Entropy Coders

The coding efficiency of CABAC is higher than those of the Q-coder, QM coder, and MQ coder, because of (a) the more precise multiplication of RangeLPS, (b) larger number of probability states for each probability model, and more precise probability estimation of coding bins; and (c) more context models (probability models) deployed for various coding contexts of different types of SEs. Because of the high computational complexity of CABAC, another entropy coding tool CAVLC is deployed in the Baseline profile and extended profile of H.264/AVC targeting low bit-rate real-time video coding. It offers compression-complexity trade-off with lower complexity, and lower coding efficiency, compared to CABAC. It is employed to encode the quantized transform coefficients of 4×4 residual blocks, while zero-order Exp-Golomb codes (EG0) are used for all other types of non-residual SEs. Adaptivity is introduced to CAVLC by allowing switching among multiple VLC tables based on the already processed SEs, and the coding efficiency of CAVLC is better than those of the previous VLC coders which used single VLC table. Instead of coding data pair of run-level as a single SE, run and level of the residual block are encoded separately in CAVLC, so that the inter-symbol redundancy can be more efficiently exploited. However, compression efficiency of CABAC is significantly higher, with typically bit rate reduction of 9–14% in the video quality range of 30–38 dB, compared to CAVLC and EG0. This is because (a) in CABAC, encoding symbols can be more precisely represented in non-integer number of bits, especially for the symbol with probability higher than 0.5, and (b) CABAC encoder is more adaptive to the non-stationary symbol statistics with efficient context modelling (probability estimation) for the coding bins of all types of SEs. Since the adoption of CABAC entropy coding in H.264/AVC, CABAC is also applied in many applications of image and video processing including motion mode and residual data of 3D dynamic mesh, prediction residual in lossless 4D medical image compression, SEs of 8×8 transform coefficients of AVS coding standard, motion vector coding of scalable video coder, parameters of depth and correction vectors in multi-view video coding. CABAC is also utilized to encode affine motion vector, and MVD of 3-D DWT-based subband video encoder [6, 7].

12. Experimental Result

The famous toy store compound image as shown in Fig.6 is taken as the input image to our proposed H.264/MPEG-4 AVC using histogram based block classification scheme. The segmented image is shown in the Fig.7. The block classification for text, hybrid, background and picture blocks are shown in the Fig. 8, 9, 10, 11 respectively. The proposed system has been simulated in MATLAB-SIMULINK environment. H.264 AVC using CABAC entropy coding is used to compress the hybrid blocks, on the other hand standard conventional algorithm such as run-length, wavelet coding and JPEG coding for background, text and picture blocks respectively. The proposed scheme is tested for different type of input compound images as shown in Fig. 13, 15, 17, 19 and their corresponding compressed images are shown in Fig. 14, 16, 18, 20.

Fig 6: Original input compound image

Fig 7: Converted grayscale image

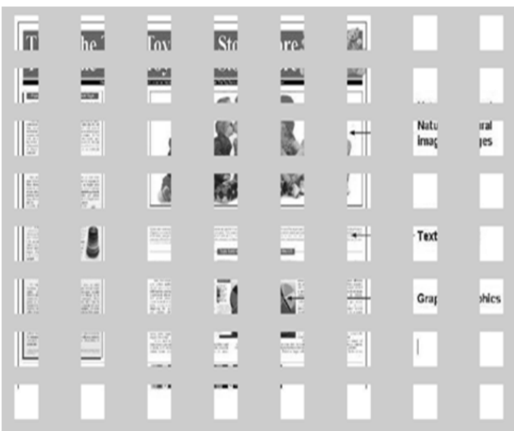

Fig 8: Segmented image

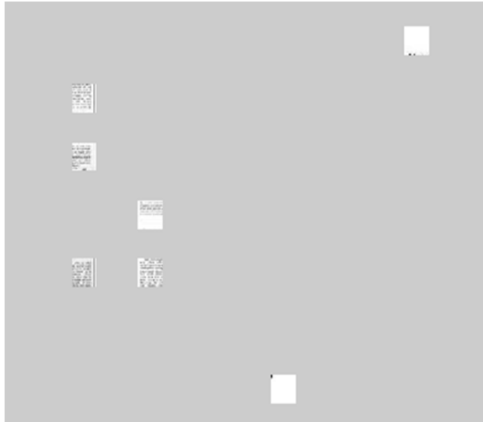

Fig 9: Text classified blocks

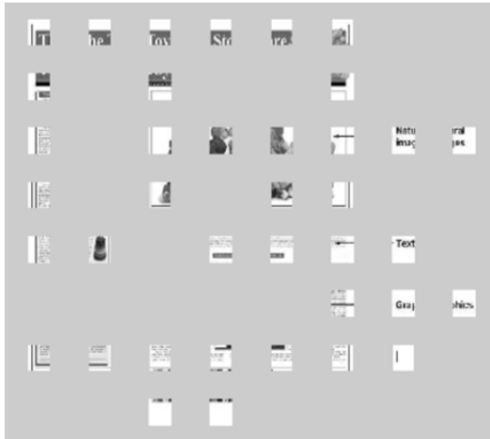

Fig 10: Hybrid classified blocks

Fig 11: Background classified blocks

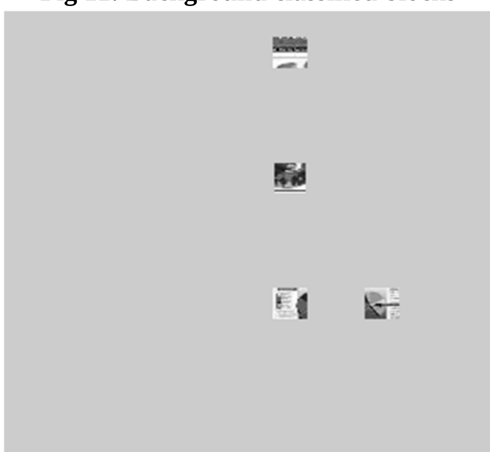

Fig 12: Picture classified blocks

Fig 13: Input compound image 1

Fig 14: Compressed compound image 1

Fig 15: Input compound image 2

Fig 16: Compressed compound image 2

Fig 17: Input compound image 3

Fig 18: Compressed compound image 3

Fig 19: Input compound image 4

Fig 20: Compressed compound image 4

CONCLUSION AND FUTURE ENHANCEMENT

The block classification scheme based on histogram method is very simple and effective, reducing the computational complexity. The histogram based block classification scheme was 92% efficient for compound images [1]. Considering the fact that sensitivity for human eyes can mismatch 3% of block classification, though the proposed scheme failed to make the same consistency for other type of images as such can be negotiated. The classified text blocks, background blocks, picture blocks and hybrid blocks are compressed using wavelet coding, run-length encoding, JPEG coding and H.264 /MPEG-4 AVC respectively. The PSNR and Compression ratio were found to be comparatively better but the proposed scheme is very heavy and complicated because of various algorithms associated with it. In future, a single compression scheme might reduce the coding complexity to greater extent by implementing methods such as genetic algorithm, neural network, sub-band coding.

ACKNOWLEDGEMENT

We sincerely thank all of our friends, family and colleagues, who helped us in difficult moments of making this project. Our special thanks to the management of DBIT Bengaluru, JISCE Kalyani, DIT University Dehradun and SSNCE Chennai. Also our sincere regards to ESL Technology, Kolkata for their extended support throughout.

REFERENCES

[1] PS Jagadeesh Kumar, Dr.Gagan Singh, "Performance evaluation of H.264 AVC using CABAC entropy coding for image compression" ICACCS, Jan'2013, Atlantis press, Dubai, pp: 320-326.

[2] S Kumar, "Histogram based block classification scheme of compound images: A Hybrid Extension" IJARCET, Vol.3, Issue 9, September'2014, pp: 3166-3120.

[3] S Kumar, "Wavelet sub-band block coding based lossless high speed compression of compound image", IJARCST, Vol.2, Issue 3, July-Sep'2014, pp: 263-268.

[4] Dr.PSJ Kumar, Mr.Indranil Sarkar "Speckle reduction of ultrasound image using wavelet transforms", IJCEA, Vol.8, Issue 3, Part 1, December'2014, pp: 122-129.

[5] Cuiling lan, Guangming Shi and Feng wu, "Compress Compound Images in H.264/MPEG-4 AVC by exploiting Spatial Correlation" IEEE Transactions on Image Processing, vol.19 no.4, pp. 946-957, April 2010.

[6] Wenpeng ding, Yan lu and Feng wu, "Enable Efficient Compound Image Compression in H.264/AVC Intra Coding" IEEE Transactions on Image Processing, Vol.10 no.3, pp. 337-340, Sep.2009.

[7] B.-f Wu, C.-C Chiu and Y. –L Chen "Algorithms for compressing compound document images with large text/background overlap", IEEE Proc.Vis. Image signal Process, Vol. 151 No. 6, pp.453-459 December 2008. Brown, L. D., Hua, H., and Gao, C. 2003. A widget framework for augmented interaction in SCAPE.

[8] D. Mukherjee, N. Memon, and A. Said, "JPEG-matched MRC compression of compound documents," Proc. IEEE Int. Conf. Image Processing, vol. 3, pp. 434–437, Oct. 2001.

AUTHOR

Dr.P.S.Jagadesh Kumar, Professor in the Department of Computer Science and Engineering, Don Bosco Institute of Technology, Bengaluru has 16 years of teaching experience, including 6 year of research experience in the field of image compression. He received his B.E degree from University of Madras in Electrical and Electronics Engineering discipline in the year 1999. He obtained his M.E degree in 2004 with specialization in Computer Science and Engineering from Annamalai University, Chidambaram and his Ph.D. from Anna University, Chennai.