HOKKAIDO UNIVERSITY

Study on Data Hiding Techniques for Videos enabling Data Size Reduction

A DISSERTATION PRESENTED
BY
XUEFEI LI
TO
GRADUATE SCHOOL OF INFORMATION SCIENCE AND TECHNOLOGY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF INFORMATION SCIENCE AND TECHNOLOGY
IN THE FIELD OF
COMPUTER SCIENCE AND INFORMATION TECHNOLOGY

HOKKAIDO UNIVERSITY
SAPPORO, HOKKAIDO
SEPTEMBER 2024

Contents

Listing of figures

List of Tables

1

Introduction

The convergence of media formats, encompassing video, audio, text, and interactivity, has transformed multimedia into an inseparable element of our daily lives. Fueled by rapid technological advancements, multimedia techniques have undergone significant development in recent years. Movies and dramas, prime examples of multimedia content, have captivated audiences for decades, typically combining video, audio, and

sometimes subtitles.

Traditionally, physical media like Capacitance Electronic Discs (CEDs), Video High Density (VHD) tapes, Video Compact Discs (VCDs), Digital Versatile Discs (DVDs), Blu-ray Discs (BDs), and Ultra HD Blu-ray Discs (UHD BDs) served as portable storage solutions for multimedia content. DVDs and BDs currently reign supreme in the physical storage realm, with UHD BDs poised to take center stage in the coming years due to their support for 4K resolution videos.

On the other hand, online Video on Demand (VOD) services have emerged, housing vast repositories of multimedia data. These services, along with streaming services, deliver multimedia content directly to users in a compressed format, balancing quality with bandwidth limitations.

1.1 Data Hiding for Compressed Videos

Data hiding for videos, a burgeoning field of technology, has recently captured significant attention [1, 2]. Researchers have actively explored video and image data hiding techniques, with some foundational methods established even before the recent surge in interest [3–6]. These techniques can be broadly categorized based on their objectives, including scrambling and watermarking for copyright protection, steganography for covert communication, and multimedia enrichment to enhance content value.

In the digital age, information security and content manipulation are constant concerns. Data hiding techniques offer a unique solution, allowing us to embed additional information within existing data objects like images, audio, or video. These techniques can serve various purposes, from copyright protection to covert communication. Here, I

delve into four prominent data hiding techniques: scrambling, watermarking, steganography, and enrichment.

1.1.1 Scrambling: Shuffling the Pieces

Imagine rearranging the pieces of a puzzle, making it difficult to understand the original image without the key. Scrambling, in the context of data hiding, operates on a similar principle. It involves modifying the original data in a reversible way, making it unintelligible without the proper decryption key. This technique is often used for real-time applications where preserving data integrity is crucial.

There are two main types of scrambling techniques:

- Spatial Scrambling: This method involves rearranging the spatial location of data elements within the host data. For example, pixels in an image might be shuffled according to a specific algorithm.
- Frequency Scrambling: Here, the data is transformed into the frequency domain (e.g., using techniques like Fourier Transform) and the frequency components are rearranged. This method is commonly used for audio and video data.

However, scrambling has limitations:

- Security Concerns: Scrambling itself doesn't provide strong security against unauthorized access. Without robust key management, the decryption key could be intercepted, revealing the original data.
- Loss of Information: Depending on the scrambling technique, some information might be lost during the process, potentially impacting the quality of the host data.

Several scrambling techniques have been proposed in recent years [7–9].

1.1.2 Watermarking: A Digital Copyright Stamp

Similar to a tiny, invisible copyright symbol embedded within a document, watermarking involves imperceptibly embedding a signal, typically containing copyright information or ownership details, into a host digital media file. This embedded data, called the watermark, is robust enough to survive common manipulations like compression, cropping, or format conversion.

There are two main types of watermarks:

- Visible Watermarking: This method intentionally leaves a visible mark on the host data, deterring unauthorized use. Logos, copyright symbols, or even invisible patterns visible under specific lighting conditions are examples.
- Invisible Watermarking: This technique aims to embed the watermark imperceptibly, making it difficult to detect without specialized tools.

Watermarking serves several purposes:

- Copyright Protection: By embedding ownership information, watermarks help identify and track copyrighted material.
- Authentication: Watermarks can verify the authenticity of digital media, ensuring it hasn't been tampered with.
- Authentication: Content Control: Watermarks can be used to restrict access or control the distribution of digital content.

However, there are limitations to consider. Watermarks can be fragile, potentially destroyed by strong compression or malicious attacks. Additionally, extracting the watermark often requires specific tools and knowledge. A number of watermarking approaches have been developed lately [10–12].

1.1.3 Steganography: The Art of Hidden Communication

Unlike watermarking, steganography prioritizes secrecy. The goal is to hide a message within a host data object in a way that's undetectable to the naked eye or standard analysis tools. Imagine hiding a secret message within the static of a noisy radio signal – that's the fundamental principle of steganography.

Steganography techniques exploit the inherent redundancy present in digital media. For example, slight variations in pixel brightness in an image or inaudible changes in audio volume can be used to encode a hidden message without affecting the overall quality of the host data.

There are several applications of steganography:

- Covert Communication: This technique allows for secure communication by embedding messages within seemingly innocuous data like images or audio files.
- Information Security: Steganography can be used to hide sensitive information within digital media for secure storage or transmission.
- Forensic Applications: Law enforcement agencies sometimes utilize steganography to embed tracking information within digital evidence.

However, steganography can be ethically questionable when used for malicious purposes. Additionally, detecting steganographic messages can be challenging, requiring specialized software and expertise. Some prominent steganography techniques have been proposed in the recent past [13–15].

1.1.4 Enrichment: Adding Value Through Data Hiding

While watermarking and steganography focus on embedding additional information, enrichment takes a different approach. This technique aims to enhance the value of the host data itself by incorporating additional information or functionalities.

For example, an image might be enriched by embedding captions or annotations directly into the data file. An audio file could be enriched with additional sound effects or synchronized lyrics. This embedded data can be accessed and utilized by compatible devices or software, enriching the user experience.

Some applications of data enrichment include:

- Multimedia Description: Enrichment can be used to embed captions, descriptions, or keywords within media files, improving accessibility for visually impaired or non-native speakers.
- Interactive Content: Embedded data can enable interactive experiences within digital media. Imagine clicking on specific areas within an image to access additional information.
- Error Correction and Recovery: Redundant data embedded in the host file can be used for error correction or recovery in case of data corruption.

While enrichment offers valuable functionalities, it's important to maintain a balance between the amount of additional data embedded and the file size. Excessive data can lead to larger file sizes, impacting storage and transmission efficiency. The field has seen a surge of interest in enrichment techniques, with some methods emerging in recent years [16, 17].

1.1.5 Targeting compressed videos

Scrambling, watermarking, steganography, and enrichment represent diverse data hiding techniques, each serving distinct purposes. Scrambling offers real-time protection but requires key management. Watermarking safeguards ownership and authenticity. Steganography facilitates covert communication but raises ethical concerns. Enrichment enhances the user experience through embedded information. As technology evolves, these techniques will continue to play a significant role in information security, copyright protection, and user experience in our increasingly digital world.

However, a critical challenge arises when dealing with compressed video data, commonly used in both physical discs and online services. Embedding additional data directly into raw, uncompressed video formats (e.g., YUV sequences) carries a high risk of data loss during the compression process, particularly after quantization.

To circumvent this issue, embedding data within compressed video streams presents a more viable solution. Common video compression formats employed in DVDs and BDs include MPEG-2 and H.264/AVC, while UHD BDs typically utilize H.265/MPEG-H Part 2 (HEVC). These formats are also prevalent in online VOD and streaming services. Within these formats, video data undergoes block-based orthogonal transforma-

tions, such as Discrete Cosine Transform (DCT) and Discrete Sine Transform (DST), to generate frequency domain coefficients. Subsequently, these coefficients are quantized for compression purposes. Notably, all processes following quantization are entirely lossless.

Therefore, quantized Transform Coefficients (qTCs) represent ideal targets for lossless data hiding within compressed videos. Several data hiding techniques leveraging qTCs have been proposed [18–24].

1.2 Data size limitation

In today's world, video has become the dominant form of communication and entertainment. From social media interactions to blockbuster movies, captivating visuals have revolutionized the way I consume information and tell stories. However, behind the scenes, a silent battle rages on – the battle for data capacity.

Data capacity refers to the amount of information a storage system can hold. In the realm of video, it dictates the quality, resolution, and overall length of a video file. Limited data capacity significantly impacts the final product. Lower resolutions, with blocky pixels and blurry details, become the norm. Additionally, video lengths become restricted, forcing creators to compromise on narrative or artistic vision.

The ever-increasing demand for high-quality video experiences further exacerbates the data capacity challenge. Viewers crave sharper visuals, immersive experiences like virtual reality, and seamless streaming without buffering interruptions. These advancements all require a significant leap in data capacity.

Fortunately, technological advancements are constantly pushing the boundaries of

data capacity. Higher storage densities in hard drives, faster data transfer speeds, and efficient compression techniques all contribute to a better video experience. Cloud storage solutions have also emerged, offering seemingly limitless data capacities, allowing for the storage and distribution of massive video files.

However, the race between data capacity and video quality is a continuous one. As creators push the boundaries of visual storytelling with higher resolutions, more complex effects, and 360-degree experiences, the demand for even greater data capacity persists.

The importance of data capacity extends beyond the technical aspects. It impacts accessibility and affordability. With limited data plans, viewers in developing regions or those with restricted internet access may be excluded from the richness of high-quality video experiences. Ensuring affordable and accessible high-capacity data plans is crucial for bridging the digital divide.

In conclusion, data capacity plays an invisible yet critical role in the world of video. It dictates the quality, length, and accessibility of the visual stories I share. As technology continues to evolve, the quest for ever-increasing data capacity remains paramount, ensuring that video continues to be a powerful tool for communication, entertainment, and artistic expression.

As discussed above, all existing multimedia storage solutions have inherent capacity limitations, fundamentally. Similarly, online VOD and streaming services prioritize maintaining an optimal compression rate. Nonetheless, conventional data hiding techniques for compressed videos often result in an overall increase in file size. This poses a significant challenge, potentially exceeding storage limitations and compromis-

ing video streaming quality. Consequently, a portion of either the video data or the additional embedded data may be irrevocably lost.

Another key aspect to consider is the reversibility of the cover data (i.e., the original video), which has a great impact on data size change. Data hiding techniques can be classified as either reversible or irreversible. Reversible techniques, while allowing for the complete restoration of the original video data, often require significant redundancy, leading to a substantial increase in file size. This characteristic proves detrimental for video discs with strict storage limitations [25–28]. Conversely, irreversible techniques operate without this high redundancy overhead, minimizing file size increases [29–32].

To preserve the integrity of both the original video content and the embedded data while maintaining high-quality video streaming, the ideal data hiding technique would demonstrably not increase the overall file size. Fortunately, a few such techniques have been proposed in the past [30–34]. Notably, the Least Significant Bit (LSB) replacement technique and its variants offer a solution for uncompressed video data, enabling data size reduction [30–33]. Another technique involves losslessly embedding specific qTC blocks within others for compressed images, but it is limited to manipulating the cover image itself and cannot embed additional external data (essentially functioning as a data-hiding-based compression technique).

1.3 Structure of this thesis

Here, I propose a novel partial scrambling technique embedding audio into video and a novel data hiding technique for compressed videos enabling data size reduction.

Coding process of compressed videos will be explained in Chapter 2, basis and

some relative data hiding techniques will be explained in Chapter 3, the proposed partial scrambling technique embedding audio into video will be explained in Chapter 4 [35, 36], with the experimental results and discussion shown in Chapter 5, the proposed data hiding technique for compressed videos enabling data size reduction will be explained in Chapter 6 [37], with the experimental results and discussion shown in Chapter 7. The last chapter, Chapter 8, is the conclusion of the thesis.

2

Video data

2.1 Overview

Processing of the techniques in my study, such as AC component sign inversion processing in video [38], is performed when uncompressed video is encoded to a format using DCT such as MPEG-2 and MPEG-4. Therefore, it will be explained along with the encoding process to MPEG-4, as an example.

2.2 Coding process of MPEG-4 video

MPEG-4 uses the $\mathrm{YC_bC_r}$ color system, just like JPEG compression. Y is the luminance component, and $\mathrm{C_b}$ and $\mathrm{C_r}$ are the chrominance components. For uncompressed video that uses the RGB color system, a color system conversion process must first be performed. To convert the RGB color system to the YCbCr color system, the following equation (2.1) is used.

$$\begin{bmatrix} \mathrm{Y} \\ \mathrm{C_b} \\ \mathrm{C_r} \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.16874 & 0.33126 & 0.500 \\ 0.500 & -0.41869 & -0.08131 \end{bmatrix} \begin{bmatrix} \mathrm{R} \\ \mathrm{G} \\ \mathrm{B} \end{bmatrix}. \tag{2.1}$$

Next, all frames are divided into 16×16 MacroBlocks (MBs). MBs are further divided into four 8×8 blocks. At this time, chroma subsampling is performed for the chroma components, so that there are two $\mathrm{C_b}$ and $\mathrm{C_r}$ blocks in one MB. After subsampling, all MBs are classified into intra-MBs and inter-MBs, and motion prediction is performed for inter-MBs. The classification and motion prediction methods for MBs are complex and do not affect the AC component sigh inversion process, so they will be omitted here.

Then, DCT is performed on all MBs in 8×8 block units. DCT generates one DC (Direct Current) component and 63 AC (Alternating Current) components for one block. The transformation equation for DCT is shown below.

Table 2.1 DC coefficient nonlinear quantization step for MPEG-4

q	DC scale for luminance	DC scale for chrominance
1 to 4	8	8
5 to 8	$2 \times q$	$(q + 13)/2$
9 to 24	$q + 8$	$(q + 13)/2$
25 to 31	$2 \times q - 16$	$q - 6$

$$F(u,v) = \frac{1}{4} k_u k_v \sum_{x=0}^{7} \sum_{y=0}^{7} f(x,y) \cos \frac{(2x+1)u\pi}{16} \cos \frac{(2y+1)v\pi}{16}, \tag{2.2}$$

where,

$$k_u, k_v = \begin{cases} 1/\sqrt{2} & (u, v = 0) \\ 1 & (otherwise). \end{cases}$$

DCT is followed by quantization. Quantization is a process that increases the compression rate by dividing the coefficients obtained by DCT and expressing them by coefficients with smaller values. When performing quantization, the compression rate can be adjusted by setting a parameter called q (qscale). In MPEG-4, unlike conventional MPEG and MPEG-2, the DC coefficient of the intra block is quantized by dividing it by the DC scale obtained by calculating the DC scale with a nonlinear quantization step determined by q, when performing DC coefficient quantization. The DC coefficient nonlinear quantization step is shown in Table 2.1.

In addition to the DC coefficients of intra blocks, there are also AC coefficients of intra blocks and DC and AC coefficients of inter blocks. For these coefficients, there are two quantization methods. The first quantization method is the same as that of con-

ventional MPEG and MPEG-2: the coefficients obtained by DCT are multiplied by 16 and then divided by the quantization coefficient values in the quantization table, and then divided by the value of q to obtain smaller coefficients. Figure 2.1 shows an example of a quantization table for the Y component in MPEG-4. The second quantization method is defined by the following equation, where C^* is the quantized DCT coefficient (qDCTC), C is the quantized DCT coefficient before quantization, and $sign(x)$ is the sign of x. The results of the quantization process are qDCTCs.

$$C^* = \begin{cases} sign(c) \times |c|/2q & (intra) \\ sign(c) \times (|c| - q/2)/2q & (inter)\,. \end{cases} \tag{2.3}$$

Once qDCTC is obtained, a proposed process is performed on qDCTC. For example, if an AC component sign inversion process is performed, the positions to conduct inversion in one block can be determined in advance using a pseudo-random number. For the AC coefficients A at the predetermined sign inversion positions, sign inversion processing is performed using the following equation, and a new AC coefficient A^* is obtained. AC coefficients other than the sign inversion positions are not processed and their values are left unchanged. An example of AC coefficient sign inversion processing is shown in Fig. 2.2.

$$A^* = -A\,. \tag{2.4}$$

When generating pseudo-random numbers, PRNGs (pseudorandom number generators) such as the Mersenne twister (commonly known as MT) [39] can be used. The generation method involves determining a seed in advance and generating pseudo-random integers within a specified range as needed. The use of pseudo-random numbers improves the resistance to decryption of the proposed processing. In addition, for the reverse processing, the used seed must be saved in a key file.

After the proposed processing, VLC (Variable Length Coding) is performed. In MPEG-4, for example, VLC is performed separately for the DC coefficients of intra blocks and other coefficients. There is one DC coefficient for each intra block in the DC coefficients of intra blocks. The DC coefficient of an intra block is calculated as the difference between the DC coefficient of the reference block and the current block. The absolute value of the difference is first used to classify the difference into a category. Then, a code with a different code length is assigned according to the category, and padding bits are added for coding. Table 2.2 shows the code table for Y component DC coefficients.

For the remaining coefficients, the scanning order is first determined by the method of AC coefficient prediction. There are three types of scanning order, which are called zigzag scan order, vertical priority scan order, and horizontal priority scan order. As an example, the zigzag scan order is shown in Fig. 2.3. Then, the 63 AC coefficients are arranged block by block according to the scanning order and encoded using 3D variable length coding.

When generating 3D variable-length codes, the number of consecutive zeros preceding a non-zero coefficient in the scan order is called zerorun, the value of the non-zero

8	17	18	19	21	23	25	27
17	18	19	21	23	25	27	28
20	21	22	23	24	26	28	30
21	22	23	24	26	28	30	32
22	23	24	26	28	30	32	35
23	24	26	28	30	32	35	38
25	26	28	30	32	35	38	41
27	28	30	32	35	38	41	45

Fig. 2.1 An example of a quantization table in MPEG-4 for Y component DC coefficients

Table 2.2 Code table for Y component DC coefficients

difference	code	padding bit
0	00	none
-1, 1	010	0, 1
-3, -2, 2, 3	011	00, 01, 10, 11
-7...-4, 4...7	100	000...011, 100...111
-15...-8, 8...15	101	0000...0111, 1000...1111
-31...-16, 16...31	110	00000...01111, 10000...11111
-63...-32, 32...63	1110	000000...011111, 100000...111111
-127...-64, 64...127	11110	0000000...0111111, 1000000...1111111
-255...-128, 128...255	111110	00000000...01111111, 10000000...11111111
-511...-256, 256...511	1111110	000000000...011111111, 100000000...111111111
-1023...-512, 512...1023	11111110	0000000000...0111111111, 1000000000...1111111111
-2047...-1024, 1024...2047	111111110	00000000000...01111111111, 10000000000...11111111111

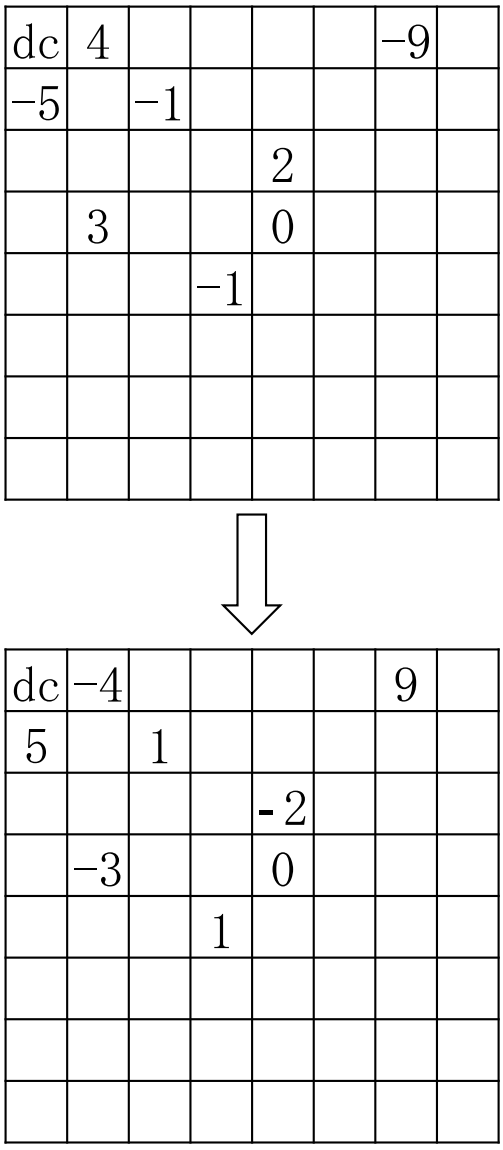


Fig. 2.2 An example of AC coefficient sign inversion processing in an 8×8 block

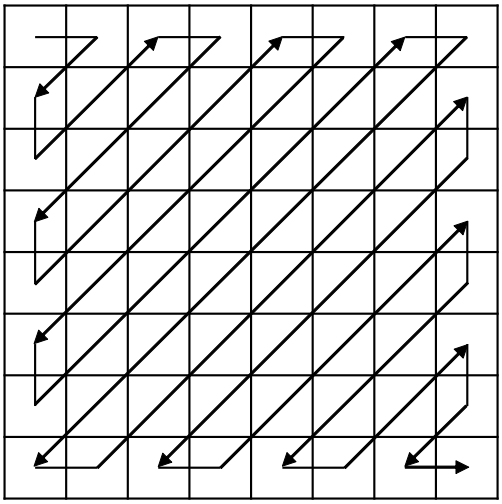


Fig. 2.3 Zigzag-scan in an 8×8 block

coefficient is called level, and whether the non-zero coefficient is the last one is called last. (last, run, level) is considered as a pair, and a variable-length code is assigned to the pair. After processing all non-zero coefficients, the processing related to the processing of qDCTC is finished. After all other processing is finished, the encoding process for MPEG-4 is finished, and a processed output video is obtained. At the same time, a key file for restoration is also output. An example of the original video and output video by the Y component AC coefficient sigh inversion processing is shown in Fig. 2.4.

The reverse process is performed when decoding from the output video to the uncompressed video. Using the key file, the reverse procedure of the proposed process is performed. During the decoding process, once qDCTC is obtained, the proposed processing is removed for the AC coefficients in the same position as during the proposed process. After the remaining decoding process, the restored uncompressed video is output.

Fig. 2.4 An example of the Y component AC coefficient sigh inversion processing (upper: the original video, lower: the output video)

3

Data hiding techniques

3.1 Impact on Data Size

Most natural (i.e., non-artificial) images exhibit a high degree of spatial redundancy. Therefore, orthogonal transformations are applied to mitigate the effects of this redundancy. Consequently, the values of the AC coefficients exhibit a tendency towards a Laplacian distribution where the location parameter $\mu = 0$, as illustrated in Fig.

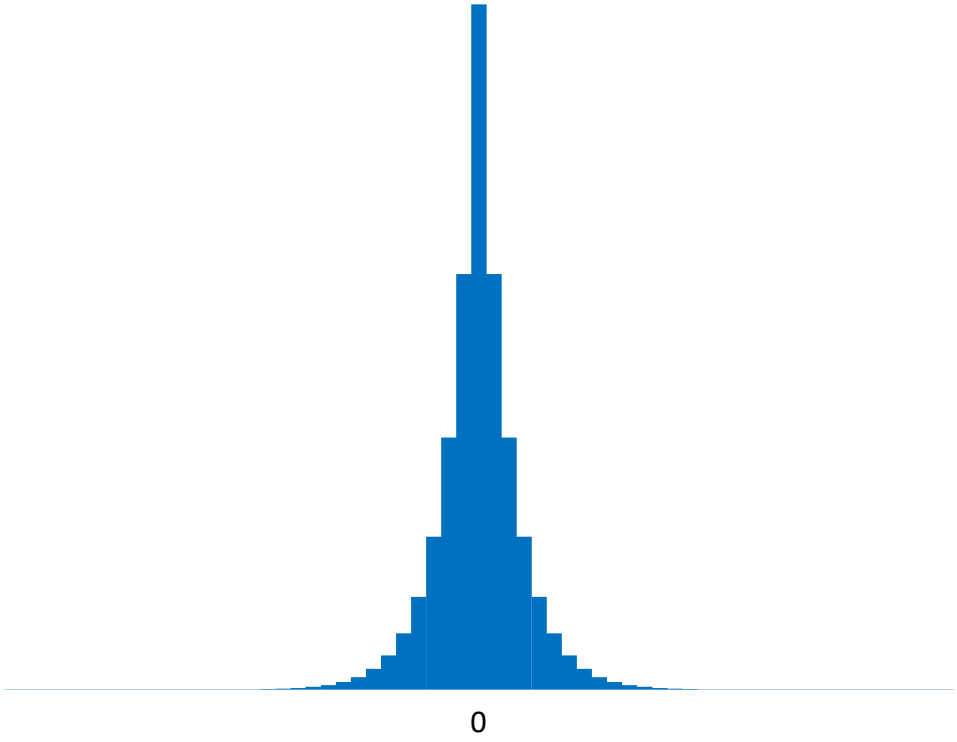


Fig. 3.1 Laplacian distribution of the AC coefficients.

3.1 [40, 41]. Leveraging the characteristics of the Laplacian distribution, the quantized transform coefficients (qTCs) will then be encoded by employing two entropy coding techniques: CAVLC (Context Adaptive Variable Length Coding) or CABAC (Context Adaptive Binary Arithmetic Coding). MPEG-2 solely supports CAVLC, HEVC exclusively supports CABAC, while H.264/AVC accommodates both. In general, entropy coding methods allocate shorter codes to more frequently occurring data.

In the case of CAVLC, qTCs are decomposed into level values and zero run-length values prior to the application of CAVLC. Subsequently, the level values and the zero run-length values are coded (as run-level pairs in MPEG-2; independently in H.264/AVC) utilizing the VLC tables. According to these tables, as long as the quantity of zero coefficients within a block remains constant, the data size will diminish when the absolute values of the AC coefficients decrease.

In the case of CABAC, only the magnitudes and positions of the non-zero AC coefficients are coded. These values are decomposed into several syntax elements and then binarized and coded. Analogous to the CAVLC case, as long as the quantity of zero coefficients within a block remains constant, the data size will diminish when the absolute values of the AC coefficients decrease.

Consequently, in order to reduce the data size, it is judicious to decrease the absolute values of the non-zero AC coefficients without diminishing the quantity of zero AC coefficients.

Fig. 3.2 Example of classical LSB replacement technique while embedding 1.

3.2 Classical LSB Replacement Technique

For a given data element, the bit planes can be organized in a sequence from the most significant to the least significant bit, with the final bit in this arrangement being denoted as the LSB (Least Significant Bit). As an illustration, the rightmost bit of an eight-bit binary number represents the LSB of that number. In the classical LSB replacement technique [42], the LSB of the target data intended for concealment will be substituted with one bit of supplementary data, potentially altering the original value of the target by a unit of one. The embedded single bit of data can be extracted by directly referencing the LSB of the target. A demonstrative example of the classical LSB replacement technique is depicted in Fig. 3.2.

3.3 Quantitative degradation adjustment method based on squared error

This section explains the principle of the quantitative degradation degree adjustment method based on the mean squared error (MSE) [43, 44]. The MSE value is calculated as follows:

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(b_{ei} - b_i)^2\,, \tag{3.1}$$

where b_{ei} indicates the i-th bit of the extracted data and b_i indicates the i-th bit of S.

These methods are for JPEG images, but since they process qDCTC, they can also be applied to video formats such as MPEG-4 that use DCT like JPEG. One of the objective image quality evaluation metrics is PSNR (Peak Signal-to-Noise Ratio) based on mean squared error. PSNR is calculated by equation 3.2, where $E[x]$ represents the average value of x, and e_{PIX} represents the error in pixel space.

$$\mathrm{PSNR} = 10\log_{10}\frac{255^2}{E[{e_{PIX}}^2]}\,. \tag{3.2}$$

Since DCT is an orthogonal transform, according to the law of conservation of energy, the energy before DCT and the energy after DCT are equal. In other words, the squared error in the pixel space before DCT and the squared error in the frequency space after DCT are equal. The user sets the value of TPSNR (Target PSNR) as an index for quantitatively adjusting the degree of degradation. Since the setting of TPSNR makes the squared error N^2 that should be caused to one 8×8 block DCT coefficient equal to the squared error $\sum {e_{PIX}}^2$ in the pixel space, it can be calculated by calculating back from Equation (3.2) as follows:

$$\mathrm{TPSNR} = 10\log_{10}\frac{64 \times 255^2}{\sum {e_{PIX}}^2}\,, \tag{3.3}$$

$$\sum {e_{PIX}}^2 = \frac{64 \times 255^2}{10^{\mathrm{TPSNR}/10}} , \tag{3.4}$$

$$N^2 = \frac{64 \times 255^2}{10^{\mathrm{TPSNR}/10}} . \tag{3.5}$$

This N^2 is called the necessary squared error. According to N^2, the squared error that occurs for one block is processed so that it reaches the necessary squared error. For example, in the case of AC coefficient sign inversion, the squared error E^2 caused by sign inversion of one dequantized AC coefficient A is expressed by the following equation.

$$E^2 = 4 \times A^2 . \tag{3.6}$$

AC coefficient sign inversion processing is performed in a predetermined order until $\sum E^2$ reaches N^2 or all AC coefficients are processed. It is expected that this processing will make the squared error $\sum E^2$ that occurs in a block close to N^2, as long as the TPSNR setting is appropriate. Therefore, it is expected that the PSNR value of the entire JPEG image will be close to the user-specified TPSNR value after all blocks have been processed.

4

Proposed method 1: Scrambling technique embedding audio into video

4.1 Overview

Popular video streaming services employ codecs such as MPEG-4 and H.264 for video and AAC and MP3 for audio. These codecs utilize the Discrete Cosine Transform

(DCT), similar to JPEG and MPEG/MPEG-2, allowing for processing in the DCT coefficient domain applicable to modern video streaming services.

Within MPEG-4 and H.264, macroblocks (MBs) are categorized into two types: intra-MB and inter-MB. According to Nemoto et al.'s research [45], inter-MBs employ frame-to-frame prediction, leading to the propagation of the scrambling effect from referenced intra-MBs. Consequently, scrambling only intra-MBs propagates this scrambling effect to inter-MBs. However, scrambling solely inter-MBs leaves intra-MBs unscrambled, and scrambling both intra-MBs and inter-MBs risks uncontrollable scrambling effects in inter-MBs, potentially resulting in varying degradation levels. Therefore, this method proposes scrambling only intra-MBs.

Furthermore, MPEG-4 and H.264 employ three frame types: I (Intra) frames, P (Predictive) frames, and B (Bidirectionally Predictive) frames. While I frames consist entirely of intra-MBs, P and B frames comprise both intra-MBs and inter-MBs, with the proportion of intra-MBs to inter-MBs varying significantly across videos. Since human perception is more sensitive to luminance (Y) compared to chrominance (Cb, Cr), this method applies partial scrambling to the Y component of intra-MBs. When embedding audio data into DCT coefficients, full data embedding is necessary for reconstruction. This requires calculating the minimum number of bits to embed in a single Y block, necessitating analysis of the minimum number of Y blocks and audio data bitrate in a video. Popular video streaming services typically employ video resolutions of 1280 $\times$ 720 and 1920 $\times$ 1080, frame rates of 24fps, 25fps, and 30fps, and audio bitrates of 128kbps and 192kbps. Additionally, MPEG-4 and H.264 employ the Group of Pictures (GOP) frame structure. While the number of frames within a GOP is variable, 12 frames

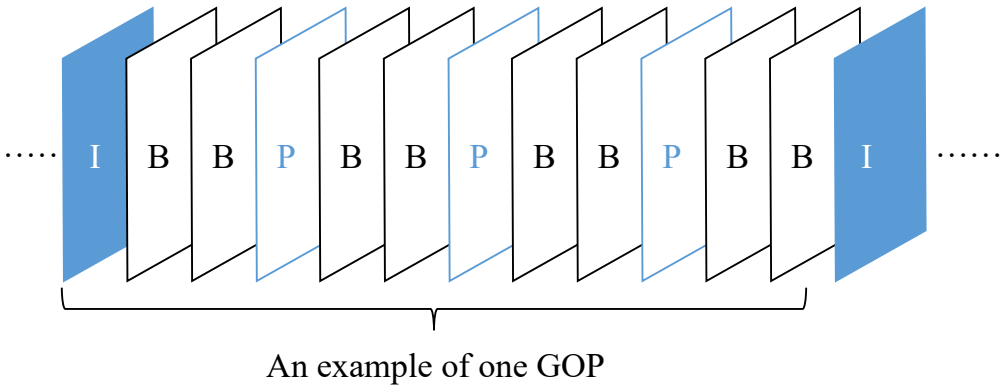


Fig. 4.1 structure of a 12-frame GOP

Table 4.1 Minimum number of bits that can be embedded in a single Y block [bits]

video \ audio	128kbps	192kbps
720-24p	5	7
720-25p	5	7
720-30p	4	6
1080-24p	2	3
1080-25p	2	3
1080-30p	2	3

is the default. Figure 4.1 illustrates the structure of a 12-frame GOP.

Using the parameters described above, the number of bits that can be embedded in a single Y block was calculated. The results are shown in Table 4.1. From these results, it can be concluded that if at least 7 bits are embedded in a single Y block, all of the audio data can be embedded. In this study, 8 bits (1 byte) of audio data are embedded in a single Y block for simplicity of processing.

On the encoding side, the input file, which is an uncompressed audio-embedded video, is first separated into audio and video data. The audio data is encoded into a

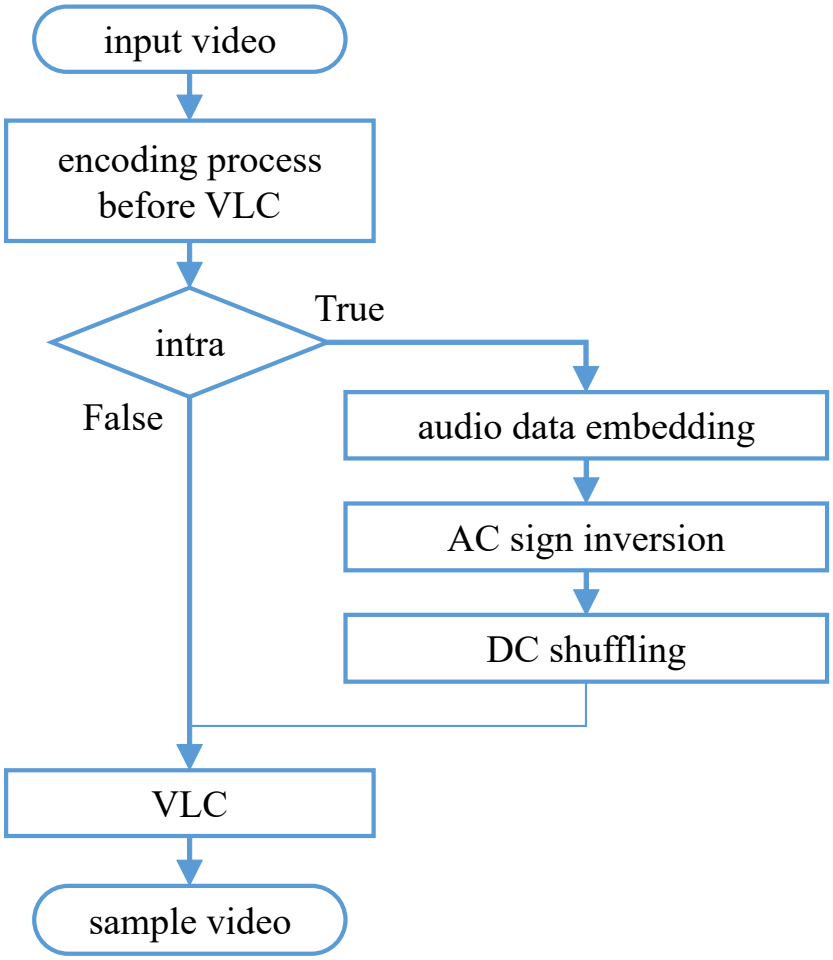


Fig. 4.2 Flowchart for encoding

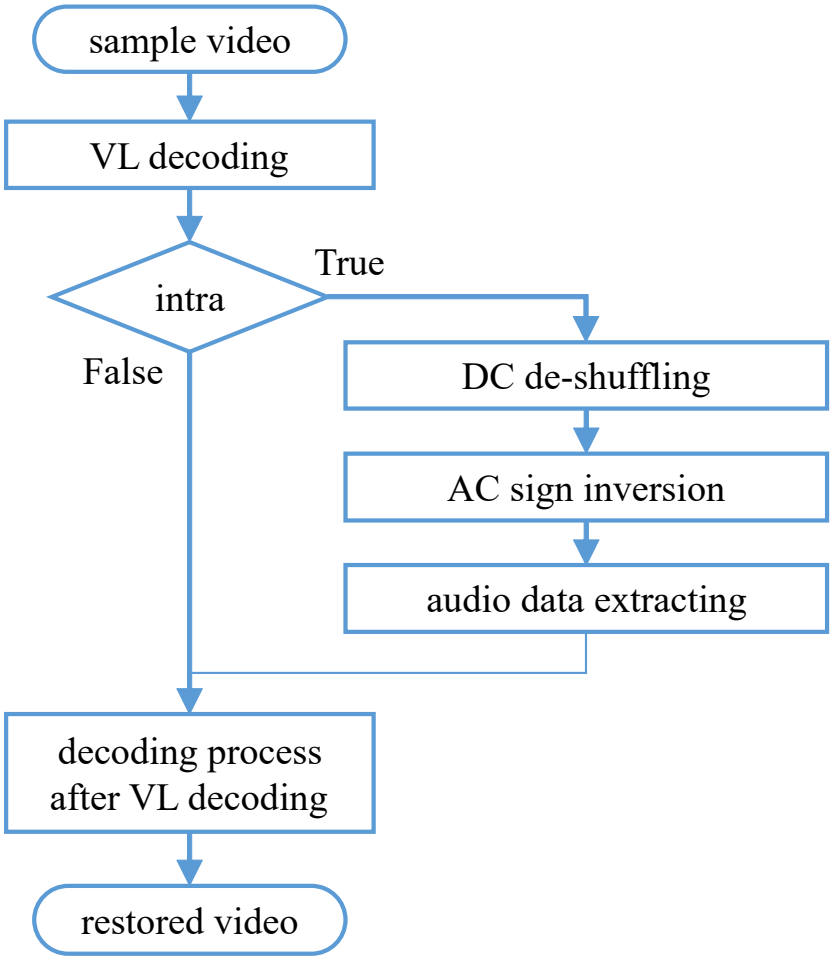


Fig. 4.3 Flowchart for decoding

suitable format, such as PCM or MP3, as needed. The video data is encoded using the MPEG-4 format, and the quantization DCT coefficients (qDCTC) are extracted. Partial scrambling processing and audio data embedding processing are performed on the qDCTC. For partial scrambling processing, DC coefficient shuffling is performed within the MB, and sign inversion is performed for AC coefficients. For embedding processing, the newly proposed method is used to embed the encoded audio data into the AC coefficients. The details of each process are explained in the following sections. After partial scrambling processing and audio data embedding processing, the remaining steps of the encoding process are performed to output the sample video. The flowchart for the encoding side is shown in Fig. 4.2.

On the decoding side, the sample video is decoded using the normal MPEG-4 decoding process, and the qDCTC are extracted. Partial scrambling removal processing and audio data extraction processing are performed on the qDCTC. After that, the remaining steps of the decoding process are performed to output the restored video. The flowchart for the decoding side is shown in Fig. 4.3.

4.2 Audio data hiding

4.2.1 Twice-plus

In this technique, since it is necessary to embed 8 bits of data into one block, the AC component after quantization in the DCT domain of the Y block is targeted for embedding.

For the audio data embedding process, a new embedding method, the TP (Twice-Plus) method, is proposed in consideration of the drawbacks of conventional methods

[46, 47]. The TP method is a method that doubles an integer and adds the 1-bit data to be embedded when embedding 1 bit of data for a certain integer.

Specifically, first, the 63 AC coefficients in one intra-Y block are represented by $A_k(k \in \{1, 2, ..., 63\})$ in raster scan order. k_{min} and k_{max} ($1 \leqslant k_{max} \leqslant 63, 1 \leqslant k_{min} \leqslant 63, k_{max} - k_{min} \geqslant 7$) are set in advance, and 8 different integers $k_1, k_2, ..., k_8$ are selected from the range of $[k_{min}, k_{max}]$ using pseudo-random numbers as the embedding positions. Among the 63 AC components, the high-frequency part is difficult for humans to perceive, while the low-frequency part is easy for humans to perceive. By utilizing this property, the degree of degradation due to embedding can be adjusted to a certain extent by adjusting kmin and kmax. In addition, by using pseudo-random numbers, the secrecy of this method can be enhanced.

The TP method utilizes the parity of integers. Let the value of an AC coefficient be A and the 1-bit audio data to be embedded be S_{in}. The 1-bit audio data S_{in} is embedded in A according to the following equation to obtain a new AC coefficient value A^*.

$$A^* = 2 \times A + S_{in} \,. \tag{4.1}$$

Since S_{in} is 1-bit data, it can only have a value of 1 or 0. If S_{in} is 0, the processed A^* becomes even, and if S_{in} is 1, the processed A^* becomes odd. Therefore, based on the parity of A^*, the embedded S_{in} value can be extracted by mod (modulo arithmetic).

The extracted 1-bit audio data S_{out} and the restored AC coefficient A^{**} are calculated by the following equations. An example of audio data embedding and extraction processing is shown in Fig. 3.4.

$$S_{out} = A^* \bmod 2\,, \tag{4.2}$$

$$A^{**} = (A^* - S_{out}) \div 2\,. \tag{4.3}$$

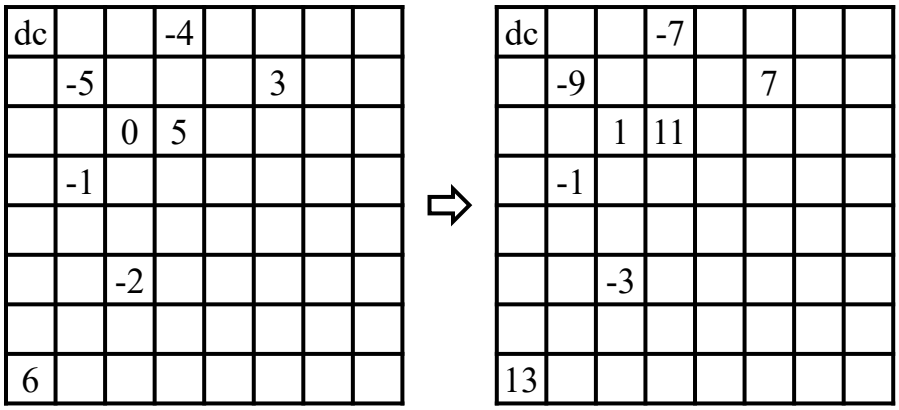
Mathematically, it can be proven using equations (4.1) to (4.3) that A and A^{**} are equal and S_{in} and S_{out} are equal. Therefore, it can be proven that the audio data embedding process in this study is reversible for both video and audio.

4.2.2 Countermeasure of overflow

In video formats such as MPEG-4, there are upper and lower limits for the values of qDCTC that can be encoded due to VLC, and increasing the absolute value of qDCTC has the risk of overflow. In the TP method, the embedding method of this study, the AC coefficient is doubled, so the absolute value of the AC coefficient increases. Therefore, there is a possibility of overflow.

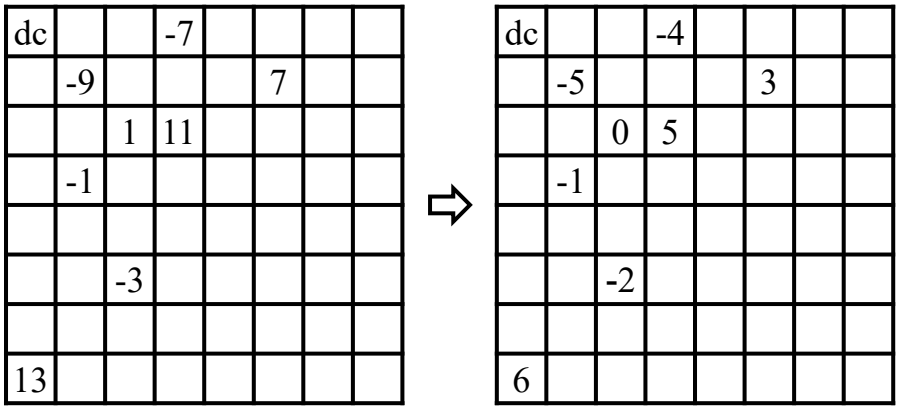
In order to ensure the reversibility of the proposed method, it is necessary to take measures against overflow. Therefore, three types of overflow countermeasures are proposed.

The first is to reduce the AC coefficient during the quantization step. Specifically, when encoding to MPEG-4, q is set to a large value. By setting q to a large value, the value of qDCTC becomes smaller overall and less than half of the upper limit, so there



While embedding 11111111

(a) embedding



While extracting 11111111

(b) extraction

Fig. 4.4 An example of audio data embedding and extraction processing

is no risk of overflow even if it is doubled.

The second is to perform preprocessing on the input video. The specific content of preprocessing is to manipulate the pixel values of the input video and make all the AC coefficients of the embedding positions less than half of the upper limit after DCT. This method is effective, but it has the disadvantage that the input video is slightly deteriorated by preprocessing.

The third is to avoid low-frequency components and embed intermediate-frequency and high-frequency components when embedding. In general, the low-frequency components of AC coefficients have relatively large values, and the values become relatively small as they become high-frequency components. By utilizing this property, it is considered possible to greatly reduce the possibility of overflow by avoiding the low-frequency components that are easy to overflow due to audio data embedding processing, setting a threshold, and embedding processing only for certain high-frequency components. However, even if the possibility of overflow can be reduced, it cannot be reduced to 0, so it cannot be said to be a complete countermeasure against overflow.

The first method is the method that is actually used in the processing of this study and is effective. On the other hand, it is also considered that a combination of the second and third methods can be effective as an overflow countermeasure.

4.3 Partial scrambling and descrambling

4.3.1 AC sign inversion

In this study, sign inversion is performed on the AC coefficients during partial scrambling processing. Compared to other methods, it has the advantage of having a small

effect on the file size of the AC coefficients, but it also has the disadvantage of having a relatively low level of secrecy. Therefore, in this method, pseudo-random numbers are used to enhance the secrecy and compensate for this disadvantage.

When performing sign inversion of AC coefficients, the number of sign flips $N(1 \leqslant N \leqslant 63)$ and the range $[i_{min}, i_{max}](1 \leqslant i_{max} \leqslant 63, 1 \leqslant i_{min} \leqslant 63, i_{max} - i_{min} \geqslant N_i - 1)$ are set as parameters to adjust the degradation. Then, N different integers $i_1, i_2, ..., i_N$ are selected from the range $[i_{min}, i_{max}]$ using pseudo-random numbers as the sign inversion positions.

Once the sign inversion positions are determined, sign inversion is performed on the AC coefficients. For the AC coefficient A to be sign-flipped, the following equation is used to perform the sign inversion and obtain the sign-flipped AC coefficient A^*.

$$A^* = -A\,. \tag{4.4}$$

When sign inversion of the AC coefficients is removed, the sign-flipped AC coefficients are flipped again to restore the original AC coefficients. The following equation is used to calculate the AC coefficients A^{**} after sign inversion removal using the sign-flipped AC coefficients A^*.

$$A^{**} = -A^*\,. \tag{4.5}$$

Mathematically, it can be proven using equations (4.4) and (4.5) that A and A^{**} are

equal. Therefore, it can be proven that the sign inversion of AC coefficients in this study is reversible.

4.3.2 DC SHUFFLING

In this study, DC coefficient shuffling within MB is performed during partial scrambling processing. In consideration of the drawbacks of conventional methods such as the method of Takayama et al. [48], the range of DC component shuffling is limited to within the MB, and shuffling processing is performed to suppress the increase in data volume. In the YUV-420p color space commonly used in VOD services, the shuffle table is contained in 6 integers, and the shuffle target is further limited to the Y block, which is more perceptible to humans, so the shuffle table can be contained in 4 different integers $T_0, T_1, T_2, T_3 (0 \leqslant T_0, T_1, T_2, T_3 \leqslant 3)$.

In the MB-internal DC shuffling process, the following equations are used to calculate the shuffled four DC coefficients ${D_{T_0}}^*, {D_{T_1}}^*, {D_{T_2}}^*, {D_{T_3}}^*$ from the four DC coefficients D_0, D_1, D_2, D_3 in one intra-MB.

$$ {D_{T_0}}^* = D_0 \,, \tag{4.6} $$

$$ {D_{T_1}}^* = D_1 \,, \tag{4.7} $$

$$D_{T_2}{}^{*} = D_2\,, \tag{4.8}$$

$$D_{T_3}{}^{*} = D_3\,. \tag{4.9}$$

When performing MB-internal DC de-shuffling, the following equations are used to calculate the restored four DC coefficients $D_0{}^{**}, D_1{}^{**}, D_2{}^{**}, D_3{}^{**}$.

$$D_0{}^{**} = D_{T_0}{}^{*}\,, \tag{4.10}$$

$$D_1{}^{**} = D_{T_1}{}^{*}\,, \tag{4.11}$$

$$D_2{}^{**} = D_{T_2}{}^{*}\,, \tag{4.12}$$

$$D_3{}^{**} = D_{T_3}{}^{*}\,. \tag{4.13}$$

Mathematically, it can be proven using equations (4.6) to (4.9) and equations (4.10) to (4.13) that D_0, D_1, D_2, D_3 and ${D_0}^{**}, {D_1}^{**}, {D_2}^{**}, {D_3}^{**}$ are each equal, respectively. Therefore, it can be proven that the MB-internal DC shuffling process in this study is reversible. Figure 4.5 shows an example of a DC shuffle table and a DC shuffle process using the DC shuffle table.

In this study, three processes of audio data embedding, AC coefficient sign inversion, and MB-internal DC shuffling are performed in the process of creating a sample video. All three of these processes have been mathematically proven to be reversible. Therefore, the processing in this study is theoretically reversible, and it is possible to restore the original video from the sample video.

4.4 Adjustable degradation degrees

This study aims to protect the copyright of digital content in VOD services. In actual VOD services, various videos with audio are provided, so it is considered necessary to adjust the video quality of sample videos as needed. If a method does not implement degradation degree adjustment, it is likely that it will not be able to cope with different application scenarios. Therefore, one of the goals of this method is to achieve degrada-

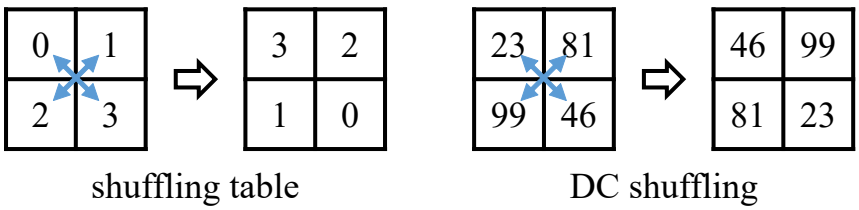


Fig. 4.5 An example of a DC shuffle table and a DC shuffle process using the DC shuffle table

tion degree adjustment.

There are two types of degradation degree adjustment: qualitative and quantitative, each of which has its own application scenarios.

Qualitative degradation degree adjustment is generally realized by setting several degradation degrees in advance using parameters and selecting one of the several degradation degrees when applying it. Qualitative degradation degree adjustment has the characteristics of being easy to understand, easy to set, and easy to apply, and is convenient for people who do not have much expertise. However, since qualitative degradation degree adjustment can only set the degradation degree in a few steps, it has the drawback that precise degradation degree adjustment cannot be performed.

Quantitative degradation degree adjustment adjusts the degradation degree by specifying a specific degradation degree scale. Therefore, the degradation degree can be set arbitrarily within the range of the achievable degradation degree. Unlike qualitative degradation degree adjustment, which can only set the degradation degree in a few steps, it can achieve relatively precise degradation degree adjustment. By fine-tuning the degradation degree after one adjustment, it is possible to achieve a degradation degree closer to the ideal. On the other hand, the numerical value of the degradation degree is difficult to understand intuitively compared to the qualitative phase setting such as strong and weak, and it is considered that it may lack the convenience of use for people who do not have a certain level of expertise.

Therefore, this study realizes both qualitative and quantitative degradation degree adjustment to cope with different application scenarios. In qualitative degradation degree adjustment, three-step qualitative degradation degree adjustment is realized by set-

ting several parameters in audio embedding processing and partial scrambling processing. In quantitative degradation degree adjustment, it is realized by setting TPSNR. Figure 4.6 shows a comparison of the two.

4.4.1 Qualitatively adjustable degradation degrees

In this study, qualitative degradation degree adjustment is realized by setting parameters for the three processes of audio data embedding, AC coefficient sign inversion, and MB-internal DC shuffling.

In audio data embedding processing, there are two parameters, k_{min} and k_{max}, that indicate the embedding range. If k_{min} and k_{max} are set small, the embedding range approaches the low-frequency component, and the degradation degree becomes stronger. Conversely, if k_{min} and k_{max} are set large, the embedding range approaches the high-frequency component, and the degradation degree becomes weaker. An example of the range setting for audio data embedding processing is shown in Fig. 4.7.

In AC coefficient sign inversion processing, there are three parameters: the number of sign flips N and the range k_{min} and k_{max}. If N is set large, the change in the AC coefficient becomes larger, and the degradation degree becomes stronger. Conversely, if N is set small, the change in the AC coefficient becomes smaller, and the degradation

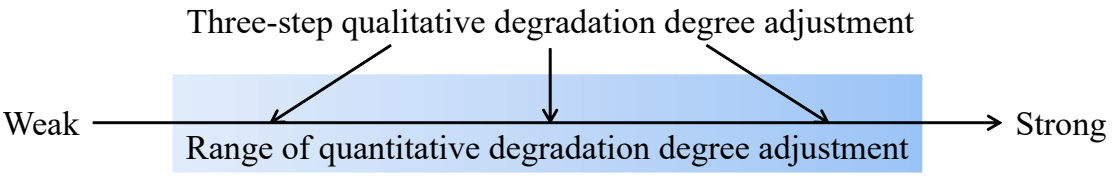


Fig. 4.6 Comparison of qualitative and quantitative degradation degree adjustment

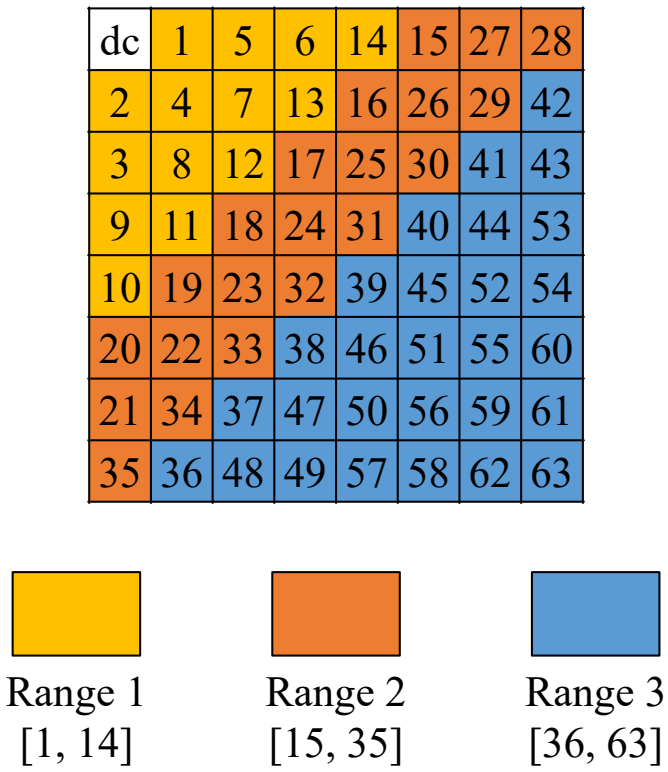


Fig. 4.7 An example of the range setting for audio data embedding processing

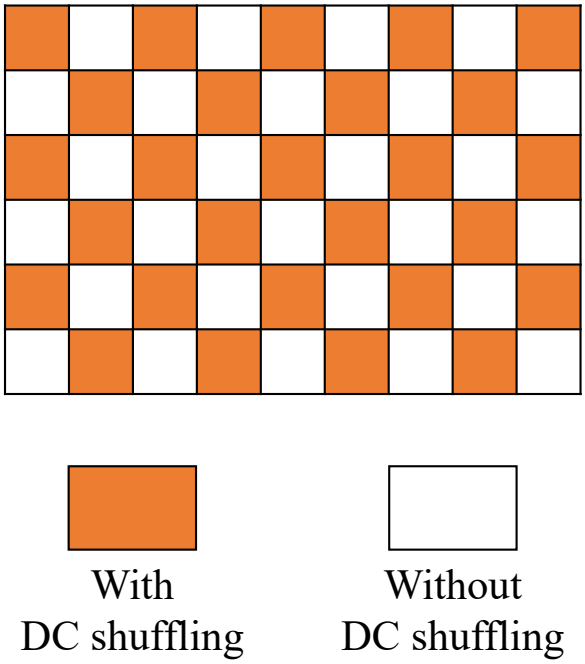


Fig. 4.8 An example of MB-internal DC shuffling processing when $D = 2$

degree becomes weaker.

If i_{min} and i_{max} are set small, the range of AC coefficient sign inversion approaches the low-frequency component, and the degradation degree becomes stronger. Conversely, if i_{min} and i_{max} are set large, the range of AC coefficient sign inversion approaches the high-frequency component, and the degradation degree becomes weaker.

In MB-internal DC shuffling processing, the shuffle table is fixed, and instead of performing the processing on all intra-MBs in the video's image data, the MB-internal DC shuffling processing is performed only on a subset of intra-MBs. The proportion of intra-MBs on which MB-internal DC shuffling processing is performed is set to $1/D(D \geqslant 1)$, and processing is performed only on $1/D$ of intra-MBs. Here, the integer D is manipulated to adjust the degradation degree. If D is set small, the change in

the DC coefficient becomes larger, and the degradation degree becomes stronger. If D is set large, the change in the DC coefficient becomes smaller, and the degradation degree becomes weaker. An example of MB-internal DC shuffling processing by setting D is shown in Fig. 4.8.

In summary, to set the degradation degree to strong, set k_{min}, k_{max} and i_{min}, i_{max} to small values, and set N and D to large values. Conversely, to set the degradation degree to weak, set k_{min}, k_{max} and i_{min}, i_{max} to large values, and set N and D to small values.

Here is the procedure for implementing qualitative degradation degree adjustment in this study:

i. Preliminary Operations

   Record the file size of the audio data as f (unit: bytes) and determine the shuffle tables T_0, T_1, T_2, T_3 for each degradation level (use 3, 2, 1, 0 as defaults). Determine the parameters D, k_{min}, k_{max} and N, i_{min}, i_{max}.

ii. Random Number Generation and Parameter Calculation

   Select one of the multiple degradation levels. Generate pseudo-random numbers using PRNG with seed. Generate 8 different integers from the range $[k_{min}, k_{max}]$ using the generated pseudo-random numbers, and represent them as $k_1, k_2, ..., k_8$. Generate N different integers from the range $[i_{min}, i_{max}]$ using the generated pseudo-random numbers, and represent them as $i_1, i_2, ..., i_N$.

iii. Audio Data Reading

   Read the audio data 8 bits at a time and store it as $S_1, S_2, ..., S_8$.

iv. Obtaining qDCTC

Obtain qDCTC for the first intra-MB of the video data.

v. DC Coefficient Shuffling

If this intra-MB is determined to be a DC shuffling processing MB by D, shuffle the DC coefficients in the four Y blocks within the MB according to the shuffle tables T_0, T_1, T_2, T_3 and equations (4.6) to (4.9).

vi. Embedding and Sign Inversion Operations

For each of the four Y blocks, perform the following operations: a. Embed S_n into A_{k_n} according to equation (4.1) for $n = 1, 2, ..., 8$. b. Perform sign inversion processing on A_{i_n} according to equation (4.4) for $n = 1, 2, ..., N$. c. Read 8 bits of audio data. If the data read is successful, store the read data as $S_1, S_2, ..., S_8$. If the data read is unsuccessful (all audio data has been embedded), set $S_1 = S_2 = ... = S_8 = 0$.

vii. Intra-MB Search

Search for the next intra-MB. a. If found, proceed to viii. b. If not found, proceed to ix.

viii. Repetition of Operations

Move to the found intra-MB and repeat v, vi, and vii.

ix. Completion and Data Recording

Record the data indicating the completion of the process, f, seed, and the selected degradation level into a key file.

By following the above procedure, an embedded audio sample video and a key file can be generated. Depending on the operational format of the VOD service provider, it is also possible to minimize the file size of the sample video by not including audio and preview a portion of the audio data by leaving a portion of the audio data in the sample video.

In addition, since the above operations are reversible, audio data extraction processing and partial scrambling processing are possible. The specific procedures are as follows:

i. Parameter Determination and Random Number Generation

Read the data values for f, seed, and the selected degradation level from the key file. Based on the recorded degradation level, set the shuffle tables T_0, T_1, T_2, T_3 and the parameters D, k_{min}, k_{max} and N, i_{min}, i_{max} using the predetermined degradation degree adjustment parameters. Generate pseudo-random numbers using PRNG with seed. Generate 8 different integers from the range $[k_{min}, k_{max}]$ using the generated pseudo-random numbers, and represent them as $k_1, k_2, ..., k_8$. Generate N different integers from the range $[i_{min}, i_{max}]$ using the generated pseudo-random numbers, and represent them as $i_1, i_2, ..., i_N$.

ii. Obtaining qDCTC

Obtain qDCTC for the first intra-MB of the video data.

iii. DC Deshuffling Processing

Perform DC deshuffling processing on the four Y blocks within the MB using equations (4.10) to (4.13) and the shuffle tables T_0, T_1, T_2, T_3.

iv. Audio Data Extraction and AC Coefficient Restoration

For each of the four Y blocks, perform the following operations: a. Perform sign inversion removal processing on A_{i_n} according to equation (4.5) for $n = 1, 2, ..., N$. b. Extract audio data S_n according to equation (4.18) for $n = 1, 2, ..., 8$ and restore AC coefficients A_{k_n} according to equation (4.3). c. If the file size of the audio data is less than f, write $S_1, S_2, ..., S_8$. If the file size of the audio data is greater than or equal to f, do not write it.

v. Intra-MB Search

Search for the next intra-MB. a. If found, proceed to vi. b. If not found, proceed to vii.

vi. Repetition of Operations

Move to the found intra-MB and repeat iii, iv, and v.

vii. Completion and Output

Complete the processing and output the restored video data and audio data.

By following the above procedures, VOD service users can view the original video with audio.

4.4.2 Quantitative adjustable degradation degrees

The quantitative degradation adjustment method in this study is a method that adjusts the quantitative degradation based on the mean squared error using TPSNR. It is realized by setting TPSNR and bringing PSNR closer to TPSNR. PSNR is calculated by Equation

3.2. Unlike conventional methods [43, 44] that process on a block basis, this study processes on an MB block basis. Therefore, after setting TPSNR, the mean squared error ${E_{MB}}^2$ that should be caused by one 16×16 MB is calculated by the following equation:

$$ {E_{MB}}^2 = \frac{256 \times 255^2}{10^{\mathrm{TPSNR}/10}} . \tag{4.14} $$

Let the 64 qDCTs in a Y block of an intra-MB be $D, A_1, A_2, ..., A_{63}$. Inverse quantize this block and let the inverse quantized DCT coefficients be $D', {A_1}', {A_2}', ..., {A_{63}}'$. By dividing $D', {A_1}', {A_2}', ..., {A_{63}}'$ by $D, A_1, A_2, ..., A_{63}$, I obtain $Q_0, Q_1, Q_2, ...Q_{63}$, which is called the scale table. Here, one AC coefficient A_j that has been decided in advance is also doubled. This A_j is an AC coefficient that determines whether or not DC shuffle processing has been performed. The mean squared error ${E_j}^2$ caused by doubling A_j is calculated using the following equation where Q_j is the value of the scale table corresponding to A_j:

$$ {E_j}^2 = {A_j}^2 \times {Q_j}^2 . \tag{4.15} $$

The mean squared error ${E_k}^2$ caused by embedding one audio data S into the AC coefficient A is calculated using the following equation using the value Q of the scale table corresponding to A:

$$E_k{}^2 = (A + S)^2 \times Q^2 \,. \tag{4.16}$$

The mean squared error $E_i{}^2$ caused by sign inversion processing of one AC coefficient A is calculated using the following equation using the value Q of the scale table corresponding to A:

$$E_i{}^2 = A^2 \times 4 \times Q^2 \,. \tag{4.17}$$

The total value E^2 of the mean squared errors caused by doubling A_j, all audio data embedding processing, and all AC coefficient sign inversion processing in one intra-MB is calculated from equations (4.15) to (4.17) by the following equation:

$$E^2 = E_j{}^2 + \sum E_k{}^2 + \sum E_i{}^2 \,. \tag{4.18}$$

If the shuffle tables are T_0, T_1, T_2, T_3, the mean squared error $E_D{}^2$ caused by MB-level DC shuffle processing in one MB is calculated by the following equation:

$$E_D{}^2 = \sum_{n=0}^{3} (D_n - D_{T_n}) \times {Q_0}^2 + (2 \times A_j + 1) \times {Q_j}^2 \,. \tag{4.19}$$

The processing procedure for achieving quantitative degradation adjustment in this

study is shown below.

i. Preprocessing

As a preliminary step, the file size of the audio data is recorded as f (unit: bytes), and a TPSNR step table is created to select the parameters to be used for each degradation level according to the TPSNR value. For each degradation level, the shuffle tables T_0, T_1, T_2, T_3 are determined (3, 2, 1, 0 are used as defaults). The parameters D, k_{min}, k_{max} and N, i_{min}, i_{max} are determined.

ii. TPSNR Setting and Parameter Selection

Set the TPSNR value and calculate ${E_{MB}}^2$ using equation (4.14). Also, using the TPSNR step table, select the parameters to be used from each degradation level according to the range of the TPSNR value.

iii. Random Number Generation

Generate pseudo-random numbers using a PRNG with a seed, and generate 9 different integers from the range $[k_{min}, k_{max}]$ with the generated pseudo-random numbers, represented by $k_1, k_2, ..., k_8, j$, and generate N different integers from the range $[i_{min}, i_{max}]$, represented by $i_1, i_2, ..., i_N$.

iv. Reading audio Data

Read the audio data 8 bits at a time and represent it as $S_1, S_2, ..., S_8$.

v. Acquiring qDCTC

Obtain the qDCTC in the first intra-MB of the video data.

vi. Processing of Four Y Blocks

Perform the following operations for each of the four Y blocks:

Double A_j. For $n = 1, 2, ..., 8$, embed S_n in A_{k_n} according to equation (4.1). For $n = 1, 2, ..., N$, perform sign inversion processing on A_{i_n} according to equation (4.4). Read the audio data 8 bits at a time. If the data read is successful, represent the read data as $S_1, S_2, ..., S_8$. If the data read is unsuccessful (all audio data has already been embedded), set $S_1 = S_2 = ... = S_8 = 0$.

vii. Calculation of E^2

Calculate E^2 using equation (4.18).

viii. Calculation of ${E_D}^2$

Calculate ${E_D}^2$ using equation (4.19).

ix. DC Shuffle Processing

Compare ${E_D}^2$ and $2 \times ({E_{MB}}^2 - E^2)$, and only if ${E_D}^2 \leqslant 2 \times ({E_{MB}}^2 - E^2)$, determine that this intra-MB is a DC shuffle processing MB. According to the shuffle tables T_0, T_1, T_2, T_3, shuffle the DC coefficients in the four Y blocks of the MB and add 1 to the doubled A_j according to equations (4.6) to (4.9).

x. Searching for the Next intra-MB

Search for the next intra-MB. If found, proceed to xi, and if not found, proceed to xii.

xi. Processing of the Found intra-MB

Move to the found intra-MB and repeat vi, vii, viii, ix, and x.

xii. Completion of Processing

Complete the processing and record f, seed, and TPSNR value to the key file.

By performing the above processing, as shown in Fig. 4.9, the mean squared error caused by the intra-MB is centered on $E_{MB}{}^2$ and fluctuates up and down, but since the total number of intra-MBs in the video is large, the average mean squared error of all intra-MBs is likely to be close to $E_{MB}{}^2$. Furthermore, as the video time increases, the number of intra-MBs also increases, so this possibility becomes higher and better results can be expected.

The following procedure is used to restore the obtained sample videos.

i. Determining Parameters

Read f, seed, and TPSNR value from the key file. Determine D, k_{min}, k_{max} and N, i_{min}, i_{max} using the TPSNR step table according to the TPSNR value and seed. Generate pseudo-random numbers using a PRNG and generate 9 different integers from the range $[k_{min}, k_{max}]$ with the generated pseudo-random numbers,

Fig. 4.9 Mean squared error: expected vs. calculated

represented by $k_1, k_2, ..., k_8, j$, and generate N different integers from the range $[i_{min}, i_{max}]$, represented by $i_1, i_2, ..., i_N$.

ii. Acquiring qDCTC

Obtain the qDCTC in the first intra-MB of the video data.

iii. Determining DC Shuffle

Determine whether DC shuffle has been performed based on the parity of A_j. Here, A_j is also restored according to equation (4.3).

iv. DC Deshuffle Processing

Only if A_j is odd, perform DC deshuffle processing using equations (4.10) to (4.13) for the four Y blocks in the MB according to the shuffle tables T_0, T_1, T_2, T_3.

v. Processing of Four Y Blocks

Perform the following operations for each of the four Y blocks:

Perform sign inversion processing on A_{i_n} according to equation (4.5) for $n = 1, 2, ..., N$. Extract the audio data S_n according to equation (4.18) and restore the AC coefficient A_{k_n} according to equation (4.3) for $n = 1, 2, ..., 8$. Write $S_1, S_2, ..., S_8$ if the file size of the audio data is less than f. Do not write if the file size of the audio data is greater than or equal to f.

vi. Searching for the Next intra-MB

Search for the next intra-MB. If found, proceed to vii, and if not found, proceed to viii.

vii. Processing of the Found intra-MB

Move to the found intra-MB and repeat iii, iv, v, and vi.

viii. Outputting Restored Data

Complete the processing and output the restored video data and audio data.

By following the above procedure, the original video is restored from the sample video. Once the restoration process is complete, VOD service billing users will be able to watch videos with audio.

5

Experiments and discussion of method 1

5.1 Overview

In order to verify the effectiveness of the method described in Chapter 4, verification experiments were conducted.

Considering the genres and types of videos actually provided by VOD services, three videos with audio were selected for the experiment: pv, mov, and news.

Pv is a music promotion video with a lot of screen movement and human subjects. Mov is a part of a movie with a lot of screen movement, but non-human objects are the main subject. News is a part of a news program with little screen movement and mainly text.

The length of each video is 30 seconds. The uncompressed experimental videos were encoded to MPEG-4 and aac without processing, and used as the original videos with audio. The video of the original video with audio is called the original video, and the audio is called the original audio. The original videos are called $pv\_0, mov\_0, news\_0$, respectively. The specifications of the original video are 720-25p, which means the resolution is 1280 $\times$ 720 and the frame rate is 25fps. The specifications of the original audio are stereo/44100Hz/192kbps.

In addition to PSNR, the SSIM (Structural Similarity) index proposed in [49] is also used as an objective measure of video degradation. SSIM is considered to be closer to human vision than PSNR. For two images **x** and **y**, SSIM is calculated by the following equation:

$$\mathrm{SSIM}(\mathbf{x},\mathbf{y}) = [l(\mathbf{x},\mathbf{y})]^{\alpha}[c(\mathbf{x},\mathbf{y})]^{\beta}[s(\mathbf{x},\mathbf{y})]^{\gamma}\,, \tag{5.1}$$

where $l(\mathbf{x},\mathbf{y})$ compares the brightness of images **x** and **y**, $c(\mathbf{x},\mathbf{y})$ compares the contrast of images **x** and **y**, and $s(\mathbf{x},\mathbf{y})$ compares the structure of images **x** and **y**. α, β and γ are parameters that define the importance of brightness, contrast, and structure, respectively. $l(\mathbf{x},\mathbf{y})$, $c(\mathbf{x},\mathbf{y})$ and $s(\mathbf{x},\mathbf{y})$ are defined by the following equations, respectively.

$$l(\mathbf{x}, \mathbf{y}) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1}\,, \tag{5.2}$$

$$c(\mathbf{x}, \mathbf{y}) = \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2}\,, \tag{5.3}$$

$$s(\mathbf{x}, \mathbf{y}) = \frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3}\,, \tag{5.4}$$

where μ_x and μ_y mean the averages of **x** and y, respectively, σ_x and σ_y mean the standard deviations of **x** and **y**, respectively, and σ_{xy} means the covariance of **x** and **y**. C_1, C_2 and C_3 are parameters to maintain the stability of $l(\mathbf{x}, \mathbf{y})$, $c(\mathbf{x}, \mathbf{y})$ and $s(\mathbf{x}, \mathbf{y})$.

In practice, when using SSIM to objectively evaluate images and videos, $\alpha = \beta = \gamma = 1$ and $C_3 = C_2/2$ are often used. In this case, SSIM is calculated by the following equation:

$$\mathrm{SSIM}(\mathbf{x}, \mathbf{y}) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}\,. \tag{5.5}$$

By default, it is set that $C_1 = 6.5025$ and $C_2 = 58.5225$. The default parameters are also used to calculate SSIM in this study. In order to eliminate the factor of image quality degradation due to MPEG-4 compression, in the experiments of this study, when

evaluating the image quality degradation of sample videos, instead of all uncompressed videos, the original videos are compared.

5.2 Experiments

5.2.1 Experiments on qualitatively adjustable degradation degree

The experiment on qualitatively adjustable degradation degree can be divided into two categories: objective verification experiment and subjective evaluation experiment. The objective verification experiment includes experiments on measuring the difference between restored video/audio and original video/audio, video quality measurement for sample videos, and file size increase rate measurement for sample videos with original audio. The subjective evaluation experiment includes experiments on subjective evaluation of degradation degree and product value.

The purpose of each experiment is as follows: experiments 1 and 6 are to investigate the reversibility of the proposed method, experiments 2, 4, and 7 are to investigate the effectiveness of the degradation degree control in this study, experiment 5 is to investigate whether the proposed method can generate samples with no product value as a partial scramble method, and experiment 3 is to investigate whether the file size of the sample video is smaller than that of the original video with audio.

The shuffle table used in the processing is $(3, 2, 1, 0)$, and the other parameter settings are shown in Table 5.1. The MOS score settings for the subjective experiment are shown in Tables 5.2 and 5.3. The number of subjects for the subjective experiment is 20, and the age range of the subjects is 20 to 55 years old.

Table 5.1 Parameters for experiments on qualitatively adjustable degradation degree

degradation degree	parameter		
	data hiding (k_{min}, k_{max})	AC sign inversion (N, i_{min}, i_{max})	DC shuffle (D)
weak	42, 49	8, 28, 35	4
medium	28, 35	8, 1, 8	2
strong	1, 8	63, 1, 63	1

Table 5.2 Degradation degree MOS of subjective evaluation experiment

MOS	degradation degree
5	none
4	not annoying
3	a little annoying
2	annoying
1	very annoying

Table 5.3 Product value MOS of subjective evaluation experiment

MOS	product value
5	exists
4	almost exists
3	partially exists
2	barely exists
1	does not exist

Table 5.4 TPSNR step table

TPSNR	strength of processing
(27dB, $+\infty$)	weak
(22dB, 27dB]	medium
(0dB, 22dB]	strong

5.2.2 Experiments on quantatively adjustable degradation degree

The experiments conducted for quantitative degradation adjustment include Experiment 6: Measurement of the difference between the restored video and audio based on MSE value and the original video and audio, and Experiment 7: Comparison of TPSNR and PSNR values for sample videos. The shuffle table used was $(3, 2, 1, 0)$, and the TPSNR step table used to set other parameters is shown in Table 5.4, and the parameter settings for each processing intensity are shown in Table 5.1. TPSNR was set to 20dB, 25dB, and 30dB for each experimental video.

5.3 Results and discussion

5.3.1 Results and discussion for experiments on qualitatively adjustable degradation degree

An objective verification experiment was conducted to compare the restored video and audio to the original video and audio. The experiment found that the MSE value of both audio and video was 0 for all three experimental videos. This indicates that the restored video and audio were indistinguishable from the original video and audio.

In addition, two other experiments were conducted to evaluate the quality of the

restored video and audio. The first experiment used PSNR and SSIM values to measure the video quality of the sample videos. The second experiment measured the file size increase rate of the sample videos with original audio. The results of these experiments are shown in Table 5.5.

In Experiment 1, since all MSE values were 0, the proposed qualitative degradation adjustment method is fully reversible for both video and audio.

In Experiment 2, for the three experimental videos, pv, mov, and news, the degradation degree changes in the same trend with the processing intensity in three steps: weak, medium, and strong. This shows that the proposed method achieves qualitative degradation adjustment. In addition, for medium and strong processing, a PSNR value of less than 25 dB was achieved for all three videos. This result shows that the proposed method is effective as a partial scrambling method.

In Experiment 3, all sample videos, especially weak and medium sample videos, have a smaller file size than the original videos with audio. This result shows that the proposed method achieves one of the research goals of reducing the total file size.

In Experiment 4, the subjective evaluation results for the degree of degradation is shown in Table 5.6, and the subjective evaluation experiment for the product value is shown in Table 5.7.

To analyze the results of Experiments 4 and 5, I show the average (Fig. 5.1) and variance (Fig. 5.2) of the MOS evaluation of the degree of degradation, as well as the comparison of the MOS of commercial value for each video (pv: Fig. 5.3, mov: Fig. 5.4, news: Fig. 5.5) from the experimental data.

From the average and variance of the MOS of the degree of degradation, I performed

Table 5.5 Results of Experiments 1 and 2

video	PSNR (dB)	SSIM	file size increase(%)
pv_weak	30.9	0.979	-8.90
pv_medium	24.0	0.893	-8.38
pv_strong	20.4	0.764	-4.84
mov_weak	30.2	0.987	-5.20
mov_medium	22.4	0.920	-4.77
mov_strong	19.5	0.843	-1.46
news_weak	24.8	0.965	-11.82
news_medium	17.0	0.781	-10.81
news_strong	13.7	0.561	-3.85

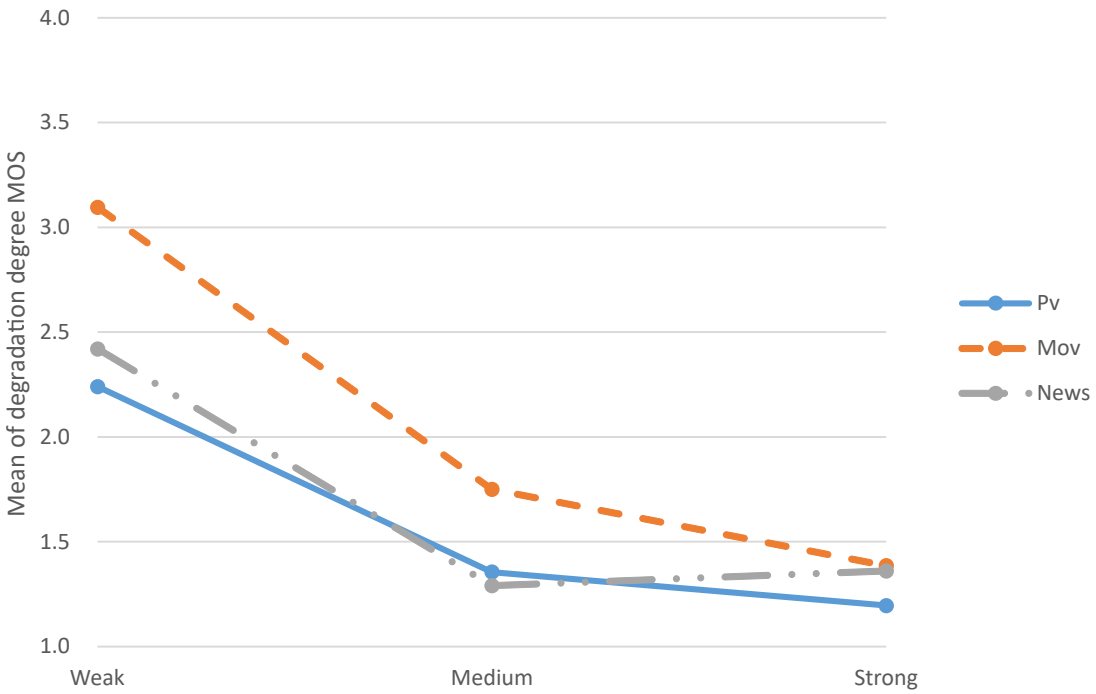


Fig. 5.1 Mean of degradation degree MOS

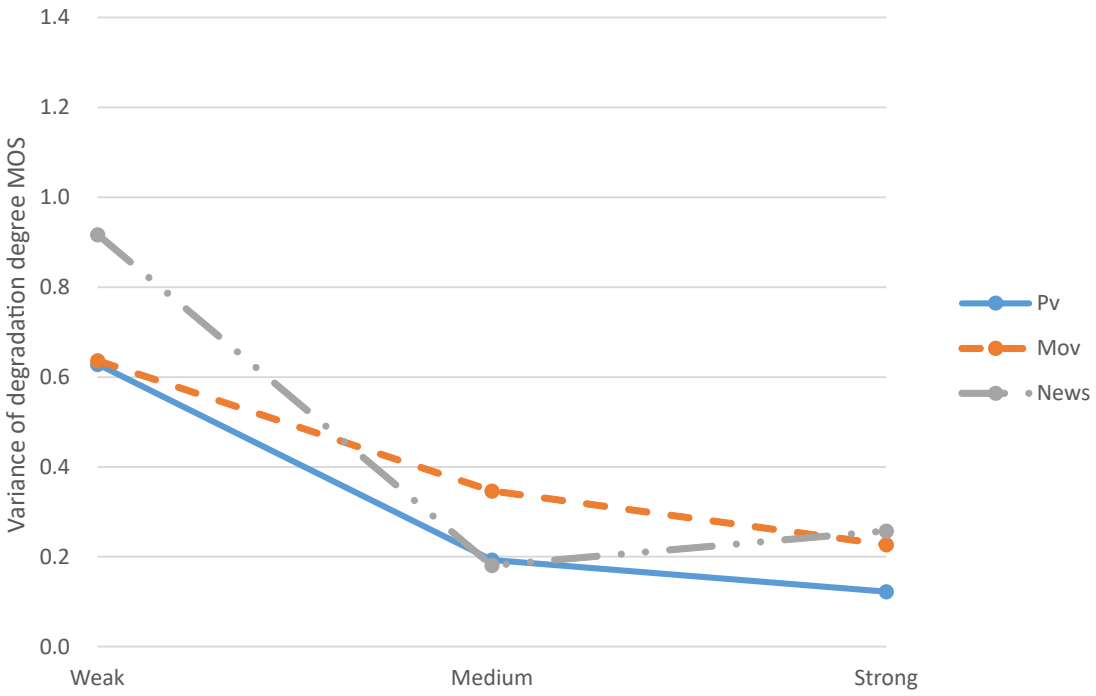


Fig. 5.2 Variance of degradation degree MOS

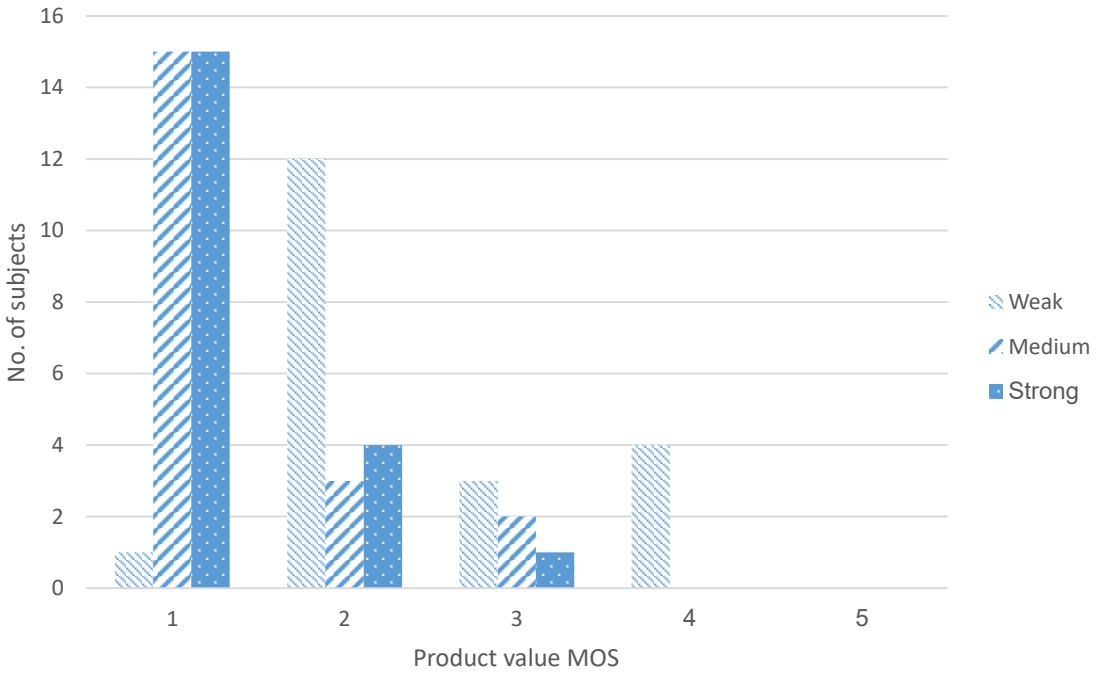


Fig. 5.3 Comparison of product value MOS for pv

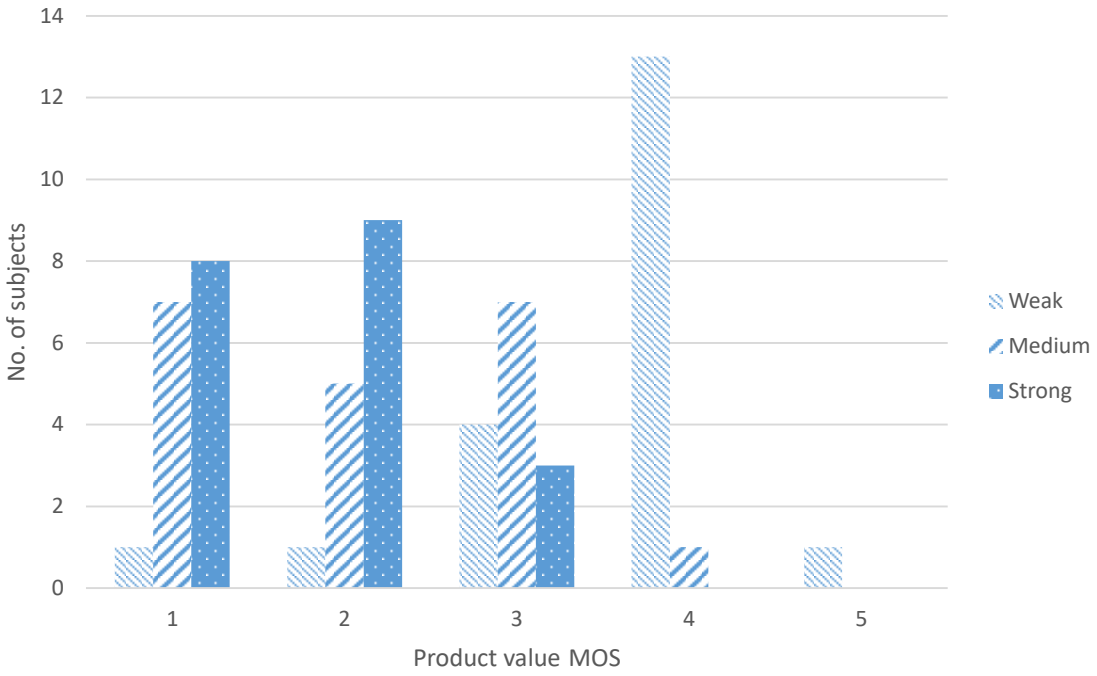


Fig. 5.4 Comparison of product value MOS for mov

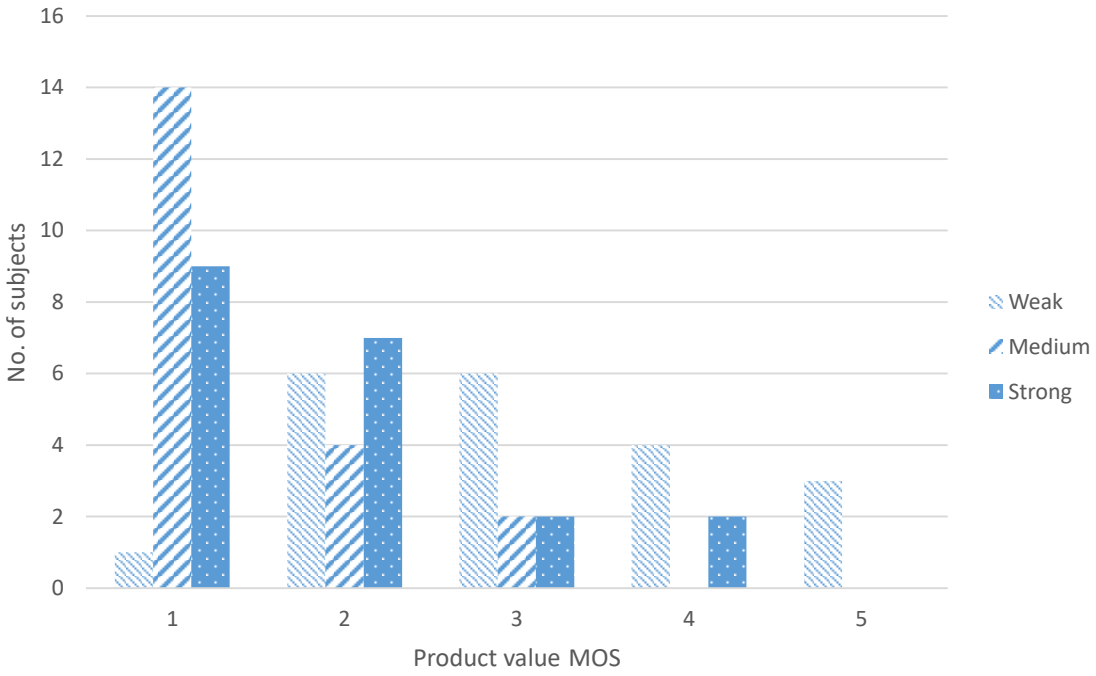


Fig. 5.5 Comparison of product value MOS for news

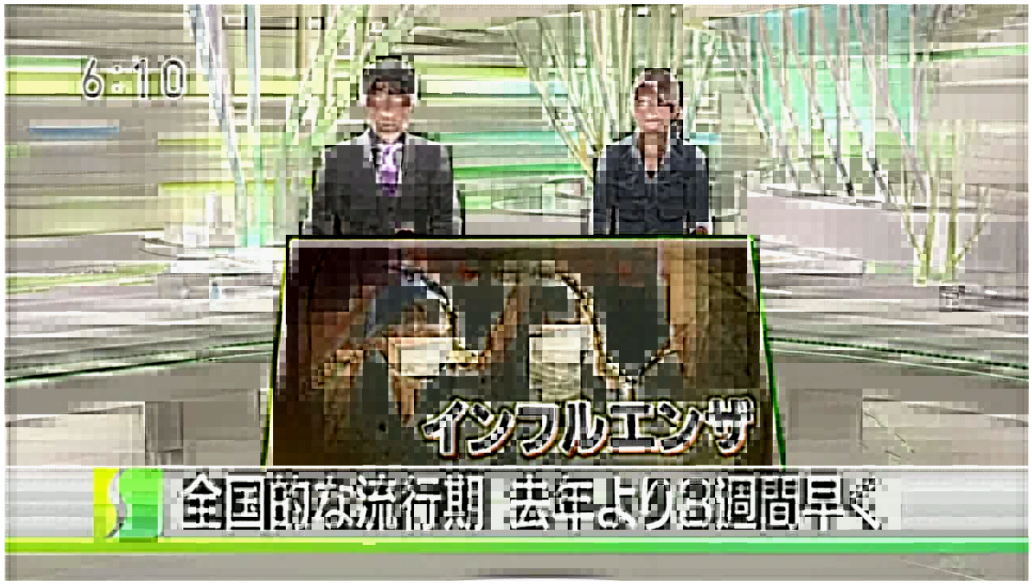
Table 5.6 Results of Experiment 4

No.	video								
	pv			mov			news		
	weak	med.	strong	weak	med.	strong	weak	med.	strong
1	1.4	1.2	1.0	1.5	1.2	1.1	1.2	1.1	1.0
2	1.5	1.0	1.0	2.0	1.0	1.0	1.5	1.0	1.0
3	3.0	1.5	1.3	4.0	2.0	1.5	4.0	2.0	2.5
4	2.0	1.5	1.2	4.0	2.5	1.5	3.0	2.0	2.5
5	2.0	1.0	1.0	3.0	1.0	1.0	2.0	1.0	1.0
6	2.0	1.0	1.0	3.0	2.0	2.0	2.0	1.0	1.0
7	3.5	2.5	1.5	4.5	2.0	1.5	2.5	1.5	1.5
8	2.0	1.5	1.0	3.0	1.8	1.0	1.6	1.2	1.0
9	2.0	1.0	1.0	2.0	1.0	1.0	1.0	1.0	1.0
10	1.5	1.2	1.0	3.0	2.5	1.5	2.0	1.0	1.0
11	4.0	2.0	2.0	4.0	2.0	1.0	4.0	1.0	1.0
12	3.5	2.0	1.5	3.4	3.0	2.5	3.6	2.0	1.5
13	2.0	1.0	1.2	3.0	2.0	2.4	2.0	1.0	1.7
14	3.0	1.0	0.5	4.0	2.0	1.2	3.0	1.0	1.0
15	1.0	1.0	1.0	2.0	1.0	1.0	1.0	1.0	1.0
16	2.0	1.0	1.2	3.0	1.0	2.0	2.0	1.0	1.5
17	1.4	1.2	1.0	3.0	2.0	1.0	3.0	2.0	1.0
18	2.0	1.5	1.0	2.5	1.0	1.5	2.0	1.0	1.0
19	2.0	1.0	1.5	4.0	2.0	1.0	3.0	1.0	2.0
20	3.0	2.0	2.0	3.0	2.0	1.0	4.0	2.0	2.0

Table 5.7 Results of Experiment 5

No.	video								
	pv			mov			news		
	weak	med.	strong	weak	med.	strong	weak	med.	strong
1	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
2	2.0	1.0	1.0	2.0	1.0	1.0	2.0	1.0	1.0
3	2.0	1.5	1.5	4.0	2.5	2.3	5.0	2.0	3.0
4	2.0	1.5	1.0	3.5	2.5	2.0	4.0	1.2	3.1
5	2.0	1.0	1.0	3.0	1.0	1.0	2.0	1.0	1.0
6	3.0	2.0	2.0	4.0	3.0	2.0	3.0	1.0	2.0
7	2.8	1.5	1.3	4.5	2.5	1.5	3.0	1.8	2.0
8	2.0	2.0	1.0	3.5	1.5	1.0	4.0	1.5	1.0
9	2.0	1.0	1.0	4.0	3.0	2.0	3.0	1.0	1.0
10	2.5	1.3	1.0	4.0	3.0	2.0	2.5	1.5	2.0
11	4.0	2.0	2.0	4.0	3.0	2.0	4.0	1.0	4.0
12	3.0	1.5	2.0	4.0	3.4	2.8	4.2	2.5	2.5
13	2.0	1.0	1.0	4.0	3.0	3.0	2.0	1.0	1.7
14	4.0	1.0	0.5	4.0	2.0	1.2	2.0	1.0	1.3
15	2.0	1.0	1.0	4.0	1.0	1.0	3.0	1.0	1.0
16	2.0	1.0	1.0	3.0	1.0	2.0	2.0	1.0	1.5
17	4.0	3.0	2.0	4.0	3.0	2.0	5.0	3.0	1.0
18	2.0	1.0	1.5	4.0	1.0	2.0	3.0	2.0	2.0
19	2.0	1.0	1.0	5.0	2.0	1.0	3.0	1.0	2.0
20	4.0	3.0	3.0	4.0	4.0	3.0	5.0	3.0	4.0

(a) medium



(b) strong

Fig. 5.6 Comparison for characters in medium vs. strong

a significance test for each experimental video and found that the subjective evaluation experiment achieved a certain degree of control over a wide range of degrees of degradation.

In the experimental video news, the objective experimental results showed that news_strong had lower PSNR and SSIM values than news_medium, indicating that it was more degraded. However, the subjective experiment showed that news_medium was more degraded than news_strong and had no commercial value. The reason for the contradiction between the results of the objective experiment and the subjective experiment is considered to be that news is a video mainly composed of text, and as shown in Fig. 5.6, the strong parameter is not suitable for videos mainly composed of text, and in that case, the text is more visible than the medium parameter.

Also, in pv, more than 90% of the subjects evaluated the medium and strong degradation as “no commercial value” or “somewhat no commercial value”, and 0% of the subjects evaluated them as “commercial value” or “somewhat commercial value” .

In mov, 85% of the subjects evaluated the strong degradation as “no commercial value” or “somewhat no commercial value”, and 0% of the subjects evaluated it as “commercial value” or “somewhat commercial value”.

In news, more than 80% of the subjects evaluated the medium and strong degradation as “no commercial value” or “somewhat no commercial value”, and 0% of the subjects evaluated it as “commercial value” or “somewhat commercial value”.

From these results, it can be said that samples with no commercial value were generated at the medium and strong processing strengths.

The sample videos of each processing strength and the original videos generated are

shown in pv: Fig. 5.7, mov: Fig. 5.8, news: Fig. 5.9.

5.3.2 Results and discussion for experiments on quantatively adjustable degradation degree

In Experiment 6, as a result of the experiment measuring the difference between the reconstructed video and audio and the original video and audio based on MSE values, the MSE values for both audio and video were all 0 in the 3 experimental videos.

In Experiment 7, the results of the experiment comparing the TPSNR and PSNR values of the sample videos are shown in Table 5.8.

In Experiment 6, it was found that the proposed quantitative degradation adjustment method in this study is perfectly reversible for both video and audio since all MSE measurement results were 0.

In Experiment 7, it was proven that the proposed quantitative degradation adjustment method in this study is effective within a certain TPSNR setting range, as the TPSNR and PSNR values of the other experimental videos are close, while the TPSNR

Table 5.8 Results of Experiment 7

video	TPSNR(dB)	PSNR(dB)
pv_30	30	29.9
pv_25	25	25.2
pv_20	20	21.8
mov_30	30	30.0
mov_25	25	25.3
mov_20	20	22.5
news_30	30	29.0
news_25	25	24.7
news_20	20	19.9

(a) original **(b)** weak

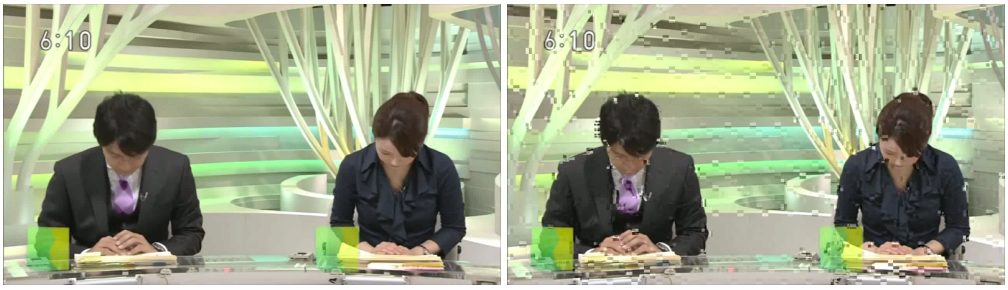
(c) medium **(d)** strong

Fig. 5.7 Comparison of degradation degree of pv

(a) original **(b)** weak

(c) medium **(d)** strong

Fig. 5.8 Comparison of degradation degree of mov



(a) original **(b)** weak

(c) medium **(d)** strong

Fig. 5.9 Comparison of degradation degree of news

and PSNR values of pv_20, mov_20, and news_30 are different.

On the other hand, it was also found that the proposed quantitative degradation adjustment method in this study has an upper and lower limit for the TPSNR value setting range. The reason for this is that the qDCTC coefficients of videos have characteristics depending on the video genre, etc., and it is not possible to correspond to all genres of videos with the same TPSNR value setting range.

Sample videos and original videos generated for each TPSNR value are shown in the following figures: pv in Fig. 5.10, mov in Fig. 5.11 news in Fig. 5.12.

(a) original **(b)** TPSNR=30dB

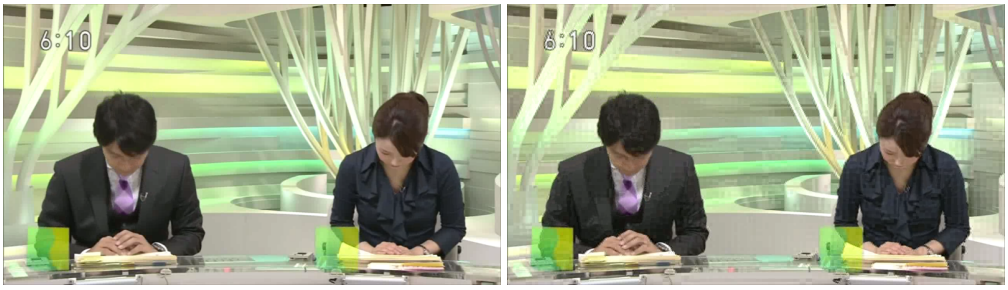
(c) TPSNR=25dB **(d)** TPSNR=20dB

Fig. 5.10 Comparison of degradation degree of pv

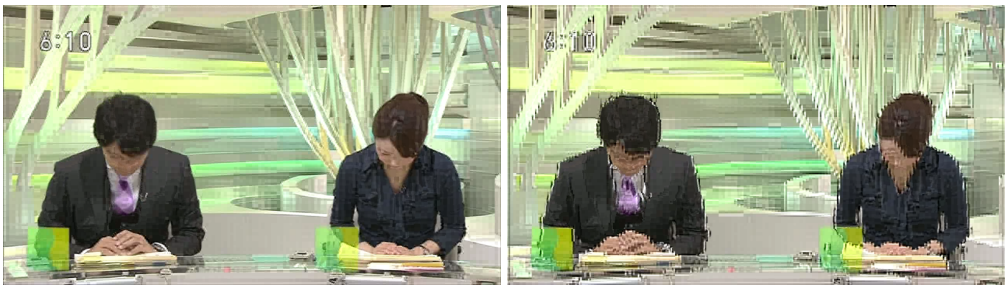
(a) original **(b)** TPSNR=30dB

(c) TPSNR=25dB **(d)** TPSNR=20dB

Fig. 5.11 Comparison of degradation degree of mov



(a) original **(b)** TPSNR=30dB



(c) TPSNR=25dB **(d)** TPSNR=20dB

Fig. 5.12 Comparison of degradation degree of news

6

Proposed method 2: Data hiding technique enabling data size reduction

6.1 Overview

In the video coding standards MPEG-2, H.264/AVC, and HEVC, frames (pictures) are categorized into I (Intra) pictures, P (Predictive) pictures, and B (Bidirectionally pre-

dictive) pictures. Each frame is segmented into blocks, which are broadly classified as intra-blocks and inter-blocks of luminance and chrominance components. Orthogonal transformations and quantization are subsequently applied within each block, yielding qTCs. As discussed in Section 4.1, since inter-blocks are fundamentally derived from intra-blocks, any impairment in the intra-blocks will propagate to inter-blocks and accumulate. Concurrently, the human visual system exhibits heightened sensitivity to DC coefficients in comparison to AC coefficients. Consequently, it is conceivable that the AC components of qTCs in intra-blocks constitute suitable targets for the data hiding process. The flowcharts illustrating the proposed processes are depicted in Figs. 6.1 and 6.2.

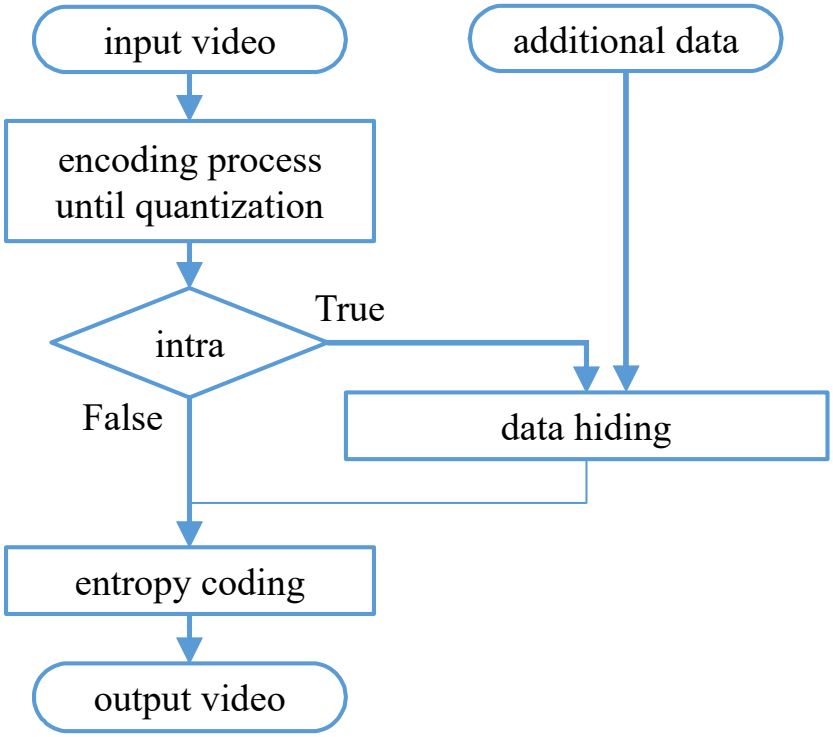


Fig. 6.1 Flowchart for embedding

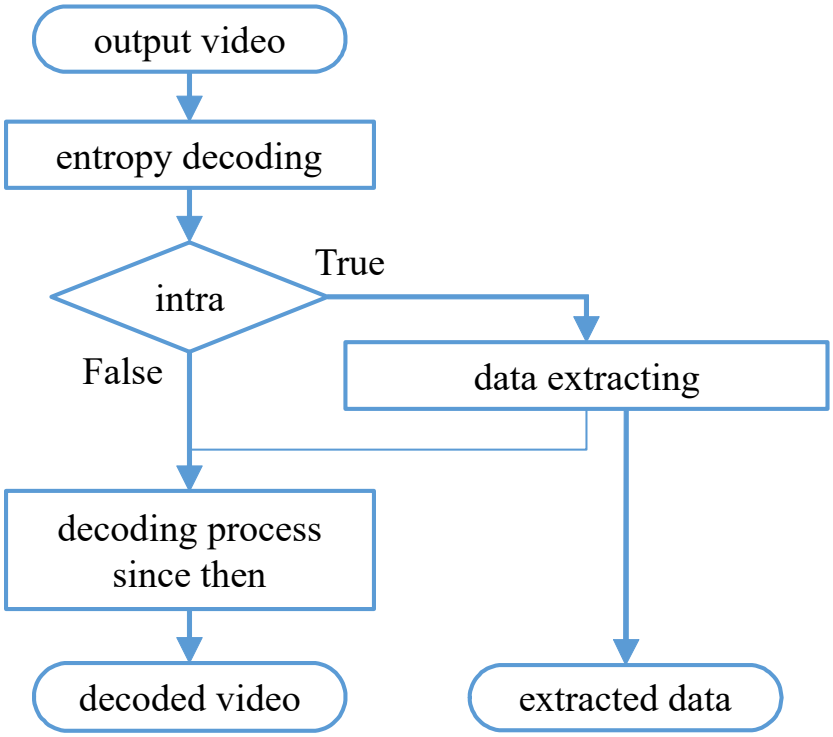


Fig. 6.2 Flowchart for extracting

6.2 Modulo Addition Technique

For a given integer data element, let its value be denoted as V. The process of embedding one bit of supplementary data with a value of A will modify V to V_e as follows:

$$V_e = \lfloor \frac{V}{2} \rfloor \times 2 + A\,, \tag{6.1}$$

where $\lfloor \rfloor$ indicates the rounding down operation.

The embedded data bit A could be extracted as follows:

$$A = V_e \bmod 2\,, \tag{6.2}$$

where mod indicates the modulo operation.

In essence, the modulo addition technique can be regarded as a variant of the classical LSB replacement technique, with the two techniques being potentially identical, contingent upon the bit-plane representation method employed for negative numbers. As elucidated in Chapter 2, in the MPEG-2, H.264/AVC, and HEVC formats, akin to other image and video formats utilizing orthogonal transformations such as JPEG, transforming an AC coefficient with a value of zero into a non-zero quantity will directly augment the data size. Evidently, both the classical LSB replacement technique and the modulo addition technique will inevitably escalate the number of non-zero AC coefficients. Furthermore, according to Eq. (6.1), if V is positive and an even multiple, the absolute magnitude of V_e will surpass that of V when $A = 1$. These characteristics pose a risk of substantial data size inflation, which is undesirable in most data hiding scenarios, such as when the remaining capacity of the video disc is insufficient. Henceforth, this technique will be referenced as "MA".

6.3 Modified Modulo Addition Technique

As a modified variant of the modulo addition technique, an alternative approach has been previously proposed [50]. In this approach, to circumvent the risk of augmenting the absolute value of an AC coefficient, the embedding algorithm was modified as

follows:

$$V_e = int(\frac{V - A}{2}) \times 2 + A\,, \tag{6.3}$$

where $int(x)$ indicates the integer part of x and can be defined as:

$$int(x) = \begin{cases} \lfloor x \rfloor & (x > 0) \\ \lceil x \rceil & (otherwise)\,, \end{cases} \tag{6.4}$$

where $\lceil\rceil$ indicates the rounding up operation.

The embedded data bit A could be extracted with Eq. (6.2) as well.

This technique effectively mitigates the risk of escalating the absolute value of non-zero AC coefficients. However, it alters the value of certain AC coefficients from zero to one. Consequently, the data size of the video may experience an increase following the data hiding process. Henceforth, this technique will be referred to as "'mMA'.

6.4 Other Related Techniques

A number of other data hiding techniques tailored for compressed images and videos have also been proposed [51–54]. Mobasseri et al. introduced a data hiding technique for MPEG-2 videos utilizing VLC mapping [51], which does not lead to an increase in the data size of the videos. However, it is solely compatible with CAVLC and necessitates a prior detection process. Furthermore, the maximum payload capacity is

contingent upon the cover video. Chaumont et al. proposed a DCT-based data hiding technique that disseminates the payload data across a group of DCT coefficients [52]. This technique can realize an enhanced image quality, but it also results in an increase in data size. Hartung et al. demonstrated the capability to embed additional data into MPEG-2 videos without data size augmentation [53], yet it carries the risk of losing the embedded data because it embeds the additional data into the inverse quantized DCT coefficients and quantizes them again (the quantization process being lossy). While these techniques are effective in their respective scenarios, each of them exhibits evident deficiencies in other scenarios, particularly in cases of data hiding into capacity-limited multimedia storage media. F4, the foundation of F5 [54], is a technique that embeds data into non-zero AC coefficients of cover data while simultaneously reducing the cover data size. However, this technique inevitably transforms certain 1 and -1 AC coefficients into zeros, thus necessitating a unique key for each cover data to record which zeros remain unchanged, which are modified from 1, and which are modified from -1. To facilitate data extraction from each cover data, this key must be merged with any other keys, and the key-sharing process is costly in most data hiding scenarios. F5 is an improved version of F4 and also exhibits this characteristic.

The optimal positions for embedding supplementary data within a block have been the subject of prior investigations [55]. The qTCs of an 8×8 block can be partitioned into three distinct frequency regions: F_L (low frequencies), F_M (medium frequencies), and F_H (high frequencies), as illustrated in Fig. 6.3. The frequency band deemed suitable for embedding purposes is F_M [55]. However, the mMA technique primarily performs data hiding within F_L. To refine this technique, AC coefficients residing in F_M

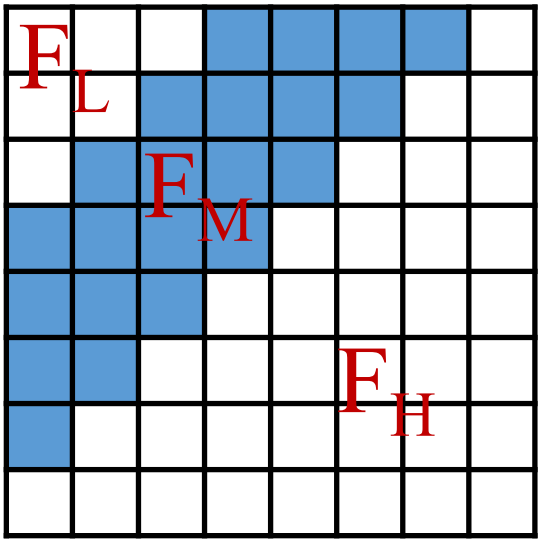


Fig. 6.3 Various frequency regions in a 8×8 block.

should be selected for embedding when the payload is relatively modest.

6.5 Details of the proposed data hiding technique

In this subsection, the proposed polarized LSB replacement technique will be explained and discussed.

The preceding discussion has elucidated that the video data size may experience an escalation subsequent to the data hiding process in MA and mMA, and the key-sharing procedure is costly in F5. In this context, I propose a novel technique, termed polarized LSB replacement, to circumvent data size augmentation and key-sharing requirements. Henceforth, this technique will be referred to as "PLSB-r". Before delving into the algorithm's explication, a partial lookup table encompassing several original qTCs and their corresponding modified counterparts for MA, mMA, and PLSB-r is presented in

Table 6.1 Modified qTC lookup table

Additional data	0				1			
Original qTC	MA	mMA	F5	PLSB-r	MA	mMA	F5	PLSB-r
4	4	4	4	4	5	3	3	3
3	2	2	2	2	3	3	3	3
2	2	2	2	2	3	1	1	1
1	0	0	0	-1	1	1	1	1
0	0	0	–	–	1	1	–	–
-1	-2	0	-1	-1	-1	-1	0	1
-2	-2	-2	-1	-1	-1	-1	-2	-2
-3	-4	-2	-3	-3	-3	-3	-2	-2
-4	-4	-4	-3	-3	-3	-3	-4	-4

Table 6.1.

The modifications to the qTC values are depicted in Fig. 6.4, where the red arrows indicate changes that augment the absolute value of qTCs, green arrows signify changes that diminish the absolute value of qTCs, and white arrows represent no alteration in the absolute value. From Table 6.1 and Fig. 6.4, it can be deduced that in contrast to MA and mMA, the proposed technique never escalates the absolute value of any qTC. Furthermore, as long as any qTC with a value greater than 1 or less than -1 is present (a condition essentially satisfied in all natural images and videos with sufficient quality), the video data size decreases due to the reduction in the absolute value of those qTCs. The algorithm of the proposed PLSB-r technique is elucidated below. First, only in cases where $V \neq 0$, V_e is acquired as follows:

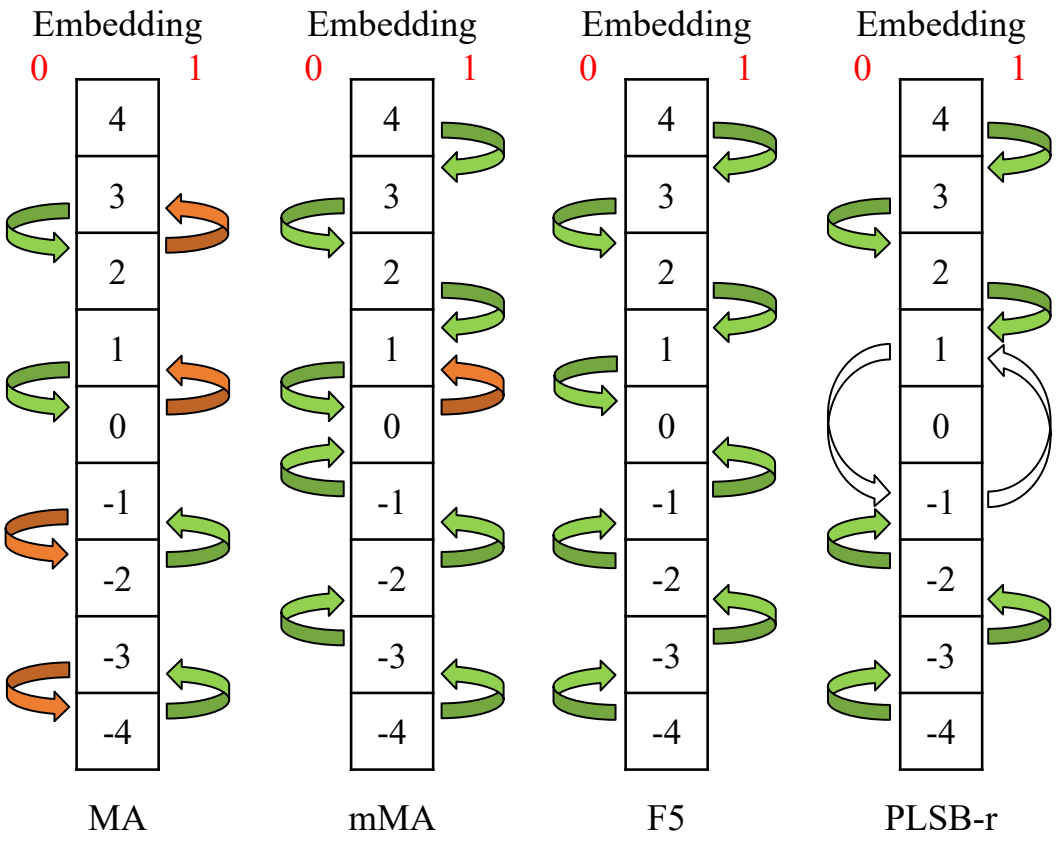


Fig. 6.4 Changes of qTC values in MA, mMA, F5 and PLSB-r

$$V_e = \begin{cases} int(\frac{V-A}{2}) \times 2 + A & (V > 0) \\ int(\frac{V-A+1}{2}) \times 2 + A & (V < 0)\,. \end{cases} \tag{6.5}$$

Next, if $V_e \leq 0$, it will be reduced by 1 as follows:

$$V_e{}' = \begin{cases} V_e & (V_e > 0) \\ V_e - 1 & (V_e \leq 0)\,. \end{cases} \tag{6.6}$$

The value V of one qTC is changed to $V_e{}'$ finally. Note that $V_e{}' \neq 0$, the additional data bit A could be extracted as follows:

$$A = \begin{cases} (V_e{}') \bmod 2 & (V_e{}' > 0) \\ (V_e{}' + 1) \bmod 2 & (V_e{}' < 0)\,. \end{cases} \tag{6.7}$$

In other words, akin to F5, 1 is extracted if the modified qTC exhibits a positive odd value or a negative even value; 0 is extracted if the modified qTC exhibits a positive even value or a negative odd value. Through this approach, the original video is not mandated for the extraction process. Diverging from F5, zero AC coefficients are disregarded, thereby preserving the quantity of zero AC coefficients, eliminating the requirement for a unique key for each cover data during data extraction. In the proposed technique, the sign of a qTC with a value of 1 or -1 may undergo inversion, i.e., -1 be-

comes 1, and 1 becomes -1. Additionally, the parity of the positive modified qTCs and negative modified qTCs conflict when the same data bit is embedded. The histogram for AC coefficients of the cover data remains symmetrical and nearly preserves a Laplacian distribution, thereby rendering it arduous for attackers to detect whether the cover data harbors concealed data. These characteristics differentiate the proposed technique entirely from conventional LSB replacement and modulo addition techniques, wherein the parity of the positive modified qTCs and negative modified qTCs is generally congruent. Owing to this distinctive feature of the proposed technique, although its robustness is relatively diminished, it proves more effective in steganography situations, where undetectability is of paramount importance, or enrichment scenarios, wherein robustness can be disregarded to a certain extent. The time-complexity of both the data hiding process and the data extraction process is $O(n)$. The processing time of the proposed technique is generally negligible, amounting to less than 1/100 of the video coding or decoding time in my experiments. Thus, the proposed technique is well-suited for real-time situations where information timeliness is critical. The scheme of the proposed technique is straightforward, facilitating its combination with other data hiding techniques that enhance security and other aspects. As a limitation of the proposed technique, the maximum payload will be inferior to MA and mMA because the embedding process is solely performed on non-zero qTCs, and the quality will be slightly worse than F5 due to the greater change in the absolute value of 1 and -1 AC coefficients. As illustrated in Table 6.1, when $n = 1$, the absolute value of an AC coefficient will never escalate after the embedding process is performed.

7

Experiments and discussion of method 2

7.1 Overview

To corroborate the efficacy of the proposed technique, I conducted several experiments with the MPEG-2 format.

In the experiments, the 15 videos employed were uncompressed raw videos in YUV format obtained from `https://media.xiph.org/video/derf/`. The starting frames

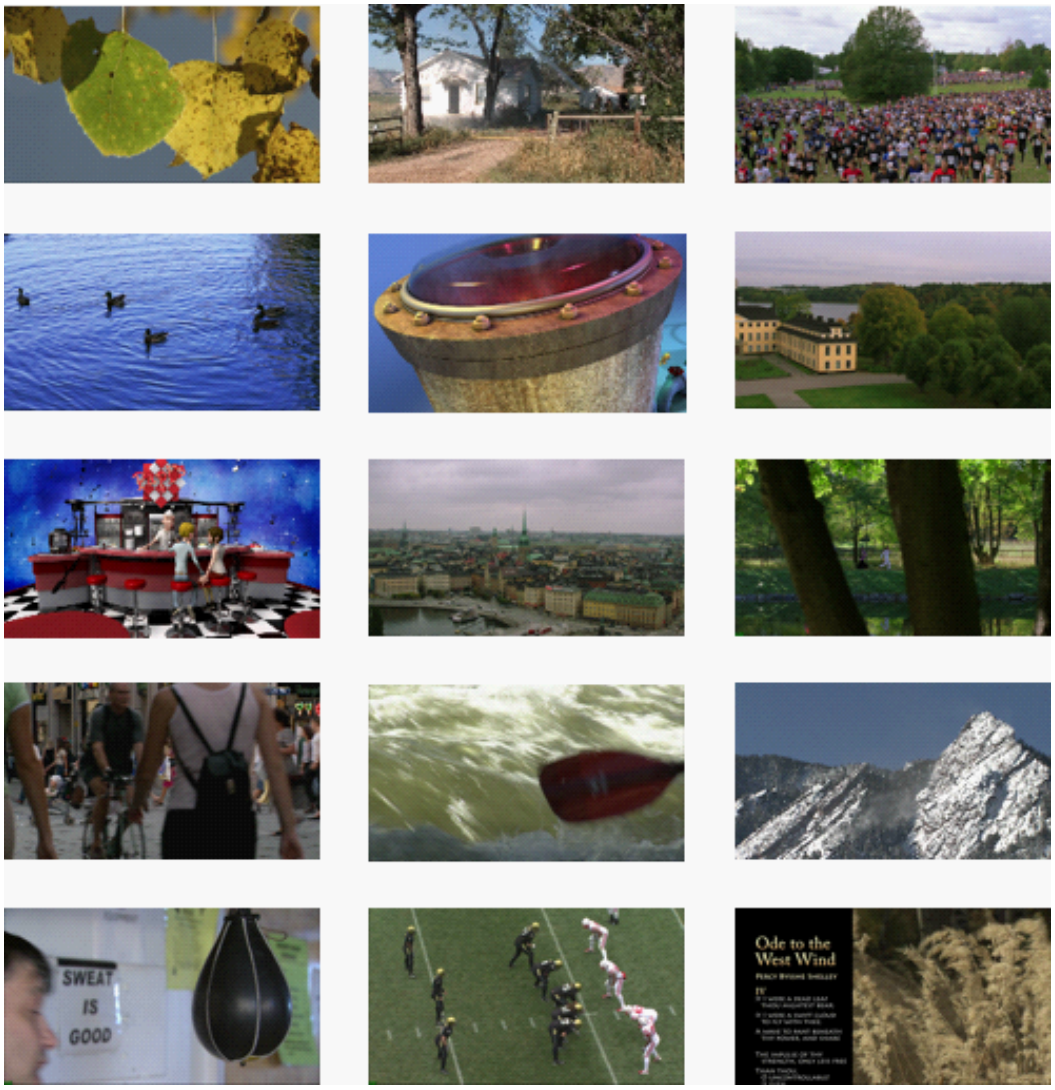


Fig. 7.1 Starting frames of the experimental videos. Horizontally from top left: a=aspen, co=controlled burn, cr=crowd run, d=ducks take off, f=factory, i=in to tree, l=life, o=old town cross, pa=park joy, pe=pedestrian area, r=red kayak, sn=snow mnt, sp=speed bag, t=touchdown pass, w=west wind easy.

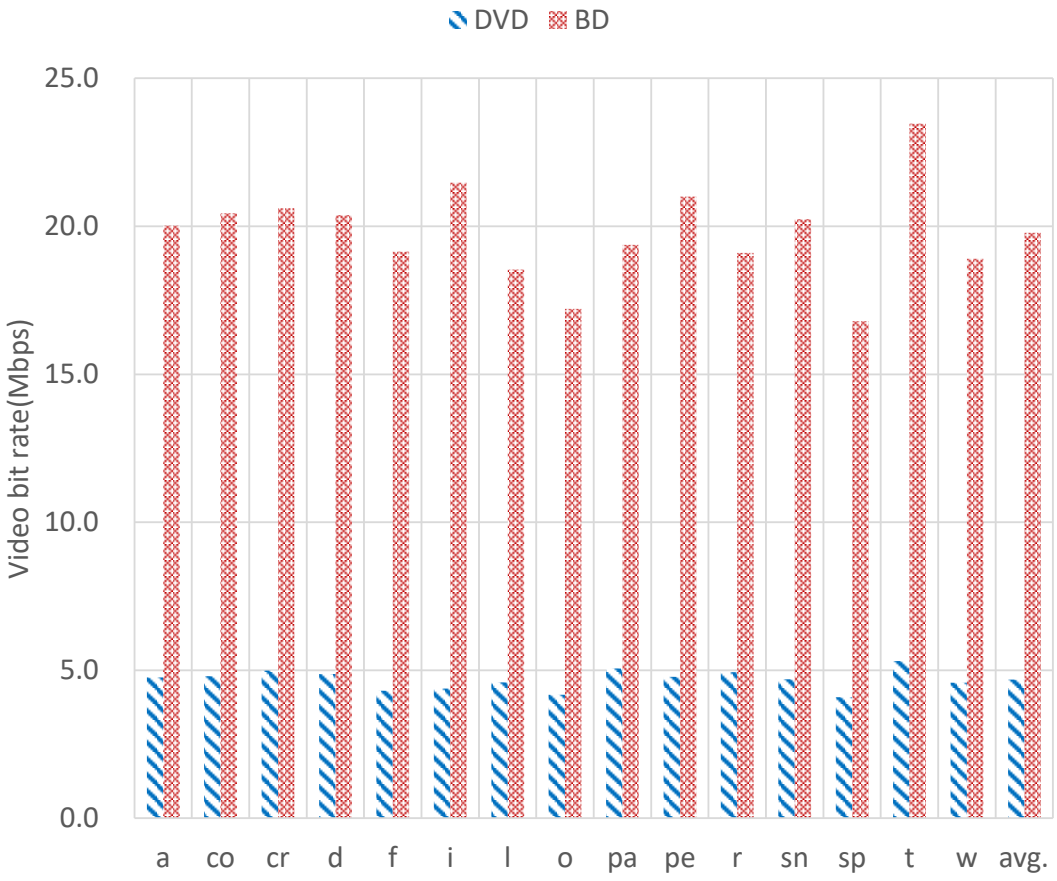


Fig. 7.2 Bit rates of the experimental videos. (Mbps)

Table 7.1 Details of the experimental videos

Target	DVD	BD
Codec	MPEG-2	
Resolution	720×480	1920×1080
Aspect ratio	16:9	
Frame rate	29.97fps	
Number of frames	300	
Number of frames in one GOP	18	12
Scan type	Progressive	
Color space	YUV	
Chroma subsampling	4:2:0	
Average bit rate	4.69Mbps	19.78Mbps

Table 7.2 QP of the experimental videos

target \ video	a	co	cr	d	f	i	l	o	pa	pe	r	sn	sp	t	w
DVD	6	11	12	13	2	5	3	5	12	13	8	8	1	3	9
BD	5	10	12	16	2	6	2	6	16	14	9	8	3	4	7

of the experimental videos are depicted in Fig. 7.1. These 15 raw videos were compressed into common DVD and BD formats, which are also generally utilized in online video services, with the specifications delineated in Table 7.1. The compressed videos devoid of embedded data will be referred to as baseline videos henceforth. To facilitate an objective comparison between the videos with embedded data and the baseline videos, fixed Quantization Parameter (QP) values were employed to maintain consistency in the coding process for the comparison objects. The QP values varied across different videos to maintain similar bit rates, as shown in Table 7.2. The bit rates of the baseline videos and their average are illustrated in Fig. 7.2. The additional data to be

embedded consisted of a randomly generated bit sequence S, which remained consistent across all experiments.

To compare the proposed technique with conventional ones, I performed MA, mMA, and PLSB-r on the experimental videos. Two conventional techniques utilizing qTCs were also implemented for the purpose of comparison [23, 24], henceforth referred to as Adap and Psy, respectively. With the objective of comparing techniques with similar payload values, in MA and mMA, four bits of additional data were embedded in the F_M of one block, while in PLSB-r, all non-zero quantized AC coefficients in intra-blocks were utilized for embedding. In Adap, the payload was adjusted to be similar to the proposed technique. However, Psy could not achieve a comparable payload to the proposed technique, so the payload in Psy was maximized.

7.2 Losslessness Verification Experiment

To validate the lossless nature of the proposed technique, I computed the MSE value between the extracted data and the original bit sequence S. The resultant MSE values were all 0, providing corroboration that the data hiding processes of the three techniques are indeed lossless.

7.3 Effectiveness Verification Experiment

In order to validate the effectiveness of the proposed techniques, I conducted an assessment of the cover videos based on criteria such as quality, payload, and file-size overhead. Payload (P) refers to the volume of concealed data within the video. File-size overhead (O) denotes the increase in data size resulting from data embedding, computed

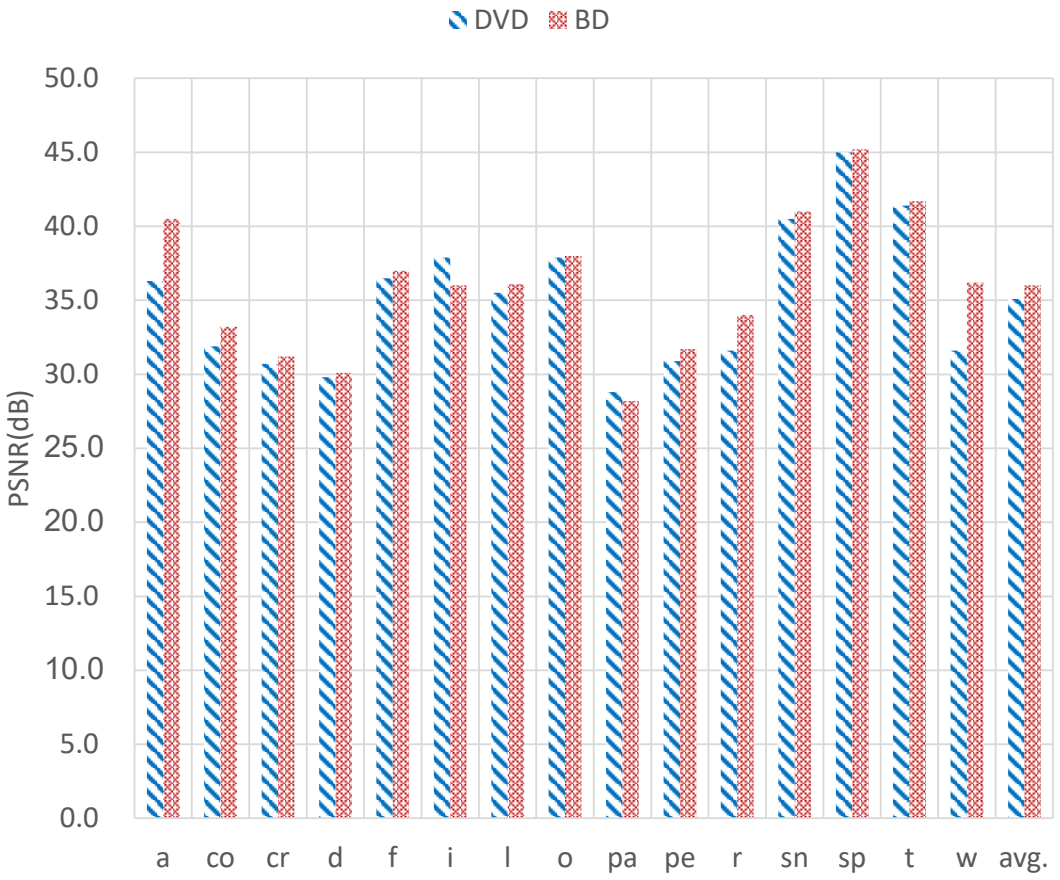


Fig. 7.3 PSNR results of the experimental videos. (dB)

as $O = F_2 - F_1$, where F_2 represents the data size post-embedding and F_1 denotes the data size pre-embedding.

Quality assessment entails the utilization of metrics such as PSNR and SSIM. These metrics are computed by comparing the enriched videos against the baseline videos. The PSNR and SSIM values for each video pair are determined by averaging across all frames.

The payload and file-size overhead outcomes are expressed as payload per second

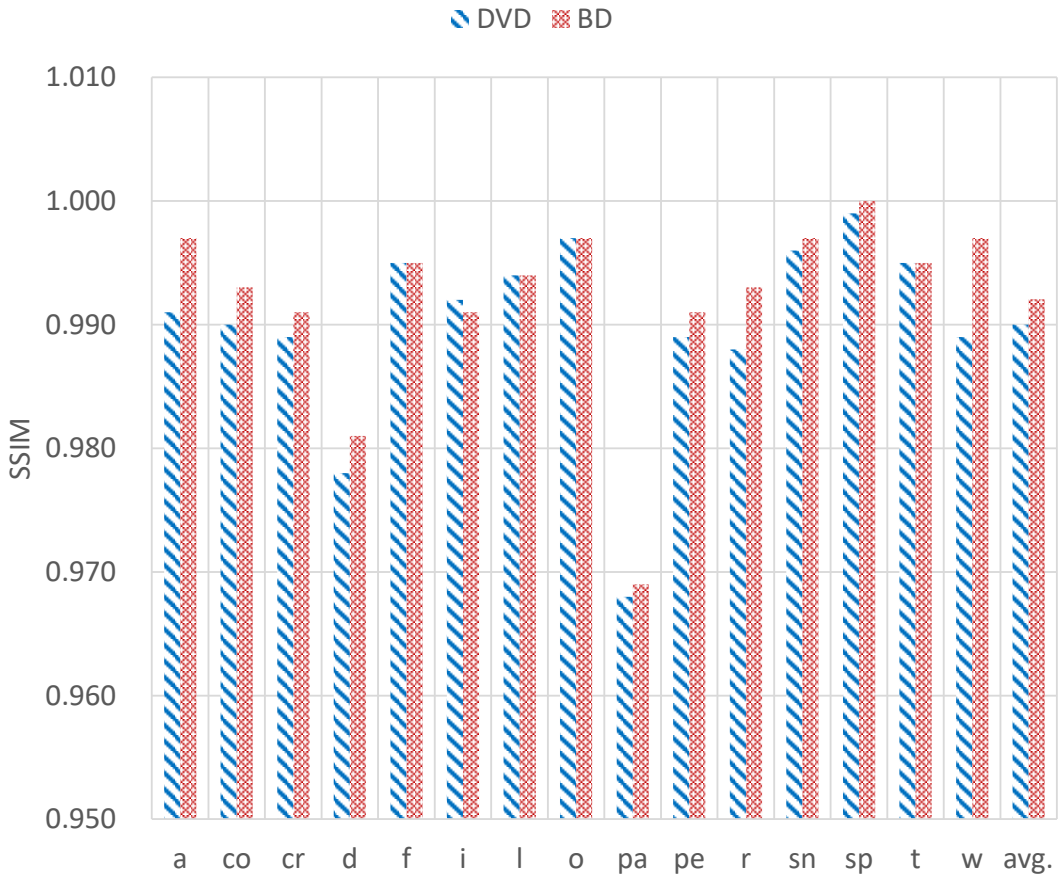


Fig. 7.4 SSIM results of the experimental videos.

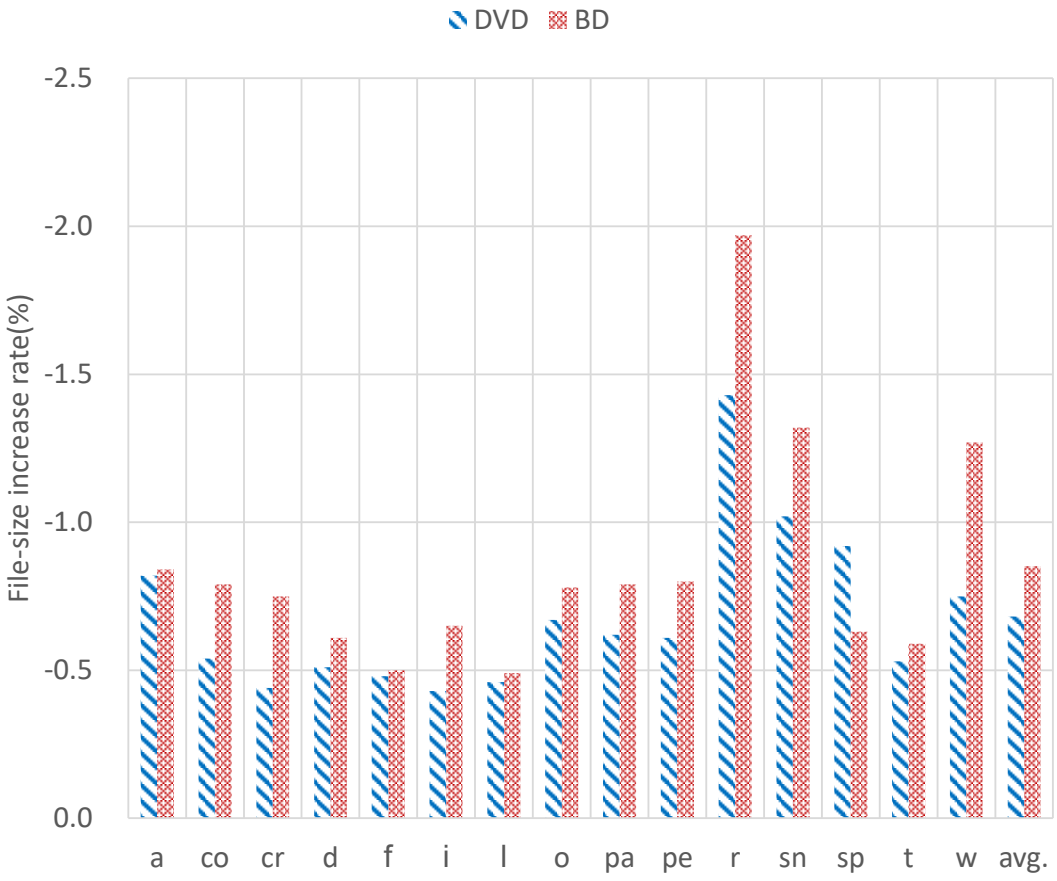


Fig. 7.5 File-size increase rate results of the experimental videos. (%)

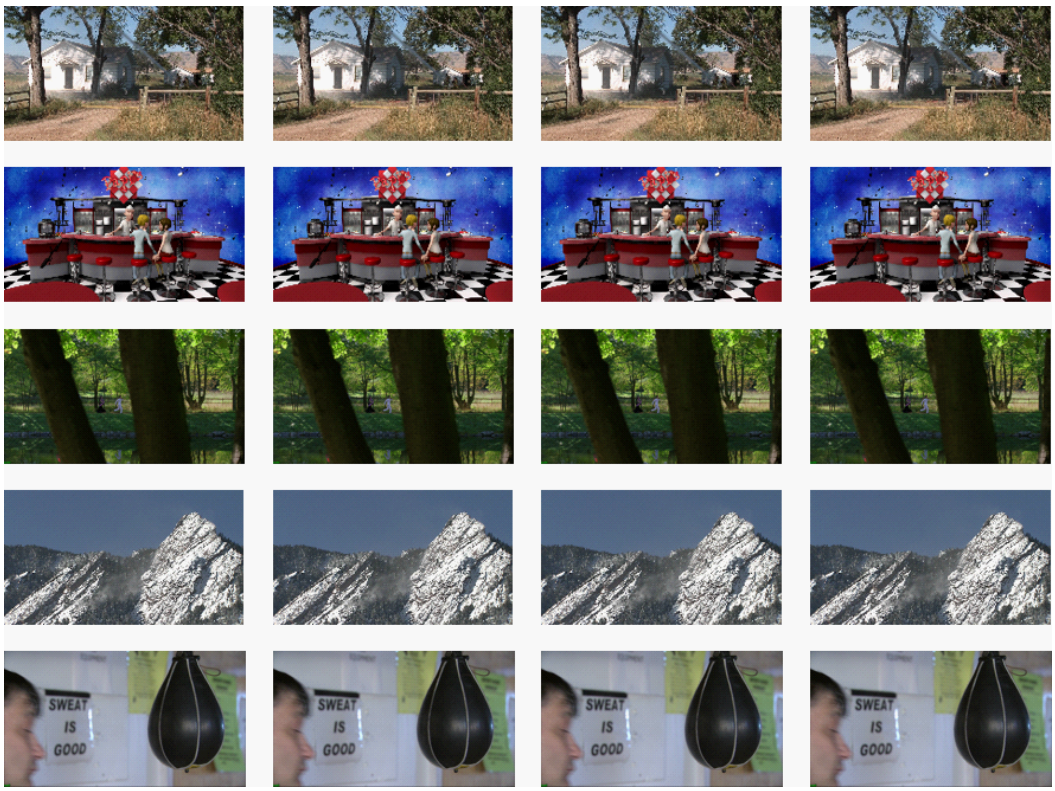


Fig. 7.6 Baseline video and data hiding results for “controlled burn”, “life”, “park joy”, “snow mnt” and “speed bag”: (a) baseline, (b) Adap, (c) Psy, (d) PLSB-r, from top to bottom.

Table 7.3 Average results of the comparison between conventional and proposed techniques

(a) DVD videos

technique	PSNR [dB]	SSIM	payload per second [Kbps]	overhead per second [Kbps]	ratio	FR [%]	PR [%]
MA	44.0	0.999	68.6	51.1	0.75	1.05	1.43
mMA	43.4	0.999	68.6	37.7	0.55	0.77	1.43
Adap	35.6	0.989	69.4	30.5	0.44	0.63	1.45
Psy	46.8	0.999	20.9	7.4	0.35	0.15	0.43
PLSB-r	35.1	0.990	66.6	-33.3	-0.50	-0.68	1.39

(b) BD videos

technique	PSNR [dB]	SSIM	payload per second [Kbps]	overhead per second [Kbps]	ratio	FR [%]	PR [%]
MA	43.7	0.999	676.1	580.3	0.86	2.92	3.34
mMA	43.6	0.999	676.1	497.0	0.74	2.49	3.34
Adap	37.0	0.992	432.8	285.0	0.66	1.42	2.14
Psy	45.8	0.999	120.6	90.4	0.75	0.43	0.60
PLSB-r	36.0	0.992	446.1	-170.3	-0.38	-0.85	2.20

and overhead per second, aiming to offer users of the proposed system a practical understanding.

The ratios of overhead to payload ($\mathrm{ratio} = O/P$) serve as an index for evaluating payload efficiency in data hiding from the perspective of overhead. File-size increase rates ($\mathrm{FR} = O/F$) and payload rates ($\mathrm{PR} = P/F$) are calculated, with F representing the baseline video's file size. Negative file-size increase rate values signify a reduction in the cover video's size following the data hiding process.

Average results from experiments on all 15 experimental videos, both for DVD and BD, are depicted in Figs. 7.3, 7.4 and 7.5. Additionally, a comparison of the proposed technique with MA, mMA, Adap, and Psy is presented in Table 7.3. For example, Fig.

7.6 showcases the output videos of Adap, Psy, and PLSB-r for “controlled burn”, “life”, “park joy”, “snow mnt” and “speed bag” post data hiding.

7.4 Discussion

Analysis of Figs. 7.3, 7.4 and 7.6 reveals that the quality of output videos containing hidden data is satisfactory. Moreover, Fig. 7.5 indicates a decrease in data sizes for all videos post data hiding.

However, as illustrated in Fig. 7.7, it’s noteworthy that Psy exhibits significantly smaller payload, suggesting it’s unsuitable for direct comparison. Relative to the other three techniques with similar payloads (e.g., for DVD videos, 68.6 Kbps for MA and mMA, 69.4 Kbps for Adap, and 66.6 Kbps for PLSB-r), the proposed technique achieves data size reduction at the expense of some degradation in video quality. Nonetheless, with PSNR values hovering around 35dB and SSIM values around 0.99, the video quality remains deemed sufficient. Moreover, Fig. 7.6 illustrates that the degradation in video quality is visually imperceptible.

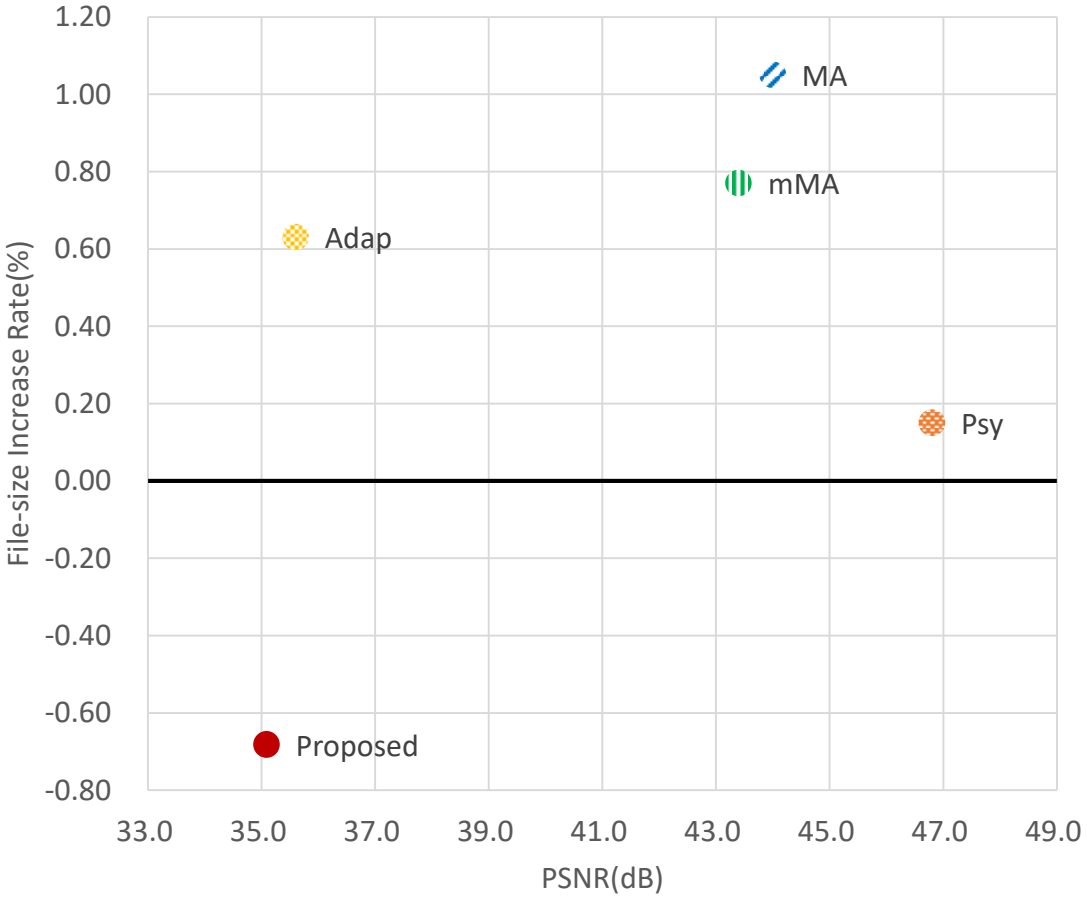


Fig. 7.7 PSNR vs. file-size increase rate.

8

Conclusion

In this study, two novel data hiding methods for video and audio data are proposed. The first method embeds audio data of video into the video data using a novel embedding technique, generate partial scrambling at the same time. The second method embeds data into a cover video using a different embedding technique. Both methods were evaluated using a variety of metrics, and it was shown that both methods can effectively

reduce the amount of data while controlling the quality of the video well.

As Method 1, I propose a partial scrambling method that enables reversible embedding between heterogeneous content. Since Discrete Cosine Transform is widely used in codecs commonly used in video streaming services, processing in the DCT coefficient domain can be applied to current video streaming services. In this study, for video with audio, when video data is encoded to MPEG4 format, audio data is embedded in the quantized DCT coefficients (AC and DC components) while generating partial scramble. The embedding process uses the parity of integers. AC coefficients are doubled and 1 bit of audio data is added. Partial scramble performs operations on DC and AC components. In DC component operation, MacroBlock-inner DC shuffle is performed. A shuffle table is prepared in advance, and the DC coefficients within the MacroBlock are exchanged according to the shuffle table. In AC component operation, sign inversion is performed to suppress the increase in file size. Experiments were conducted to measure the difference between the reconstructed video and audio and the original video and audio using MSE, and it was confirmed that both video and audio data could be reconstructed. In addition, experiments were conducted to measure the degree of degradation of the video of the sample video against the original video using PSNR and SSIM, and it was confirmed that the degree of degradation was appropriate. Finally, an experiment was conducted to measure the increase in file size of the sample video against the original video, and it was shown that the file size was reduced.

As Method 2, I propose a data hiding method that reduces the size of video data. From the video compression algorithm, it is known that an increase in the absolute value of the AC coefficient leads directly to an increase in the size of the video file.

Conventional data hiding methods that target AC coefficients cannot avoid increasing the absolute value of AC coefficients without a composite key. However, this study has realized a method that does not increase the absolute value of AC coefficients without a composite key by performing reverse operations on positive and negative AC coefficients based on the least significant bit replacement method, and further by avoiding zero coefficients. In addition, it has resilience to embedding detection by maintaining the Laplacian distribution of AC coefficients. This method can be applied as steganography, and multimedia enrichment. If combined with techniques that enhance robustness, it also has the potential to be applied as electronic watermarking. Experiments were conducted to measure the difference between the extracted data and the embedded data using MSE, and it was confirmed that the embedded data could be fully reconstructed. In addition, experiments were conducted to measure the degree of degradation of the video using PSNR and SSIM, and it was shown that the quality of the video could be maintained at a high level since the degree of degradation was 35 dB or more in PSNR and 0.98 or more in SSIM. Finally, an experiment was conducted to measure the increase in file size of the processed video against the original video, and it was shown that the file size was reduced in all cases.

The two proposed methods have different purposes, but both have made new contributions to data reduction in the field of data hiding.

Acknowledgments

I WOULD LIKE TO express my profound gratitude to my esteemed advisor, Prof. Yuji Sakamoto, as well as Prof. Seok Kang, for their unwavering support throughout my doctoral studies and related research endeavors. Their patience, motivation, and vast knowledge have been invaluable. Their guidance has been instrumental in navigating the challenges of research and the writing of this thesis. Furthermore, I extend my sincere appreciation to the remaining members of my advisory committee: Prof. Miki Haseyama, Prof. Yoshinori Dobashi, and Prof. Takahiro Ogawa. Not only have their insightful comments and encouragement been invaluable, but their probing questions have also inspired me to broaden my research perspectives. My heartfelt thanks also go to my fellow lab colleagues, who have provided me with invaluable assistance. Their camaraderie and support have made my doctoral journey more convenient and enjoyable. I am grateful to my fellow lab members for their aid in both academic and personal spheres. As an international student in Japan, I feel fortunate to have received

such generous support from others. Throughout the doctoral program, we collaborated on research, learning from one another in a mutually enriching experience. Last but certainly not least, I would like to express my deepest gratitude to my family: my parents, my parents-in-law, and my wife, whose unwavering emotional support has been a bedrock throughout the writing of this thesis and my life in general.

Research achievements

Journals

- Xuefei Li, Seok Kang and Yuji Sakamoto, “Lossless Data Hiding Technique Reducing Cover Data Size for Compressed Videos,” Journal of Electronic Imaging 33, no. 01, 2024.
- Yasunori Yamamoto, Xuefei Li, Shingo Shimaura, Seok Kang and Yuji Sakamoto, “Reversible Privacy Protection Technique for Surveillance Camera Images,” The journal of the Institute of Image Information and Television Engineers 71 (3), J110-J117, 2017.

International conferences

- Xuefei Li, Seok Kang and Yuji Sakamoto, “A Partial Scrambling Technique with Adjustable Degradation Degrees Embedding Different Types of Contents,” In 2016 International Workshop on Advanced Image Technology (IWAIT), 2C-5,

2016.

- Xuefei Li, Yasutoshi Miura, Seok Kang and Yuji Sakamoto, “A Scrambling Technique Embedding Soundtracks into Videos for Streaming Media,” In 2018 International Workshop on Advanced Image Technology (IWAIT), pp. 1-4, IEEE, 2018.

OTHERS

- IWAIT 2016 Best Paper Awards, In 2016 International Workshop on Advanced Image Technology (IWAIT), 2C-5, 2016.

References

[1] I. Cox, M. Miller, J. Bloom, J. Fridrich, and T. Kalker, *Digital Watermarking and Steganography*. USA CA San Francisco: Morgan Kaufmann Publishers Inc., 2008.

[2] Y. Tew and K. Wong, "An overview of information hiding in H.264/AVC compressed video," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 24, no. 2, pp. 305–319, 2014.

[3] Y. Yamamoto, X. Li, S. Shimaura, S. Kang, and Y. Sakamoto, "Reversible privacy protection technique for surveillance camera images," *The journal of the Institute of Image Information and Television Engineers*, vol. 71, no. 3, pp. J110–J117, 2017.

[4] P. W. Wong, "A watermark for image integrity and ownership verification," in *Image Processing, Image Quality, Image Capture Systems Conference*, 1998.

[5] J. Fridrich, M. Goljan, and R. Du, “Lossless data embedding—new paradigm in digital watermarking,” *EURASIP Journal on Advances in Signal Processing*, vol. 2002, no. 2, pp. 185–196, 2002.

[6] T. Kalker and F. Willems, “Capacity bounds and constructions for reversible data-hiding,” in *2002 14th International Conference on Digital Signal Processing Proceedings. DSP 2002 (Cat. No.02TH8628)*, 2002.

[7] K. M. Hosny, S. T. Kamal, and M. M. Darwish, “A color image encryption technique using block scrambling and chaos,” *Multimedia Tools and Applications*, vol. 81, no. 1, pp. 505–525, 2022.

[8] C.-L. Li, Y. Zhou, H.-M. Li, W. Feng, and J.-R. Du, “Image encryption scheme with bit-level scrambling and multiplication diffusion,” *Multimedia Tools and Applications*, vol. 80, pp. 18479–18501, 2021.

[9] N. Dolati, A. Beheshti, and H. Azadegan, “A selective encryption for H. 264/AVC videos based on scrambling,” *Multimedia Tools and Applications*, vol. 80, no. 2, pp. 2319–2338, 2021.

[10] X. Zhang, R. Li, J. Yu, Y. Xu, W. Li, and J. Zhang, “Editguard: Versatile image watermarking for tamper localization and copyright protection,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11964–11974, 2024.

[11] X. Luo, Y. Li, H. Chang, C. Liu, P. Milanfar, and F. Yang, “DVMark: a deep multiscale framework for video watermarking,” IEEE, 2023.

[12] P. Ayubi, M. Jafari Barani, M. Yousefi Valandar, B. Yosefnezhad Irani, and R. Sedagheh Maskan Sadigh, “A new chaotic complex map for robust video watermarking,” vol. 54, pp. 1237–1280, Springer, 2021.

[13] S.-P. Lu, R. Wang, T. Zhong, and P. L. Rosin, “Large-capacity image steganography based on invertible neural networks,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10816–10825, 2021.

[14] X. Liao, J. Yin, M. Chen, and Z. Qin, “Adaptive payload distribution in multiple images steganography based on image texture features,” *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 2, pp. 897–911, 2020.

[15] R. J. Mstafa, Y. M. Younis, H. I. Hussein, and M. Atto, “A new video steganography scheme based on Shi-Tomasi corner detector,” *IEEE Access*, vol. 8, pp. 161825–161837, 2020.

[16] T. Arnold and L. Tilton, “Enriching historic photography with structured data using image region segmentation,” in *Proceedings of the 1st International Workshop on Artificial Intelligence for Historical Image Enrichment and Access*, pp. 1–10, 2020.

[17] B. Purnama, M. Kartika, K. Robiatul, F. Indriani, M. Kubo, and K. Satou, “The enrichment of texture information to improve optical flow for silhouette image,” *International Journal of Advanced Computer Science and Applications*, vol. 12, no. 2, 2021.

[18] G. Xuan, Y. Q. Shi, Z. Ni, P. Chai, X. Cui, and X. Tong, "Reversible data hiding for JPEG images based on histogram pairs," in *Lecture Notes in Computer Science*, pp. 715–727, Springer Berlin Heidelberg, 2007.

[19] P. Zheng, B. Zhao, and M. Liu, "An efficient method to hide information in MPEG video sequences," in *2009 International Conference on Multimedia Information Networking and Security*, IEEE, 2009.

[20] Z. Shahid, M. Chaumont, and W. Puech, "Considering the reconstruction loop for data hiding of intra- and inter-frames of H.264/AVC," *Signal, Image and Video Processing*, vol. 7, no. 1, pp. 75–93, 2011.

[21] S. Gujjunoori and B. B. Amberker, "A DCT based near reversible data embedding scheme for MPEG-4 video," in *Lecture Notes in Electrical Engineering*, pp. 69–79, Springer India, 2013.

[22] F. Huang, X. Qu, H. J. Kim, and J. Huang, "Reversible data hiding in JPEG images," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 9, pp. 1610–1621, 2016.

[23] Y. Chen, H. Wang, H. Wu, Z. Wu, T. Li, and M. Asad, "Adaptive video data hiding through cost assignment and STCs," *IEEE Transactions on Dependable and Secure Computing*, p. 1–1, 2019.

[24] F. Muhammad and E. Ferda, "Video steganography based on DCT psychovisual and object motion," *Bulletin of Electrical Engineering and Informatics*, vol. 9, no. 3, p. 1015–1023, 2020.

[25] Z. Ni, Y. Shi, N. Ansari, and W. Su, "Reversible data hiding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 3, pp. 354–362, 2006.

[26] J. Tian, "Reversible watermarking by difference expansion," in *Proceedings of workshop on multimedia and security*, vol. 19, ACM, 2002.

[27] L. R. Mathews and V. A. C. Haran, "Histogram shifting based reversible data hiding using block division and pixel differences," in *2014 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT)*, IEEE, 2014.

[28] Z. Shahid and W. Puech, "A histogram shifting based RDH scheme for H.264/AVC with controllable drift," in *SPIE Proceedings* (A. M. Alattar, N. D. Memon, and C. D. Heitzenrater, eds.), SPIE, 2013.

[29] S. M. Kim, S. B. Kim, Y. Hong, and C. S. Won, "Data hiding on H.264/AVC compressed video," in *Lecture Notes in Computer Science*, pp. 698–707, Springer Berlin Heidelberg, 2007.

[30] Y. K. Lee and L. H. Chen, "High capacity image steganographic model," *IEE Proceedings - Vision, Image, and Signal Processing*, vol. 147, no. 3, p. 288, 2000.

[31] M. U. Celik, G. Sharma, A. M. Tekalp, and E. Saber, "Lossless generalized-LSB data embedding," *IEEE Transactions on Image Processing*, vol. 14, no. 2, pp. 253–266, 2005.

[32] C. Lalengmawia, A. Bhattacharya, and A. Datta, "Image steganography using advanced encryption standard for implantation of audio/video data," in *2016 In-*

ternational Conference on Recent Trends in Information Technology (ICRTIT), IEEE, 2016.

[33] M. T. Al-Bayati and M. M. Al-Jarrah, "DuoHide: A secure system for hiding multimedia files in dual cover images," in *2016 9th International Conference on Developments in eSystems Engineering (DeSE)*, IEEE, 2016.

[34] R. Jafari, D. Ziou, and A. Mammeri, "Increasing compression of JPEG images using steganography," in *2011 IEEE International Symposium on Robotic and Sensors Environments (ROSE)*, IEEE, 2011.

[35] X. Li, S. Kang, and Y. Sakamoto, "A partial scrambling technique with adjustable degradation degrees embedding different types of contents," in *2016 International Workshop on Advanced Image Technology (IWAIT)*, vol. 2C, 2016.

[36] X. Li, Y. Miura, S. Kang, and Y. Sakamoto, "A scrambling technique embedding soundtracks into videos for streaming media," in *2018 International Workshop on Advanced Image Technology (IWAIT)*, pp. 1–4, IEEE, 2018.

[37] X. Li, S. Kang, and Y. Sakamoto, "Lossless data hiding technique reducing cover data size for compressed videos," *Journal of Electronic Imaging*, vol. 33, no. 01, 2024.

[38] F. Dufaux and T. Ebrahimi, "Scrambling for privacy protection in video surveillance systems," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, no. 8, pp. 1168–1174, 2008.

[39] M. Matsumoto and T. Nishimura, “Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator,” *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, vol. 8, no. 1, pp. 3–30, 1998.

[40] E. Y. Lam and J. W. Goodman, “A mathematical analysis of the DCT coefficient distributions for images,” *IEEE Transactions on Image Processing*, vol. 9, no. 10, pp. 1661–1666, 2000.

[41] N. Kamaci and G. AlRegib, “Improved DCT coefficient distribution modeling for h.264-like video coders based on block classification,” in *2011 18th IEEE International Conference on Image Processing*, IEEE, 2011.

[42] C.-K. Chan and L.-M. Cheng, “Hiding data in images by simple LSB substitution,” *Pattern recognition*, vol. 37, no. 3, pp. 469–474, 2004.

[43] K. Sawada and S. Kang, “Degraded watermarking method using pseudo-random number,” in *Proceedings of the Engineering Sciences Society Conference of IEICE*, no. A-7-18, IEICE, 2007.

[44] S. Shimaura, S. Harada, S. Kang, and Y. Sakamoto, “An evaluation on controlling file size increase in the image partial scrambling technique enabled quantitative degradation degree control by exchanging DCT coefficients,” in *IEICE Technical Report, 113(136)*, pp. 167–172, IEICE, 2013.

[45] Y. Nemoto, Y. Toyota, S. Sakazawa, L. Zhao, and H. Yamamoto, “A study on video scrambling considering inter-frame prediction,” in *IEICE technical report, Image engineering 105(500)*, pp. 207–212, IEICE, 2006.

[46] K. Wong and K. Tanaka, "DCT based scalable scrambling method with reversible data hiding functionality," in *2010 4th International Symposium on Communications, Control and Signal Processing (ISCCSP)*, pp. 1–4, IEEE, 2010.

[47] R. M. Rad and K. Wong, "An efficient sign prediction method for DCT coefficients and its application to reversible data embedding in scrambled JPEG image," in *2013 IEEE International Conference on Image Processing*, pp. 4442–4446, IEEE, 2013.

[48] M. Takayama, K. Tanaka, A. Yoneyama, and Y. Nakajima, "A video scrambling scheme applicable to local region without data expansion," in *2006 IEEE International Conference on Multimedia and Expo*, pp. 1349–1352, IEEE, 2006.

[49] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

[50] Y. Miura, X. Li, S. Kang, and Y. Sakamoto, "Additional data hiding technique for specified regions of JPEG image," in *IEICE Technical Report, 116(34)*, pp. 1–6, IEICE, 2016.

[51] B. G. Mobasseri and M. P. Marcinak, "Watermarking of MPEG-2 video in compressed domain using VLC mapping," in *Proceedings of the 7th workshop on Multimedia and security*, ACM, 2005.

[52] M. Chaumont and W. Puech, “A DCT-based data-hiding method to embed the color information in a JPEG grey level image,” in *2006 14th European Signal Processing Conference*, 2006.

[53] F. Hartung and B. Girod, “Digital watermarking of MPEG-2 coded video in the bitstream domain,” in *1997 IEEE International Conference on Acoustics, Speech, and Signal Processing*, IEEE Comput. Soc. Press, 1997.

[54] A. Westfeld, “F5—a steganographic algorithm,” in *Information Hiding*, pp. 289–302, Springer Berlin Heidelberg, 2001.

[55] S. A. Parah, J. A. Sheikh, U. I. Assad, and G. M. Bhat, “Hiding in encrypted images: a three tier security data hiding technique,” *Multidimensional Systems and Signal Processing*, vol. 28, no. 2, pp. 549–572, 2015.