International Journal of Emerging Technology in Computer Science & Electronics (IJETCSE)
ISSN: 0976-1353 Volume 24 Issue 4 – MARCH 2017.

# Compressing Video using Asymmetric Algorithm and implementing Blind Video Watermarking Techniques

R.Sriramkumar[#1] and S.Majinababy[*2]

[#] *M.Tech, Department of CSE, Assistant Professor, Kings college of engineering, Thanjavur. India*
[*] *M.Tech, Department of CSE,SASTRA University, Thanjavur, India*

***Abstract*— Compression is to reduce the file size of image , audio and video files. All the web images you get on the site are compressed, typically in the JPEG or GIF formats, most modems use compression, HDTV will be compressed using MPEG-2, and several file systems automatically compress files when stored, and the rest of us do it by hand. The neat thing about compression, as with the other topics we will cover in this course, is that the algorithms used in the real world make heavy use of a wide set of algorithmic tools, including sorting, hash tables, tries, and FFTsthe capacity and speed of storage devices have been tremendously improving. Today, new kinds of cheaper and more efficient memory devices are constantly emerging . In my paper the Discrete Cosine Transform (DCT) based blind video watermarking algorithm is proposed, which is perceptually invisible and robust against rotation and collusion attacks and pixels will not be broken so that after compression of video we can get the quality as orginal. To make the scheme resistant against rotation, watermark is embedded within the square blocks, placed on the middle position of every luminance channel. Then Zernike moments of those square blocks are calculated.**

***Index Terms*— DCT, DAS, LZW, WMT, Zernike**

## I. Introduction

Video compression is the process of encoding a video file in such a way that it consumes less space than the original file and is easier to transmit over the network/Internet. It is a type of compression technique that reduces the size of video file formats by eliminating redundant and non-functional data from the original video file. Video compression techniques were started in 1984 when the images and audio files where gone through wide range of focusing level by the people Lossy audio compression is used in a wide range of applications. In addition to the direct applications (MP3 players or computers), digitally compressed audio streams are used in most video DVDs, digital television, streaming media on the internet, satellite and cable radio, and increasingly in terrestrial radio broadcasts. storage media like diskettes, hard disks, CDs, USB Flash Disks, and tapes. Most data that we store in our computer devices are digital each unit of information packed in binary form. This binary nature of the source is how much information it really contains, and which better way to represent that information in a smaller number of binary digits, or bits. The compressed file is ultimately a concatenation of thousands or even millions of bit strings. Clearly, the cost of sending data over communications networks is minimized if the files are highly compressed. Digital watermarking is the method of embedding data into digital multimedia content. This is used to verify the credibility of the content or to recognize the identity of the digital content's owner. Using Watermarking techniques The rotation invariance property of the Complex Zernike moments [2] is exploited to predict the rotation angle of the video at the time of extraction of watermark bits. To make the scheme robust against collusion, design of the scheme is done in such a way that the embedding blocks will vary for the successive frames of the video.

Visible Digital Watermarking: Visible data is embedded as the watermark. This can be a logo or a text that denotes a digital medium's owner. Invisible Digital Watermarking: The data embedded is invisible or, in case of audio content, inaudible. Robust watermarks involve blending signal amplitude with large bandwidth sizes and a short message length. Frequency domain capabilities and mixed-domain techniques, when added to signals, are believed to provide the right amount of robustness in order to guard against watermark attacks.
The publisher Playboy has used an invisible form of digital watermarking to detect where its copyrighted material has been illegally posted on other websites.

A Pseudo Random Number (PRN) generator and a permutation vector are used to achieve the goal. The experimental results show that the scheme is robust against conventional video attacks, rotation attack and collusion attacks. A PRNG suitable for cryptographic applications is called a cryptographically secure PRNG (CSPRNG). A requirement for a CSPRNG is that an adversary not knowing the seed has only negligible advantage in distinguishing the generator's output sequence from a random sequence. In other words, while a PRNG is only required to pass certain statistical tests, a CSPRNG must pass all statistical tests that are restricted to polynomial time in the size of the seed.

Though a proof of this property is beyond the current state of the art of computational complexity theory, strong evidence may be provided by reducing the CSPRNG to a problem that is assumed to be hard, such as integer factorization. In general, years of review may be required before an algorithm can be certified as a CSPRNG.

## II. MACHINE LEARNING

There is a close connection between machine learning and compression: a system that predicts the posterior probabilities of a sequence given its entire history can be used for optimal data compression (by using arithmetic coding on the output distribution) while an optimal compressor can be used for prediction (by finding the symbol that compresses best, given the previous history). This equivalence has been used as a justification for using data compression as a benchmark for "general intelligence data compression can be viewed as a special case of data differencing: Data differencing consists of producing a difference given a source and a target, with patching producing a target given a source and a difference, while data compression consists of producing a compressed file given a target, and decompression consists of producing a target given only a compressed file. Thus, one can consider data compression as data differencing with empty source data, the compressed file corresponding to a "difference from nothing." This is the same as considering absolute entropy (corresponding to data compression) as a special case of relative entropy (corresponding to data differencing) with no initial data.

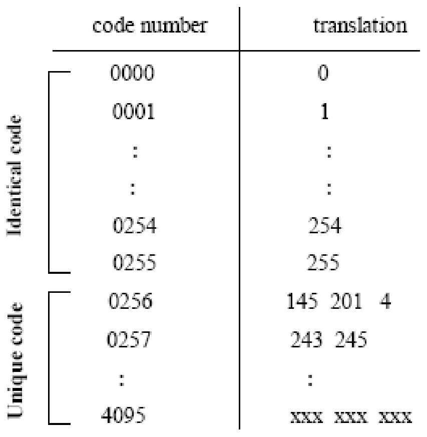
## III. COMPRESSING DATA

Compression is useful because it reduces resources required to store and transmit data. [6]Computational resources are consumed in the compression process and, usually, in the reversal of the process (decompression). Data compression is subject to a space–time complexity trade-off. For instance, a compression scheme for video may require expensive hardware for the video to be decompressed fast enough to be viewed as it is being decompressed, and the option to decompress the video in full before watching it may be inconvenient or require additional storage. The design of data compression schemes involves trade-offs among various factors, including the degree of compression, the amount of distortion introduced (when using lossy data compression), and the computational resources required to compress and decompress the data

[1]Multimedia compression is employing tools and techniques in order to reduce the file size of various media formats. With the development of World Wide Web the importance of compress algorithm was highlighted because it performs faster in networks due to its highly reduced file size.

A compressed file appears to increase the speed of data transfer over an uncompressed file. Data compression involves a certain amount of raw data being processed and stored in a much more compact form. The processing is done by computer applications that use specific compression algorithms. There are many of these, each with its own strengths and weaknesses and particular areas where they are used. Since raw digital video produces enormous file sizes, the video file must be compressed so that it can be stored and transferred. File sizes need to be reduced so they can fit on a CD-ROM (700 MB) or DVD (4 GB) or transferred over the web at 1-2 Mbpm. File compression is achieved through a codec, a compression/decompression algorithm that reformats data so it takes up less space. Video compression is lossy compression that uses the MPEG format as the standard. A variety of different codec's are available which produce widely varying results, both in terms of compression ratios and in terms of image quality

LZW Compression is used in this paper and it is used to encode the 12 bits data



| | code number | translation |
|---|---|---|
| Identical code | 0000 | 0 |
| | 0001 | 1 |
| | : | : |
| | : | : |
| | 0254 | 254 |
| | 0255 | 255 |
| Unique code | 0256 | 145 201 4 |
| | 0257 | 243 245 |
| | : | : |
| | 4095 | xxx xxx xxx |

Fig 1: Code table compression

### A. *LZW COMPRESSION ALGORITHM*

```
1 Initialize table with single character strings
2 String = first input character
3 WHILE not end of input stream
4 Char = next input character
5 IF String + Char is in the string table
6 String = String + Char
7 ELSE
8 output the code for String
9 add String + Char to the string table
10 String = Char
11 END WHILE
12 output code for String
```

## IV. HUFFMAN'S ALGORITHM

Huffman"s original algorithm and the one-pass Algorithm FGK. First let us define the notation we use

n = alphabet size;

aj = jth letter in the alphabet;

t = number of letters in the message processed so far;

at = ai1,, ai2,, . . . , ait, the first t letters of the message;

k = number of distinct letters processed so far; wj = number of occurrences of aj processed so, far;

lj = distance from the root of the Huffman tree to aj"s leaf.

The constraints are 1 ≤ j, k ≤ n, and 0 ≤ wj ≤ t. In many applications, the final value oft is much greater than n. For example, a book written in English on a conventional typewriter might correspond to t ≈ 106 and n = 87. The ASCII alphabet size is n = 128. Huffman‟s two-pass algorithm[5] operates by first computing the letter frequencies wj in the entire message. A leaf node is created for each letter aj that occurs in the message; the weight of aj‟s leaf is its frequency wj. The meat of the algorithm is the following procedure for processing the leaves and constructing a binary tree of minimum weighted external path length *Σj wj lj* Store the k leaves in a list L; While L contains at least two nodes do begin Remove from L two nodes x and y of smallest weight; Create a new node p, and make p the parent of x and y; p‟s weight := x‟s weight + y‟s weight; Insert p into L end;

However, its only drawback is a heavy computational load. The node remaining in L at the end of the algorithm is the root of the desired binary tree. We call a tree that can be constructed in this way a "Huffman tree." It is easy to show by contradiction that its weighted external path length is minimum among all possible binary trees for the given leaves. In each iteration of the **while** loop, there may be a choice of which two nodes of minimum weight to remove from L. Different choices may produce structurally different Huffman trees, but all possible Huffman trees will have the same weighted external path length. In the second pass of Huffman‟s algorithm, the message is encoded using the Huffman tree constructed in pass 1. The first thing the sender transmits to the receiver is the shape of the Huffman tree and the correspondence between the leaves and the letters of the alphabet. This is followed by the encodings of the individual letters in the message. Each occurrence of *aj* is encoded by the sequence of 0‟s and 1‟s that specifies the path from the root of the tree to *aj*‟s leaf, using the convention that "0" means "to the left" and "1" means "to the right." To retrieve the original message, the receiver first reconstructs the Huffman tree on the basis of the shape and leaf information. Then the receiver navigates through the tree by starting at the root and following the path specified by the 0 and 1 bits until a leaf is reached. The letter corresponding to that leaf is output, and the navigation begins again at the root. The two main disadvantages of Huffman‟s algorithm are its two-pass nature and the overhead required to transmit the shape of the tree. In section we explore alternative one-pass methods, in which letters are encoded "on the fly." We do not use a static code based on a single binary tree, since we are not allowed an initial pass to determine the letter frequencies necessary for computing an optimal tree. Instead the coding is based on a dynamically varying Huffman tree. That is, the tree used to process the (t + 1)st letter is a Huffman tree with respect to $\mu t$ . The sender encodes the (t + 1)st letter ait, in the message by the sequence of 0‟s and 1‟s that specifies the path from the root to ait„s leaf. The receiver then recovers the original letter by the corresponding traversal of its copy of the tree. Both sender and receiver then modify their copies of the tree before the next letter is processed so that it becomes a Huffman tree for μt+ 1.

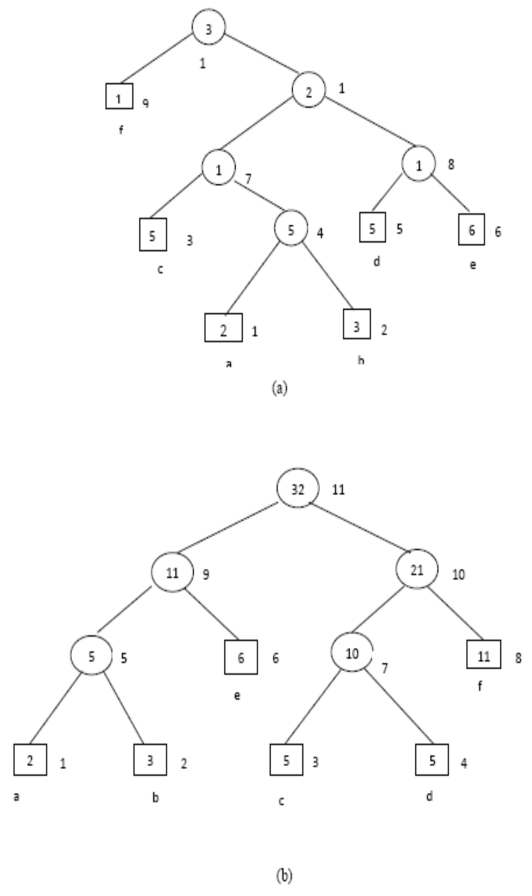


Figure 1.2 (a) The current status of the dynamic Huffman tree, which is a Huffman tree for $\mu t$ , the first *t* letters in the message. The encoding for "b" is "1011", given by the path from the root to the leaf for "b". (b) The tree resulting from the interchange process. It is a Huffman tree for $\mu t$ and has the property that the weights of the traversed nodes can be incremented by I without violating the sibling property.

## A. *COMPRESSION PROCESSOR*

It consists of CAMs, an 5- byte shift register, a shift and update control, and a codeword output circuit. The word widths of CAMs increase gradually from 2 bytes up to 5 bytes with 5 different address spaces: 256, 64, 32, 8 and 8 words. The input string is shifted into the 5-byte shift register. The shift operation can be implemented by barrel shifter for achieving a faster speed. Thus there are 5 bytes can be searched from all CAMs simultaneously. In general, it is possible that there are several dictionaries in the dictionary set matched with the incoming string at the same time with different string lengths. The matched address within a dictionary along with the dictionary number of the dictionary that has largest number of bytes matched is outputted as the output codeword, which is detected and combined by the priority encoder. The maximum length string matched along with the next character is then written into the next entry pointed by the update pointer (UP) of the next dictionary (CAM) enabled by the shift and dictionary update control circuit. Each dictionary has its own UP that always points to the word to be inserted next. Each update pointer counts from

0 up to its maximum value and then back to 0. Hence, the FIFO update policy is realized. The update operation is inhibited if the next dictionary number is greater than or equal to the maximum dictionary number

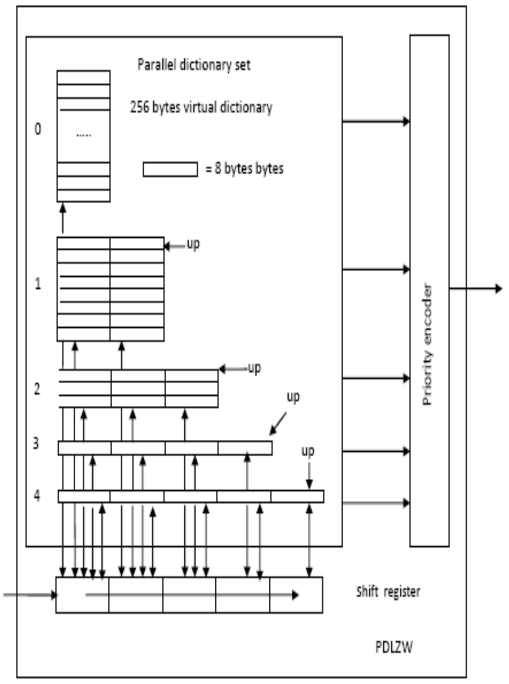


Fig 1.2 Architecture of Compression technique LZW

## V. ASYMMETRIC ALGORITHM

The block-matching algorithm (BMA) is the most popular and widely used algorithm for motion estimation due to its simplicity and reasonable performance. In BMA, an image frame is divided into non-overlapping rectangular blocks with equal or variable block sizes. The pixels in each block are assumed to have the same motion. The motion vector (MV) of a block is estimated by searching for its best match within a search window in the previous frame. The distortion between the current block and each searching block is employed as a matching criterion. The resulting MV is used to generate a motion compensated prediction block. The motion-compensated prediction difference blocks (called residue blocks) and the MVs are encoded and then sent to the decoder. Among the various BMAs, the full search (FS) is global optimal and most straightforward algorithm because it searches the entire search window for the best matching block

### *A. DIRECTIONAL ASYMMETRIC SEARCH*

The center-biased characteristic has been reported and widely used in many studies. It means that most MVs are very close to zero motion. Thus, the best search strategy is to search from the center of the search window and its nearest neighbors. On the other hand, although the hypothesis of monotonic error surface is not always true, it holds true primarily in the nearby region around the minimum error point (global or local). This implies that the minimum error point can be found along the direction of the block error from the highest to the lowest point. As long as the error direction is known, only the points along the search path need to be checked.

## VI. 6 ERROR DIRECTION DETERMINATION

Most fast BMAs find the MBD in each step using symmetric patterns. The location of the MBD in the current step is the center of the next step until the MBD occurs at the center. Under this condition, the information regarding block distortions has not been effectively used. Thus far, for most fast BMAs, it is used only for finding the MBD. Actually, it reveals not only the MBD but also the error direction. In our study, the error direction (or search direction) in each step is defined as the direction from the location of the maximum block distortion toward the minimum block distortion. Herein S denotes a search pattern, BD(k) denotes the block distortion of the search point k, and k S ∈ is a point in S.

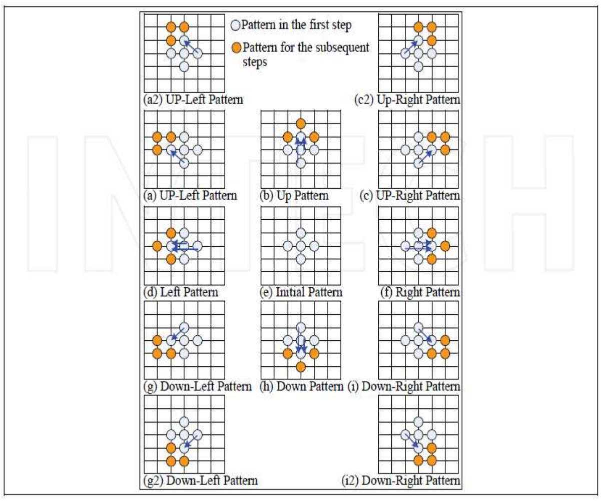


Fig 2: Search pattern of DAS

## VII. WATER MARKING TECHNIQUES

firstly, the binary watermark image *W* (with values -1 or 1) is transformed into a noise-like two dimensional binary sequence using the chaotic mixing method. In the chaotic mixing method, a mapping matrix *AN (k)* , with *LN ->LN* , is generated, where *LN* is a two-dimensional indexes set, and it is applied iteratively to the watermark pattern *W*, as shown by

$$W^d = W^{(i)} = A_N^i(k)W^{(0)}, \quad i = 1, 2, \ldots, P-1$$

$$A_N(k) = L_N \rightarrow L_N, \begin{pmatrix} x_{n+1} \\ y_{n+1} \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ k & k+1 \end{pmatrix} \begin{pmatrix} x_n \\ y_n \end{pmatrix} \pmod N$$

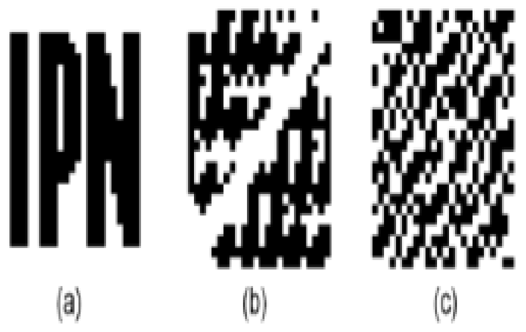
where, $W^{(0)}$is the original watermark pattern, $W^{(i)}$ is a noise-like watermark pattern after the *i-th* application of $A_N$ *(k)*, *(xn , yn)* € *LN* , *k* € *[*1, *N*] < *Z* , P-*1* is the total number of possible watermarks that can be generated using and *N* is the size of

Two keys are required to reconstruct the watermark pattern: the first key is *k* and the iteration number *i* is the second one. Fig shows the watermark patterns generated by with different iteration *i* .

The whole host video sequence is used for watermark embedding. Firstly the [3] first level decomposition of each frame of the luminance channel *Fr,(r=1,2,..,R)* is computed using a two dimensional Daubechies wavelet function, where R is the total number of frames. Then the noise-like watermark generated by is embedded into the information subband,N*LL1*..Here the wavelet coefficients of frame *Fr* are denoted by

*Xr( r=* 1,2,…, *R).* Next this noise-like watermark is embedded into the magnitudes of the wavelet coefficients *Xr* , which are segmented into non overlapping blocks of size 3x3. Then the mean of each block is computed and denoted by*M* . Subsequently, a bit of watermark is embedded by changing the value of center coefficient *Vc* of each block to get a modified value *Vc* . Based on two thresholds ( 1 *Th* , 2 *Th* ) and an intensity factor α , the computation of *Vc* is carried out

The network transportation path can be viewed as a noisy channel. The reference digital watermark is modulated with multimedia data (carrier) and transmitted onto the noisy channel. The watermark undergoes the same changes suffered by the multimedia data, so that the watermark degradation can be used to estimate the overall alterations of the multimedia data caused by noise or by attacks. At the receiver side the embedded digital watermark is extracted and compared to the original reference watermark in order to measure the integrity and authenticity of the received multimedia data. It is obvious that the proposed framework integrate the timing mechanism to the multimedia data naturally.



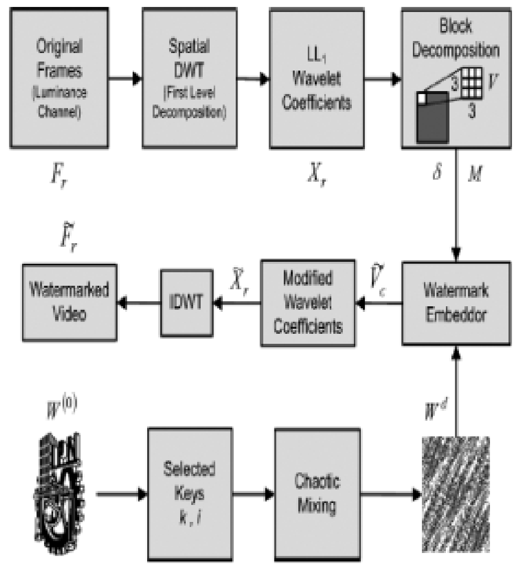


Figure 3:Flowchart of the video watermark embedded scheme

## A. IMAGE AUTHENTICATION PROCEDURE

In the image authentication procedure shown in Fig. 4,given corrupted images by transmission and their associated digital signatures, the proposed scheme authenticates both the integrity and the source of the received image by applying the following process on the image in the following order: (1) perform content-adaptive error concealment, if some blocks are damaged; (2) extract the SDS of the received image using the same method used in image signing; (3) decrypt the signature by using the sender's public key; (4) perform a content authenticity verification procedure using both the decrypted signature and the extracted one to calculate the degree of authenticity (DY) and un-authenticity (DN); (5) deem the image authenticated if DY > DN; otherwise (6) the attacked areas are detected using an attack detector.

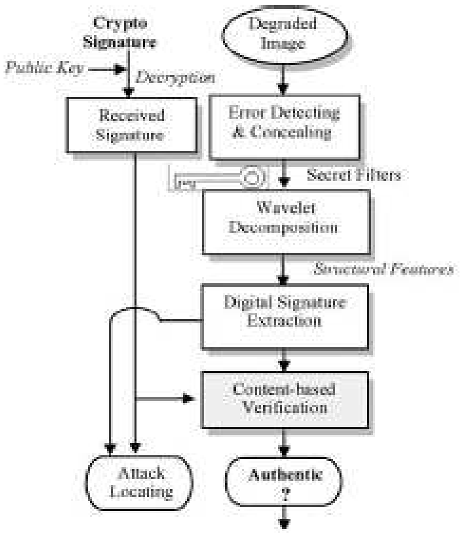


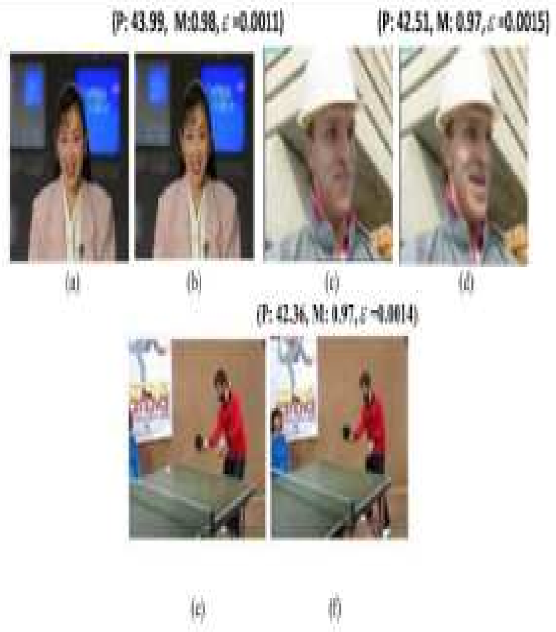
Figure 4: Image authentication procedures

## VIII. RESULT

In the proposed scheme, the watermark information is embedded in every frame of the video to make the algorithm robust against frame dropping and frame insertion.

YCbCr (luminance, chrominance-red, chrominance-blue) color model is chosen instead of RGB (red, green, blue) color model and the luminance parts of the video are considered as the target embedding area. This is due to the fact that the RGB color space representation has the most correlated components, while the YCbCr color components are the least correlated components . The correlated RGB components are not suitable to embed the watermark. In RGB color space the perceived color quality of a video frame is dependent on all components. Thus, embedding watermark bits into one component independently of the other RGB components is not the best choice. On the other hand the YCbCr permits to extract uncorrelated components and it favors the separation of the achromatic part from the chromatic parts of the color image. To achieve high robustness and large embedding capacity, the proposed scheme uses the least correlated YCbCr components of the color image. The color image is represented by Y, Cb and Cr components. The present scheme uses luminance component for data embedding to make the scheme robust against compression. This is due to the fact that during compression no down sampling is done in the luminance component That means the loss in luminance component during compression is less than the chrominance component. So the watermark bits can be detected effectively if the bits are embedded in the luminance component.

- A binary logo of size (M x N) is chosen as the watermark logo. Hence, every bit of the binary logo is either ‘‘0” or ‘‘1”. To make a non-zero sequence, all the ‘‘0”s are converted to ‘‘1”s.
- To make the scheme robust against rotation attack, luminance block’s center is chosen as the center. A square area of size (P x P) is chosen as the embedding area where the unit circle has to be placed at the time of computation of Complex Zernike moments. The size of (P x P) is multiple of (8 x 8). This is due to the fact that the present scheme uses a block of size (8 x 8) to embed a watermark bit.
- To make the scheme robust against collusion attacks, a Pseudo Random Number (PRN) is used to select the embedding blocks (8 x 8). It also ensures that the embedding blocks will vary for the successive frames. To generate the seed value of the PRN generator a permutation vector is used and is kept as a secret key (K). Here, permutation vector is a user defined secret key (K) that is used as an input (seed value) to the PRN generator. At the time of extraction


(P: 43.99, M:0.98, ε =0.0011)
(P: 42.51, M: 0.97, ε =0.0015)
(a)
(b)
(c)
(d)
(P: 42.36, M: 0.97, ε =0.0014)
(e)
(f)

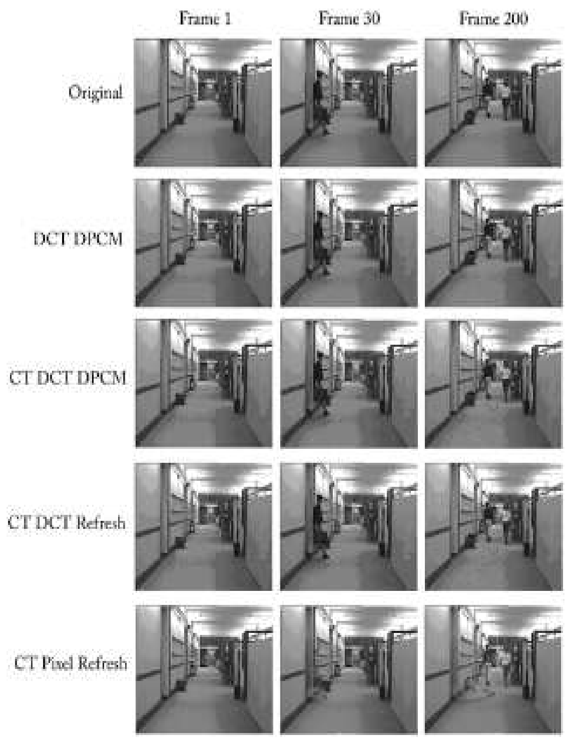


Figure 4.1 Compression of video

## IX. SIMULATION RESULTS

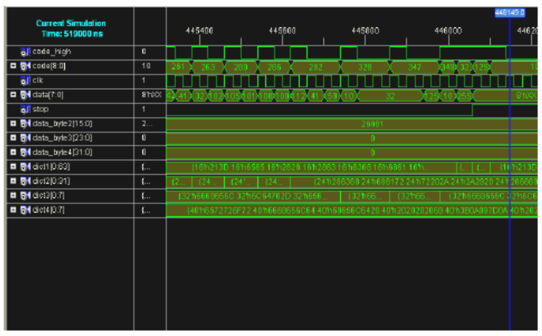


Figure 5 LZW Output

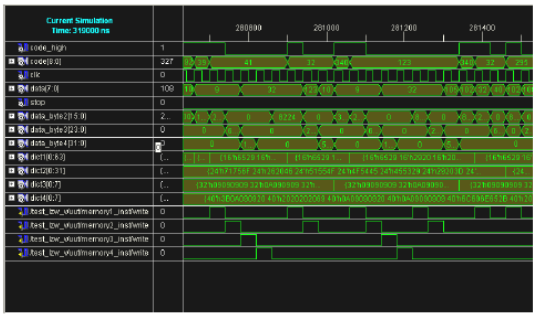

Current Simulation Time: 310000 ns
code_high
code[8:0]
clk
data[7:0]
stop
data_byte2[15:0]
data_byte3[23:0]
data_byte4[31:0]
dict1[0:63]
dict2[0:31]
dict3[0:7]
dict4[0:7]
test_lzw_vlsi/memory1_inst/write
test_lzw_vlsi/memory2_inst/write
test_lzw_vlsi/memory3_inst/write
test_lzw_vlsi/memory4_inst/write

Figure 5.1 Write operation

The proposed two-stage compression/decompression processor given in Fig 5.3 has been synthesized and simulated using Verilog HDL. The resulting chip has a die area of 4.3× 4.3mm and a core area of 3.3 ×3.3 mm . The simulated power dissipation is between 632 and 700 mW at the operating frequency of 100 MHz. The compression rate is between 16.7 and 125 MB/s; the decompression rate is between 25 and 83 MB/s. Since we use D-type flip-flops associated with needed gates as the basic memory cells of CAMs (the dictionary set in the PDLZW processor) and of ordered list (in the AHDB processor), these two parts occupy most of the chip area. The remainder only consumes about 20% of the chip area. To reduce the chip area and increase performance, the full-custom approach can be used. A flip-flop may take between 10 to 20 times the area of a six-transistor static RAM cell , a basic CAM cell may take up to 1.5 times the area (nine transistors) of a static RAM cell. Thus, the area of the chip will be reduced dramatically when full-custom technology is used. However, our HDL-based approach can be easily adapted to any technology, such as FPGA, CPLD, or cell library

The resulting architecture shows that it is not only to reduce the hardware cost significantly but also easy to be realized in VLSI technology since the entire architecture is around the parallel dictionary set and an ordered list such that the control logic is essentially trivial. In addition, in the case of executable files, the performance of the proposed architecture is competitive with that of the LZW algorithm (compress). The data rate for the compression processor is at least 1 and up to 5 B per memory cycle. [7]The memory cycle is mainly determined by the cycle time of CAMs but it is quite small since the maximum capacity of CAMs is only 64 × 2 B for the PDLZW processor and 414 B for the AHDB processor.

## X. CONCLUSION

The experimental results show that the scheme provides robustness against rotation of video in any angle, collusion attack of Type-1 and Type-2 and conventional video attacks, including a Rayleigh fading wireless channel. Future work can be concentrated on further performance improvement of the proposed scheme, as well as the development of hardware implementation of the proposed scheme through field programmable gate array (FPGA). H.264 presents a huge step forward in video compression technology. It offers techniques that enable better compression efficiencies due to more accurate prediction capabilities, as well as improved resilience to errors. It provides new possibilities for creating better video encoders that enable higher quality video streams, higher frame rates and higher resolutions at maintained bit rates (compared with previous standards), or, conversely, the same quality video at lower bit rates.

## XI. ACKNOWLEDGMENT

Author thanks for the guidance of Mr.J.Jegan, Ap, KCE. for giving me a helpful guidance to geta idea for the concept of watermarking scheme. Under the reference of digital watermarking using multi resolution wavelet decomposition by KUNDER D, HATZINAKOS D

## REFERENCES

[1] T. Akiyama et al., MPEG-2 video codec using image compression DSP, IEEE Transactions on Consumer Electronics 40 (3) (1994) 466472. 1074I. Ahmad et al. / Parallel Computing 28 (2002) 10391078

[2] S.M. Akramullah, I. Ahmad, M.L. Liou, Optimization of H.263 video encoding using a single processor computer: Performance trade-o Systems for Video Technology 11 (8) (2001) 901915

[3] Khayam, S.A., 2003. The discrete cosine transform (DCT) – theory and application: information theory and coding, <http://www.lokminglui.com/DCT_TR802.pdf>.

[4] Ahmad, I.; Zheng, W.; Luo, J. & Liou, M. (2006). A fast adaptive motion estimation algorithm, IEEE Trans. Circuits Syst. Video Technol., Vol. 16, No. 3, pp. 420-438, ISSN: 1051-8215.

[5] Chao, C., Tie-gang, G., Li-zong, L., 2008. A compressed video watermarking scheme with temporal synchronization. In: Proc. Of IEEE Congress on Image and Signal Processing, Sanya, China, 605–612.

[6] Amlan karmakar., Amit phadikar., Baisakhi sur phadikar., 2016. A blind video watermarking scheme resistant to rotation and collusion attacks.king saud university,india,199-210.

[7] KUNDER D., HATZINAKOS D.: ‘Digital watermarking using multi resolution wavelet decomposition’. Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing, Seattle, Washington, 2008.