Data Reduction in MMBD Computing

Yosef Hasan Jbara

Abstract Internet of Things technology is emerging very quickly in human facilities of all types, such as smart home and industry, which leads to a large boom in multimedia big data due to the connection of approximately 50 billion devices to the internet in 2020. It is really a challenging task to manage the IoT multimedia data regarding storage and transmission. The only way to handle this complicated storage and transmission problem is the process of compression techniques. Multimedia data is compressed by reducing its redundancy. Compression algorithms face numerous difficulties because of the large size, high streaming rate, and the high quality of the data, due to their different types and modality of acquisition. This chapter provides an overarching view of data compression challenges related to big data and IoT environment. In this chapter, we provide an overview of the various data compression techniques employed for multimedia big data computing, such as run-length coding, Huffman coding, arithmetic coding, delta modulation, discrete cosine transform, fast Fourier transform, joint photograph expert group, moving picture expert group, and H.261, including the essential theory, the taxonomy, necessary algorithmic details, mathematical foundations, and their relative benefits and disadvantages.

Keywords Internet of Things · Multimedia big data · JPEG · MPEG · Quality of service · CODEC

1 Introduction

The Internet of Things (IoT) will induce a vast influx of big data as a result of the massive spreading of sensors across almost every industry. The emerging popularity of IoT-based Internet-friendly low-cost devices and the explosive use of smart device technology, drive users to generate and transmit multimedia data over the internet.

Y. H. Jbara (✉)
Computer Science, College of Engineering and Information Technology,
Buraydha Colleges, Al Qassim, Saudi Arabia
e-mail: yosefjbara@ieee.org

© Springer Nature Singapore Pte Ltd. 2020
S. Tanwar et al. (eds.), *Multimedia Big Data Computing for IoT Applications*, Intelligent Systems Reference Library 163,
https://doi.org/10.1007/978-981-13-8759-3_8

IoT-enabled devices and objects upload tremendous amounts of audio, image, video, and text data in real time [1]. In 2012, the number of radio-frequency identification (RFID) tags and IoT-enabled devices in commercial use was 12 million. This quantity is predicted to increase to 210 billion in 2021. The multimedia data from IoT-enabled devices belong to big data sets, and this information is typically gathered for analysis and control. The total turnover of the multimedia big data business was 10.2 billion dollars in 2011. It grew to 54.3 billion dollars in 2017. There will be a boom in multimedia big data due to the connection of approximately 50 billion users to the internet in 2020. Therefore, the two technologies IoT and multimedia big data are growing in parallel. IoT is a network of smart devices that collect and exchange data. It has been estimated that trillions of the data will be created every hour by 31 billion devices in 2020 and 76 billion in 2025 [2].

The substantial amount of multimedia content that is obtainable among IoT components contains enormous amounts of text, audio, image, video, and animation data. These data originate from entry-level IoT products such as Amazon Dash Buttons to virtual reality (VR) and augmented reality (AR) IoT devices such as Samsung Gear VR [3]. In this big data era, multimedia data have become a unique type of big data because of the colossal generation of multimedia datasets and their subsequent storage, transmission and processing.

The traditional multimedia compression techniques face challenges because of the characteristics, namely, massive volume and complex structure, of IoT data sets [4]. Multimedia big data are any combination of large-scale signals such as text, graphics, art, sound, and video elements, which all are partially unknown complex structures in nature. All these multimedia component signals have a skeletal resemblance and are described in terms of a statistical parameter in time and space, drift curvature, redundancies, and the quality of experience. Naturally, the modeling and representation of such unstructured, multimodal and heterogeneous multimedia big data are complicated processes.

The processing steps of the life cycle of multimedia big data are acquisition, compression, storage, processing, interpretation, assessment, and security. Initially, the big data are acquired from IoT-enabled devices. Before further storage and processing, the data are compressed and interpreted. The data transfer is subject to security checks, and the assessment is carried out with acquired data and credible data. These multimedia big data are compressed to ensure efficient storage and communication, as their vast volume exceeds the capacity of the transmission channel and the storage capacities of computing machines. Moreover, multimedia big data face challenges in achieving fast storage and processing due to their large data volumes [5].

Multimedia data reduction results in two main benefits: reduced memory size in the local disk and increased speed of data transfer between the network and the disk. Multimedia big data face substantial problems, after acquisition from sensors, RFIDs, and interconnected smart devices in storage and processing. Therefore, the compression of multimedia big data is an inevitable process in IoT applications [6]. The initially acquired data require an effective compression algorithm that uses time domain or frequency domain methods, to overcome the limitations of the traditional memory and computational resources. Such compression algorithms face numerous difficulties that are due to the enormous size, faster streaming rate, quality, and various types and modalities of acquisition of the data. The very fast multimedia big data reduction techniques must be employed, to manage the large data volume [7].

In a multimedia data reduction technique, an encoding process is applied to represent that data using fewer bits by utilizing a precise encoding system. Storage, throughput, and interaction are the fundamental coding requirements for data reduction techniques. The multimedia data reduction techniques are of high significance in IoT because of the reduction in disk space for storage, which enhances reading and writing. The speed of file transfer is also increased with variable dynamic range and byte order independence via these reduction techniques [8]. Additional disk space will be needed to store multimedia components such as images and graphics, audio, videos, and animations. We require 175 kB of memory space to save audio that has been recorded for one second in stereo quality. A 320 $\times$ 240 $\times$ 8-bit grayscale image requires 77 kB memory space, a 1100 $\times$ 900 $\times$ 24-bit color image requires 3 MB disk space, and a 640 $\times$ 480 $\times$ 24 $\times$ 30 frames/s video requires 27.6 MB/s of memory. The interaction between the sender and receiver should be very fast to avoid transmission delays in the multimedia system [9].

Compression methods are broadly classified into lossless and lossy methods. In a lossless compression technique, the original data are completely recovered from the compressed data. Every bit of the original data is recovered from the compressed data. A lossy compression technique reduces the acceptable level of data in the coding process and rebuilds an estimation of the original data in decoding [10].

The fundamental categories of multimedia data reduction techniques are entropy coding, source coding, channel coding, and hybrid coding. Methods in the entropy coding category use run-length coding, Huffman coding, and arithmetic coding. DPCM and DM are the most widely used prediction techniques in source coding. Prior to the encoding, the original multimedia data are transformed into a compact data format to realize more effective coding by employing fast Fourier transform (FFT) and discrete cosine transform (DCT) transformations [11]. The most widely used compression models under the hybrid coding category, are Joint Photographic Experts Group (JPEG), Moving Picture expert group (MPEG), and H.261, which are utilized for the compression of multimedia such as text, graphics, art, sound, and video.

The run-length coding technique replaces each repeated value by the length of its run and its value based on entropy. Compression is realized by eliminating the redundancy that is due to the continuous occurrence of the same value [12]. The Huffman coding algorithm uses the shortest code for the most frequently occurring values and the longest code for the least frequently occurring values based on entropy. Compression is realized by using variable bit lengths for different values.

In arithmetic coding allows “blending” of bits in a data stream are “blended” and the total number of required bits is calculated based on the sum of the self-information using a message interval and a sequence interval. In the differential coding algorithm, the difference between two continuous data values is encoded instead of the data [13]. In vector quantization, the original data are split into chunks of bytes. Then, each chunk is encoded with a list of patterns. Lempel–Ziv–Welch (LZW) compression uses a code table with 4096 data patterns. If the input data are matched with an entry in the table, the corresponding data are replaced with that matched pattern.

JPEG is the first lossy monochrome and color image compression norm and was developed by the Joint Photographic Experts Group. The functional systems of JPEG are the baseline system, an extended system, and a special lossless function. The first level, namely, the baseline system, decompresses color images to realize the required level of compression. Various encoding techniques, such as variable-length encoding, progressive encoding, and hierarchical mode encoding are processed in the second level of the JPEG extended system [14]. In the last level, the special lossless function guarantees the resolution of the multimedia data, which ensures that there is an exact match between the compressed and original images.

The four phases of the JPEG compression process are preparation, processing, quantization, and entropy encoding. In the preparation phase, color space conversion is performed. The color image is converted into Y, C_b, C_r from RGB format. The processing stage divides the luminance and chrominance information into 8×8 square blocks and applies a 2D DCT to each block, which removes the preliminary data redundancy. The DCT translates the multimedia data of each 8×8 block from spatial information into an efficient frequency space representation that is better suited for compression.

In this stage, quantization is applied on 8×8 DCT coefficients by discarding the grayscale and color information. The quantization process reduces the DCT coefficients that correspond to less than the value of the basis function to zero. Now, the DCT array contains more zeros, which leads to sufficient compression in the final stage. In the final entropy coding stage, quantized DCT coefficients are organized and arranged in a "zigzag" pattern. Then, the run-length encoding algorithm is employed on similar frequency groups and the inserted length coding zeros. The remaining pixels are encoded using Huffman coding.

MPEG, which stands for Moving Picture Expert Group, is an international standard for both audio and video compression. It decreases the required storage and space bandwidth via means of both encoding and decoding processes [15]. The standards are MPEG 1, MPEG 2, and MPEG 4, which differ in terms of the video resolution, the frame rate, and the bit rate. MPEG 1 supports source input format (SIF) videos, with resolutions of 352×288 pixels at 25 frames/s for the Phase Alternating Line (PAL) system and 352×240 pixels at 30 frames/s for the National Television Systems Committee (NTSC) system and a bit rate of 1.5 Mb/s. The maximum target bit rate of MPEG-2 is between 4 and 9 Mb/s, with 720×480-pixel resolution video at 30 frames/s [16]. In MPEG, video is divided into macroblocks via a video sequence layer, a grouping layer, and a slice layer. Each macroblock consists of 16×16 arrays of luminance pixels and two 8×8 arrays of associated chrominance pixels. The macro blocks are further divided into distinct 8×8 blocks and transform coding is applied on each block. The remaining processes are the same as in JPEG.

This chapter is prepared to deliver complete details of the various categories of compression methods, by including a full and useful nomenclature, a detailed investigation in terms of architecture, advantages, and shortcomings and collective practices. We also delve into the most significant impact of the compression of multimedia big data, with an excellent and lucid review of the current multimedia IoT data compression techniques. So we hope this chapter be able to help as a valuable reference for teachers and researcher involved in IoT data compression and processing, which has remarkable openings in both education and employment. This chapter also delivers an instrumental allusion for IoT, multimedia professionals, and computer scientists. We organized this chapter with five key sections broadly including all features of multimedia data compression: Multimedia basics, Elements, and File Formats, Quality of Service, Principles of Compression Techniques, and Image Compression Standards. In this chapter, we discuss the most significant impact of the compression of multimedia big data. We hope this chapter will serve as a valuable reference for teachers and researchers who are involved in IoT data compression and processing, which have remarkable applications in both education and research.

The remainder of this chapter is organized into four sections: In Sect. 2, we describe the multimedia elements, the preliminaries of differential formats and quality of service. Section 3 reviews several multimedia compression techniques and their principles. We discuss critical open problems in reference models and protocols and recent developments along with future directions in Sect. 4. Lastly, Sect. 5 concludes this chapter with an encapsulated summary of data reduction in multimedia big data computing.

2 Multimedia Basics

The advancement of IoT has had a large impact on multimedia data. Multimedia is an interactive form of communication in which multiple types of media data such as text, audio, graphics, animation, and video, are conveyed. In recent years, with significant advances in the state-of-the-art semantic web techniques and technologies, IoT has explored a tremendous amount of multimedia big data [17].

According to a recent survey by Cisco and IBM, every day, IoT-enabled devices, such as smart consumer appliances and lab instruments, will generate 2560 quintillion data (1 quintillion = 10 17). They also predicted that this amount of IoT multimedia data will increase to 40 yottabytes in 2020 (1 yottabyte = 1024 bytes) and every person will generate 5200 GB per day in the IoT era. Because of this extremely large volume and the intangible nature of fundamental data formats, IoT multimedia will encounter substantial problems in storage and transmission. Therefore, it is essential to understand exclusively the basics of multimedia big data and to have the ,in-depth knowledge of the file formats and the quality of service of multimedia big data systems [18]. To handle multimedia data, three fundamental requirements must be specified for compression, namely, the video frame size, frame rate, and synchronization of audio and video, along with the encoding and decoding speeds, end-to-end delay and data retrieval rate.

The data that are collected and exchanged between smart devices are multimedia data, which are a complex mixture of audio, video, text, image, and animation. It is challenging to handle this vast amount of data. Big data techniques are used to manage the large data that have been obtained from IoT devices and other sources. Since big data are large and complex, it is challenging to store and retrieve the rapidly increasing volume of multimedia data from smart devices.

For multimedia big data, substantial problems are encountered after acquisition from sensors, RFIDs, and interconnected smart devices in storage and processing. Thus, the compression of multimedia big data is an inevitable process in IoT applications. The initially acquired data require an effective compression algorithm that uses time domain or frequency domain methods, to overcome the limitations in the traditional memory and computational resources. These compression algorithms face numerous difficulties because of the enormous size, high streaming rate and high quality of the data, which results from the various types and modalities of acquisition.

The challenging task of storing multimedia data is performed by utilizing one of the most critical signal processing techniques: data compression. Data compression is a complex process that involves sequences of several operations. An encoding operation with an appropriate representation technique removes the redundancy in the data set, which results in an overall reduction in the size of data. However, multimedia compression uses various types of tools and procedures to decrease the file sizes for text, audio, image, video and animation formats. In turn, multimedia compression reduces the amount of memory that is necessary for storing media files and reduces the amount of bandwidth that is required for the transmission of media files.

2.1 Elements and File Formats

Multimedia is a complex type of data, with a combination of audio, animation, graphics, image, text, and video data. Any IoT application consists of any or all the multimedia elements.

This section discusses the most important data types that are used in multimedia big data in IoT applications. This basic information provides the reader with the necessary background knowledge for understanding the compression techniques and standards of this big data era.

2.1.1 Text

Text and symbols are fundamental data that are important for communication in any medium. The forms of text information include ASCII/Unicode, HTML, Postscript, and PDF. The standard file type for text only is TXT and those for text with other elements are DOC, DOCX, and PDF [19].

2.1.2 Graphics and Images

These are important data elements in IoT multimedia big data computation. The main sources for image data in IoT are digital still cameras and video cameras. All computers and software compatibly support available file formats such as GIF (Graphics Interchange Format), TIFF (Tagged Image File Format), JPEG (Joint Photographic Experts Groups), EPS (Encapsulated PostScript), BMP (Bitmap Image File), PNG (Portable Network Graphics), and RAW [20]. In this section, we present introductory information of various image and graphics file formats.

TIFF was developed by Aldus and is best-suited for high-quality prints, professional publications, and archival copies. This TIFF file format requires a large memory space for storage because of its large size. An image that is in this format can be stored in bi-level (black and white), grayscale, palette (indexed), or RGB (true color). Compression is optional for TIFF images. Huffman and LZW algorithms are used for compression. After compression, the file is larger compared to GIF and JPEG files. TIFF images can be displayed by web browsers using inbuilt plug-ins [21].

Microsoft developed the BMP image file format for their Windows operating system. Like TIFF, BMP images are of high quality, with or without compression but require more memory space for storage. This file format is device-independent and can be displayed by any display by Windows operating systems. The bitmap file format supports monochrome, 16-color, 256-color, and 224-color images. The default file extension is .bmp [22].

JPEG is a standardized image compression mechanism. It was devised by the Joint Photographic Experts Group for compressing either full-color or grayscale images via a lossy encoding technique. It performs well with photographs and artwork. The size of the image file is small because it stores 24-bit-per-pixel color data instead of 8-bit. In the last entropy coding stage, quantized DCT coefficients are arranged in a "zigzag" pattern. A run-length encoding algorithm is employed on similar frequency groups and inserted length coding zeros. The remaining pixels are encoded using Huffman coding. It can easily realize a 20:1 compression ratio [23].

GIF is a standard file format that was developed by CompuServ for the storage and transmission of color raster image information. Since it uses lossy compression encoding techniques, the resulting files are small and portable. It can also be used to store screenshots, letters, geometric line drawings, sharp images, and flat color images. It enables the display of high-quality, high-resolution graphics on any display using graphics hardware. It has unique features such as animation and transparency. The file extension is .gif.

The PNG image format was developed by Unisys as an alternative to the GIF format for storing monochrome images with a color depth of 16-bits per pixel, color images with a color depth of 48-bits per pixel and indexed images with a palette of 256 colors [24]. Royalties should be paid to the proprietor Unisys by the user. Its compression ratio exceeds that of GIF by 5–25%. It has gamma correction and error correction mechanisms, which lead to device independence and integrity. The file extension is .png.

The EPS file extension is used for the Encapsulated PostScript image file type, which is a vector file model. It was developed for software applications such as CORELDRAW. Since this image file format contains vector information, no compression technique can be used. Its file extension is .eps.

The RAW file format represents unprocessed "raw" pixels with the intensity values and spatial coordinates of the pixels, which is the same as film negative, but in digital form. This "raw" pixel information is obtained directly from image acquisition devices such as video camera sensors. In this raw image file, each pixel is associated with an RGB intensity value. These RGB values undergo a demosaicking process. Modern digital cameras convert this raw file into a JPEG or TIFF file via proper software processes and store it in the memory card. This image file format is most suitable for photography without compression. The file extensions are .raw, .cr2, and .orf [25].

2.1.3 Audio

Audio can enrich information and strengthen concepts that are presented in the form of graphics or text. The contents of the presentation will be more informative because of the associated audio files [26]. The audio formats that are available for multimedia communication are MP3, WAV, WMA, MIDI, RealAudio, AAC, OGG, and MP4.

MIDI stands for Musical Instrument Digital Interface, which is a software interface between electronic musical instruments and a personal computer through the sound card. Like other audio file formats, MIDI files contain not only sound information but also digital notes and are suitable for playing via electronics. This file format is efficiently handled by both music hardware and computers, but not supported by internet web browsers. The file extension is midi.

Real Media devised an audio file format, namely, RealAudio, that enables the streaming of audio with the file extension.ram or.rm. Unfortunately, web browsers are unable to handle this file type.

Microsoft developed WMA (Windows Media Audio) exclusively for music players. It is suitable for Windows computers, but not web browsers. The file extension is .wma.

Similarly, Apple developed an audio file format, namely, Advanced Audio Coding, for iTunes. It is suitable for Apple computers, but not web browsers. The file extension is .acc.

The file format OGG is only supported by MTML5 and has file extension .ogg. The Xiph.Org Foundation, combined with HTML5, developed this file format.

The famous audio format MP3 is an integral part of the MPEG file format. All music players can access this audio file format effectively. A file of this format has a high compression ratio and high quality and is supported by all browsers. The file extension is .mp3. MP4 is used for both audio and video files and has the extension .mp4.

2.1.4 Video

Videos are widely used to carry information, but require high bandwidth for downloading. Video file formats combine a container file and a codec file. The container file contains bearing all the information about the file structure, along with the type of codec that is being used. The codec contains the entire procedure for coding and decoding the video data. The more common types of container formats that are available in the market are MPEG, MPEG4, AVI, WMV, QuickTime, Real Video, Flash, and WebM [27]; these formats are briefly described below to facilitate understanding of the IoT multimedia compression techniques and standards that are described in the next section.

MPEG, which stands for Moving Picture Expert Group, is an international standard for both audio and video compression, which is used to create downloadable movies. It supports all web browsers, but not HTML5. It requires less storage and bandwidth due to its encoding and decoding processes. The standards are MPEG 1, MPEG 2, and MPEG 4, which differ in terms of video resolution, frame rate, and bit rate. MPEG 1 supports SIF video, has resolutions of 352×240 pixels at 30 frames/s for the NTSC system and 352×288 pixels at 25 frames/s for the PAL system and a bit rate of 1.5 Mb/s. The maximum target bit rate of MPEG-2 is between 4 and 9 Mb/s, with a video resolution of 720×480 at 30 frames/s. It is compatible with both the Apple QuickTime Player and the Microsoft Windows Media Player. The file extension is.mpg or.mpeg. MPEG-4 is specially designed to handle movies in TV and video cameras. It can easily upload audio and video streams online. Moreover, it also supports HTML 5 and YouTube users. In this format, video and audio compression is achieved via MPEG-4 video encoding and Advance Audio Coding. The file extension is .mp4.

The Video Interleave audio format was developed by Microsoft for electronic devices gadgets such as TVs and video cameras. It performs efficiently on Widows-based PCs but is not supported by internet browsers. It stores audio and video data after encoding with different codecs and less compression. Therefore, it is widely used by internet users. It supports all players, including the Apple QuickTime Player, Microsoft Windows Media Player, VideoLAN VLC media player and Null soft Winam on both Windows and Mac platforms. The file extension is .avi.

Microsoft also developed another video file format, namely, Windows Media Video, which has the file extension.wmv for internet streaming applications. Since electronic devices such as TV and video cameras are also compatible with the Windows Media Video file format, videos are stored in this format. It also performs efficiently in Windows-based PCs but is not supported by Internet browsers. After the installation of a plug-in, Mac and UNIX-based computers can also play this format [28].

The ASF file format was developed by Microsoft for synchronized streaming videos. It is available as a container file that consists of WMA (Windows Media Audio) and WMV (Windows Media Video) with copyrights. It is also available with many other codecs. It supports still image, text, and metadata. The OS-independent version is used to deliver multimedia across various protocols. Its file extension is.asf, which stands for Advanced Streaming Format.

QuickTime is a multimedia technology that was developed by Apple Computer for producing graphics, sound, and movies. It is also used to store such multimedia content. It has been developed for wide use on both Mac and PC. It is rarely used on the internet. The file extension is .mov.

The Real Media developed the RealVideo video file format for the internet. It allows low bandwidth video stream on the internet with a reduced amount of quality. Web browsers can't use this file format. The file extension is.ram. The RealNetworks created another video file format named RealMedia. To stream the media file over the internet with audio and video content, RealMedia file format is used. RealMedia is supported by both Mac and Pcs media players with the file extension .rm.

Flash video format is compact, complex and very common in all browsers. Macromedia has the ownership of this file format. Flash video format works well and good in Pcs, Mac, and UNIX platforms. Because of its cross-platform usage, it reaches all users progressively. It supports both progressive and streaming downloads with compression. The file extension is .flv.

Macromedia developed the Flash movie format with the inclusion of text, graphics, and animation. It needs flash plug-in to play in web browsers. So, all the latest web browsers are preinstalled with this flash plug-in. The file extension is .swf.

The 3GP format is exclusively designed to transfer both audio and video data files among 3G mobile and the internet. In most of the smartphone acquire videos and transfer in online using this file format. It is accessible by the "Apple QuickTime Player, Real Networks Real Player, Video LAN VLC media player, M Player MIK-SOFT, and Mobile 3GP Converter", both in mac and windows operating systems. The file extension is .3gp.

The browser developers "Mozilla, Opera, Adobe, and Google" devised the video file format named WebM to support HTML5. The file extension is.webm. Theora Ogg is HTML-5 supported file format and Xiph.Org Foundation is the owner of this file format. The file extension is .ogg.

2.1.5 Animation

Animation is an illustrative concept of moving graphics. The animation effect depends on the characteristics of the graphic that is used for the animation. Various tools, such as Authorware, Dreamweaver, Director, and Flash are used to generate animations. Flash can animate bitmap graphics and vector graphics; vector animation files are smaller than bitmap animation files. Open-source tools such as pencil and blender are also used to create simple 2D animations [29].

2.2 *QoS—Quality of Service*

An essential metric in multimedia networks, real-time multimedia systems, and distributed multimedia is quality of Service (QoS). QoS may be dynamic and its purpose is to provide an acceptable level of system performance assurance. QoS may be established for two general aspects: services and traffic. It is essential to set the QoS level for the system and require the same QoS level for traffic. The capacity of a multimedia network is measured according to the level of the guarantee for satisfying the requirements of service and traffic [30].

The variation in the QoS parameter over time complicates the maintenance of QoS in distributed multimedia systems. Therefore, we have to monitor the QoS level continuously and maintain the required QoS level by blocking lower priority tasks when necessary.

The two types of QoS are currently available in multimedia big data systems and networks: resource reservation QoS (service) and prioritization QoS (traffic) The objective of service QoS is to allocate system resources according to the network's bandwidth management policy. The traffic QoS assigns system resources according to the broadcast requirements. Since these QoS types are opposite in nature, any multimedia system must be designed to use these two QoS types in a coordinated manner to satisfy the network requirements.

The main application functionalities of a multimedia data system are common consumer gratification and correct demonstration of the hypermedia artifact to the user. Starting audio 5 s after video and synchronizing rolling text and a voice-over are examples of application functionalities. QoS depends on both the application and the network and is subject to negotiation between system components. A video game and a web page differ in terms of QoS factors, even though they are both multimedia applications.

The five categories of distributed multimedia QoS parameters are oriented in terms of performance, format, synchronization, cost, and user. The end-to-end delay, bit rate, and jitter are the QoS parameter for performance-oriented category. The format-oriented QoS parameters are the video resolution, frame rate, and storage format. The skew between the beginnings of sequences is the synchronization-oriented QoS parameter. The QoS connection and data transmission charge parameter belong to the cost-oriented category. The QoS parameters that describe the image and sound quality are user-oriented QoS parameters. The relative significance of these five parameter types depends heavily on the multimedia application to be distributed. The significance of the end-to-end delay differs between noninteractive applications such as "presentation" and interactive applications such as "conversational".

QoS factor processing involves the following three linked operations in a multimedia distributed system:

1. Evaluating the QoS factor that corresponds to customer satisfaction with the execution performance.
2. Relating QoS factors with the evaluation of system outcomes.
3. Modifying system components to satisfy individual requirements.

3 Principles of Compression Techniques

The fundamental categories of data compression techniques are entropy coding, source coding, channel coding, and hybrid coding. The main principle of entropy coding is to ignore the semantics of the data, which falls under the category of lossless compression. Source coding is a lossy compression technique that compresses the data based on the semantics of the information. In the communication channel, we adapted the channel coding, which introduced redundancy during encoding. Hybrid coding combines entropy and source coding. Each category of compression has a technique for data reduction. The three techniques of entropy coding are run-length coding, Huffman coding, and arithmetic coding. Source coding uses two types of prediction technique: DPCM and DM; two types of transformation techniques: FFT and DCT; and three types of coding techniques: bit position, subsampling, and subband coding. Vector quantization is another technique in source coding. JPEG, MPEG, and H.261 are coding techniques in the hybrid coding category of compression [31].

3.1 Run-Length Coding

Run-length coding is suitable if a data stream has a long sequence of identical symbols, such as "ABCEEEEEEDACBBBBB". Figure 1 illustrates the algorithm for RLE.

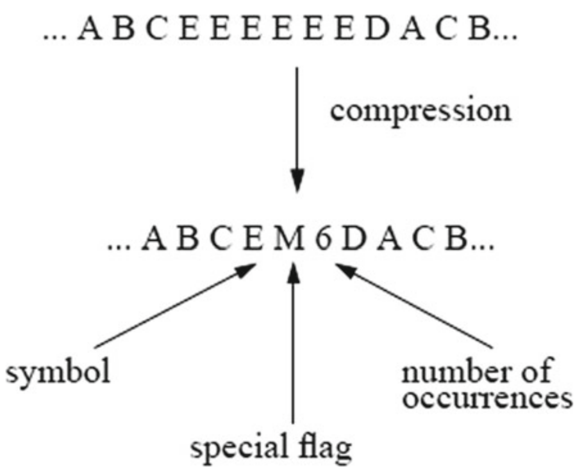


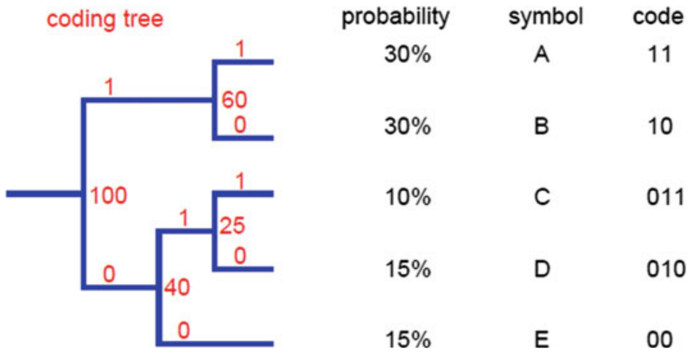
Fig. 1 Run-length encoding algorithm

3.2 *Huffman Coding*

Huffman coding is most suitable for compression if the data stream has more occurrences of some symbols than others, as in the example that is shown in Fig. 2. The fundamental principle is to encode the repeatedly occurring symbols with codes of shorter length. Figure 2 shows the encoding procedure for Huffman coding. The resultant coded output is 111001101000.

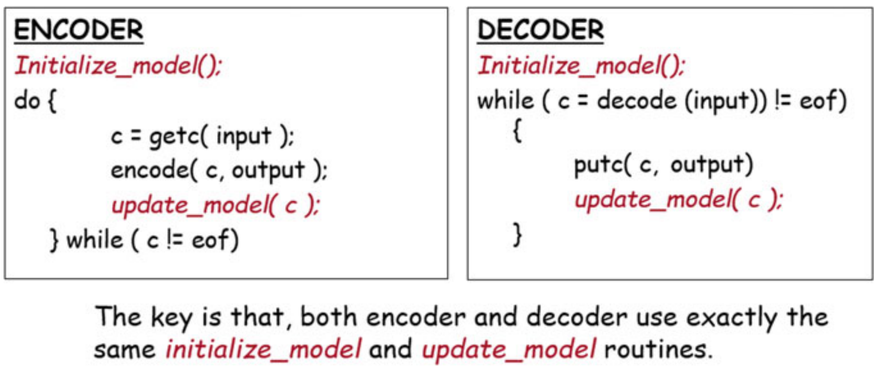
3.3 *Adaptive Huffman Coding*

The main limitation of Huffman coding is its dependence on the statistical knowledge of the data. Since statistical knowledge is not available for live video and audio, adaptive Huffman coding is better suited for compressing live data. The adaptive Huffman encoding and decoding algorithm is shown in Fig. 3.



probability	symbol	code
30%	A	11
30%	B	10
10%	C	011
15%	D	010
15%	E	00

Fig. 2 Huffman encoding procedure



```
ENCODER
Initialize_model();
do {
          c = getc( input );
          encode( c, output );
          update_model( c );
    } while ( c != eof)
```

```
DECODER
Initialize_model();
while ( c = decode (input)) != eof)
   {
          putc( c, output)
          update_model( c );
   }
```

The key is that, both encoder and decoder use exactly the same *initialize_model* and *update_model* routines.

Fig. 3 Adaptive Huffman encoding and decoding algorithm

3.4 *Arithmetic Coding*

The basic principle of arithmetic coding is to encode frequently occurring symbols with the minimum number of bits and rarely occurring symbols with higher numbers of bits. Each symbol of the ensemble narrows the interval between 0 and 1 to reduce the number of bits. Arithmetic coding assumes an explicit probabilistic model of the source [32, 33]. Arithmetic coding uses the symbol-wise recursive method, in which a single data symbol is used for each iteration. It estimates the separation of the interval between 0 and 1. The entire interval is divided according to the probabilities of the symbols. For each symbol, this algorithm estimates the interval in one recursion. Thus, it encodes the data by generating a code string that corresponds to a fractional value that lies between 0 and 1.

Figure 4 illustrates the arithmetic coding for an ensemble, namely, $p(\mathrm{A}) = 0.2$, $p(\mathrm{B}) = 0.3$, and $p(\mathrm{C}) = 0.5$, and the code generation is presented in Table 1.

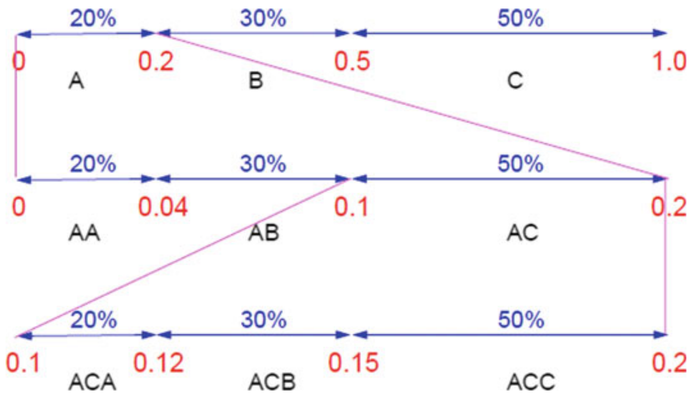


Fig. 4 Code generation for arithmetic coding

Table 1 Arithmetic-coded output

Symbols	Lower bound	Upper bound	Output
A	0.0	0.2	–
AC	0.1	0.2	–
ACB	0.12	1.15	1
A	0	0.2	–
AA	0	0.02	0

3.5 Source Coding

The compression can be very efficiently performed on frequency domain data, compared to time domain counterparts. In this compression method, the original time domain data are transformed into their frequency domain counterparts via FFT and DCT. Subband source coding compresses only the critical frequency ranges of the data stream and discards the other frequency ranges. This method is used in vocoder for speech communication [34].

The simple audio compression systems are G.711, G.722, and G.723. The G.711 audio coding scheme uses pulse code modulation (PCM) with 64 kbps. The G.722 audio coding scheme uses delta pulse code modulation (DPCM) with data rates of 48, 56, and 64 kbps. The G.723 audio coding scheme uses Multi-pulse maximum likelihood quantizer data rates of 6 to 3 kbps and algebraic codebook excitation linear prediction (ACELP) rates of 5 to 3 kbps.

Video and audio conferencing compression methods that are used currently are G.728, AV.253, and IS-54. The G.728 scheme is employed using the low delay code excited linear prediction (LD-CELP) coding with a coding speed of 16 kbps and a duration of less than 2 ms. This complex CODEC algorithm requires 40 MIPS to complete coding and decoding. The coding speed of AV.253 is 32 kbps. IS-54 uses VSELP with a coding speed 13 kbps. It performs well on voice data and poorly on music.

The two types of coding methods for mobile telephone networks are RPE-LTP (GSM) and GSM half-rate coders. RPE-LTP (GSM) stands for regular pulse excitation—long-term predictor. Its coding speed is 13 kbps and it is used for speech communication in European GSM networks. The coding speed range of GSM half-rate coders is 5.6–6.25 kbps.

JPEG stands for Joint Photographic Expert Group, which developed this compression standard with the joint support of ISO/IEC JTC1/SC2/WG10 and Commission Q.16 of CCITT SGVIII. It is an international standard for the coding and compression of both grayscale and color images with a compression ratio of 10:1. JPEG is a common compression system. It does not depend on the image resolution, image or pixel aspect ratio, color coordinate system, image complexity, or statistical characteristics. It is a clear interchange format of encoded data that is implemented in both software and hardware. MOTION JPEG is used for video compression by encoding image sequences.

4 Image Compression Standards

IoT-enabled digital imaging devices, such as scanners and digital cameras, produce a massive amount of image data, which motivates the development of image compression standards for efficient processing and storage. The JPEG standards are JPEG, JPEG XT, JPEG-LS, JPEG 2000, JPEG XR, JBIG, AIC, JPSearch, JPEG Systems, JPEG XS, and JPEG Pleno. In this section, we demonstrate that the JPEG STANDARD, which is the wavelet-based JPEG 2000 standard, which is more efficient in compressing IoT multimedia big data.

4.1 JPEG Standard

The Joint Photographic Experts Group developed the JPEG standard for the efficient compression of images in 1992, which was the result of continuous efforts that began in 1986. They released the latest version of JPEG in 1994. JPEG is a lossy compression method, in which a high compression ratio was realized by discarding some visual information [35]. This loss of information did not affect the visual quality of the natural scene images. This will have an effect in medical images. In the JPEG compression standard, the reconstructed image is not an exact replica of the original image. The compressed output data stream is produced from the input image, with a continuous process of DCT, quantization and binary encoding on 8 $\times$ 8-pixel blocks.

The five functional steps of the JPEG image compression technique are color space conversion, spatial subsampling, DCT, quantization, and coding, as illustrated in Fig. 5. In the first stage, the RGB image is converted into Y, C_b, C_r. In the second stage, the C_b and C_r image components are subsampled separately with a size of 2 $\times$ 2 pixels. In the third stage, the spatial domain data are converted into their frequency domain counterparts by applying 2D DCT on Y, C_b, C_r individually. The frequency domain information is quantized in the fourth phase and, finally, the quantized frequency domain information is encoded using Huffman coding.

The JPEG encoding process is illustrated in detail in Fig. 6. The processes that are involved in encoding are the color transform, 8 $\times$ 8 DCT, uniform scalar quantization, DPCM, zigzag scan, variable-length coding, run-length coding, and marker insertion. Finally, the compressed data of the input image are output as a JPEG file.

4.1.1 Color Transform and Level Offset

The transformation matrix that is presented as Eq. 1 is used to convert the color image from RGB format to YUV format, which represents the luminance and chrominance

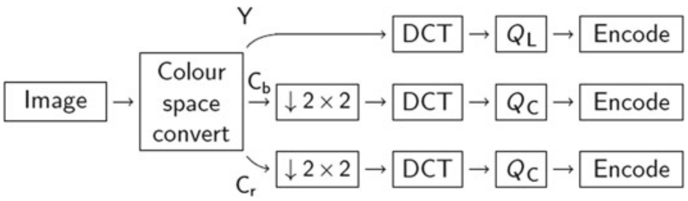


Fig. 5 Functional steps of JPEG image compression

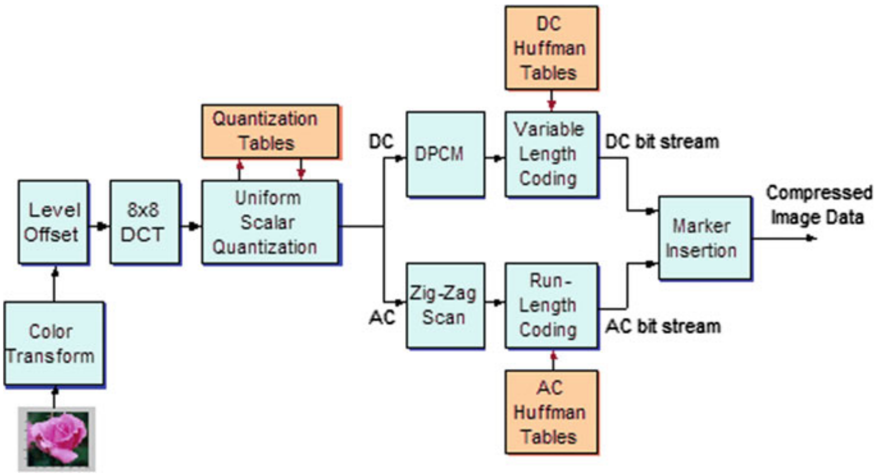


Fig. 6 JPEG algorithm encoding

images. The proper level offset of [0, 128, 128] is also added to the resultant transformed image, namely, YUV. In addition, the further U and V images are subsampled to reduce the bit rate.

$$v_i^T = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ -0.1687 & -0.3313 & 0.5 \\ 0.5 & -0.4187 & -0.0813 \end{pmatrix} u_i^T + \begin{pmatrix} 0 \\ 128 \\ 128 \end{pmatrix}. \tag{1}$$

4.1.2 Discrete Cosine Transform

The 2D DCT transforms the pixels in $f(x, y)$ to the corresponding DCT coefficients, which are denoted as $F(u, v)$ via Eq. 2. The chrominance and luminance images are divided into subimages of size 8×8. Then, DCT is performed on each 8×8 data matrix separately. Figure 7 presents the 8×8 subimage and its DCT coefficient matrix. The value in the upper left corner of the DCT coefficient matrix is the total signal energy. All other coefficients represent the high-frequency components of the image, which contain negligible amounts of visible information and can be discarded with a small amount of distortion. The forward transform of block fb is expressed in Eq. 2.

$$F_b(u, v) = \frac{C(u)}{\sqrt{N/2}} \frac{C(v)}{\sqrt{N/2}} \sum_{x=0}^{N-1} \sum_{yo0}^{N-1} f_b(x, y) \cos \frac{(2x+1)u\pi}{2N} \cos \frac{(2y+1)v\pi}{2N} \tag{2}$$

where $0 \le u, v < 8$ and $C(u) = \left\{ \begin{matrix} \frac{1}{\sqrt{2}} & u = 0 \\ 1 & u > 0 \end{matrix} \right\}$

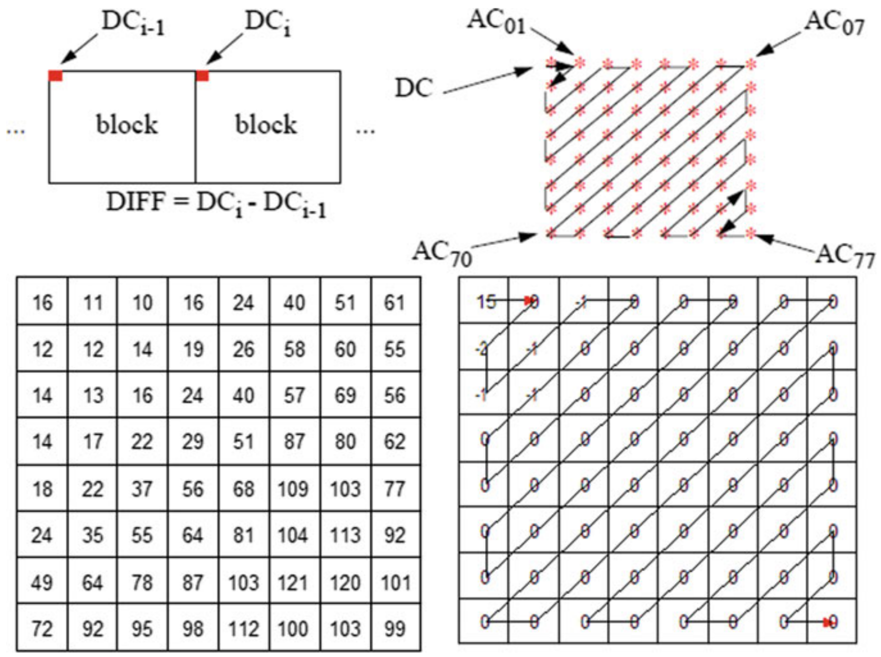
There are 64 DCT coefficients of the 8 × 8 block: 1 DC coefficient and 63 AC coefficients. The DC coefficient of the entire image contains all available information in that 8 × 8 block image. The 63 AC coefficients are ordered in "zig-zag" form with zero coefficients. Then, all the resultant coefficients are transformed into a code via Huffman coding to reduce the number of bits that are required to represent the coefficients. In this stage, a sufficient amount of compression is realized with an acceptable loss of information, without affecting the quality of the image. Since the pixels that correspond to higher frequency coefficients are discarded in DCT, the quality of the resultant image does not affect the visual nature of the original image.

4.1.3 Quantization

If the quantization step size is large, the image distortion is more severe because the image quality and the degree of quantization are inversely proportional to each other, however, the compression ratio is small. The selection of a suitable degree of quantization is crucial in the JPEG compression technique. Since the human eye naturally focuses more on low-frequency areas than on high-frequency areas, JPEG uses a larger quantization step size for a higher frequency components and a smaller step size for lower frequency components. In the 8 × 8 quantization matrix, the upper left coefficient has a small step size and the upper right coefficients have larger step sizes. The quantizer sets many high-frequency coefficients to zero to facilitate coding, hence, higher frequency coefficients lead to efficient compression.

4.1.4 Compression Technique

In the quantization process, many DCT coefficients are set to zero. Then, the quantized DCT coefficients are further encoded via run-length encoding. In run-length coding, each image block is represented by the quantized DCT coefficient and the number of zeros. The result of run-length coding is a record of the number of zeros and the corresponding nonzero coefficient. The DCT coefficients are processed in a zigzag pattern, as illustrated in Fig. 7, to combine the runs of zeros. In addition, it creates a pair of data that contains the information about the current coefficients, where the first element indicates the number of zeros and the second element is the number of bits that are needed to represent the coefficient. This pair was assigned with a variable-length codeword via either Huffman or Arithmetic coding. For every block of data, JPEG specifies two types of information: the code word of the pair and the code word for the amplitude of that coefficient. Along with the two codes, an end-of-block marker will be provided. Then, the same process will be repeated for all the blocks. The final segment data in the output stream is the end-of-file marker.



16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

15	0	-1	0	0	0	0	0
-2	-1	0	0	0	0	0	0
-1	-1	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Fig. 7 Luminance quantization table (left) and the quantized DCT coefficients (right)

4.2 *JPEG Compression Modes*

The JPEG standard employs four modes of compression techniques to compress effectively all types of images: sequential, progressive, lossless and hierarchical. JPEG employs both Huffman and Arithmetic encoding procedures with either 8- or 12-bit resolution. Figure 8 illustrates the compression modes that are associated with the encoding techniques.

In the sequential mode of compression, images are encoded from top to bottom with precisions of 8 and 12 bits. In a single scan, each color component is encoded.

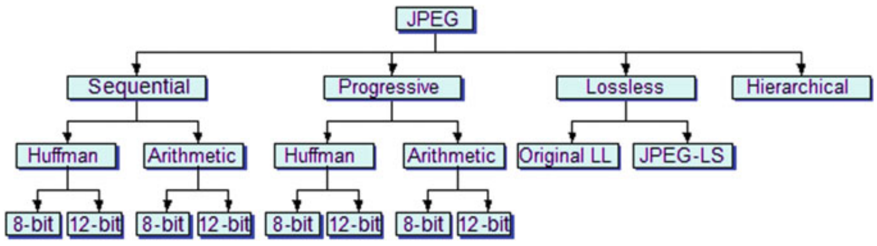


Fig. 8 JPEG operation modes

The sequential mode uses both Huffman encoding and arithmetic coding. In the progressive compression mode, several scans are employed for encoding image modules. The first scan produces a coarse style of the image and the subsequent scans produce progressively smoother versions of that image. Each component scanned a minimum of 2 and a maximum of 896 times. The lossless compression mode produces an exact replica of the original image with a small compression ratio. It is used much less frequently and is more suitable for the compression of medical images.

In the hierarchical compression mode, frames are created from the image by dividing the entire image into subimages. Each frame contains more than one scan of the image. The first frame generates a low-resolution version of the image and the remaining frames generate higher resolution versions of the same image.

4.2.1 JPEG Compressed Data Format

Figure 9 illustrates the JPEG compressed data format. Each data format starts with SOI and ends with EOI. In between SOI and EOI, various scans of the same frame with the frame header are available. The arrangement of a simple compressed image data format corresponds to a frame that begins with the frame header. The table information is provided before the frame header. In between SOI and EOI, scanning information is provided. The define number of lines (DLN) segment separates successive scans.

4.2.2 Decompression of JPEG

The block diagram shown in Fig. 10 provides a detailed functional description of the reconstruction procedure for a compressed JPEG image. This system is the exact reverse of the JPEG compression algorithm.

In the first step, the compressed data stream is applied to the table construction phase, from which the quantization table, DC Huffman table, and AC Huffman table are reconstructed. The decoded header information is used to determine the size and precision of the decompressed image. The DC and AC codes that belong to each 8×8 block are separated from the compressed data stream. The DC is decoded by retrieving the DC coefficient from the DC Huffman table. The sum of the DC coder value and the DC value of the previous block is obtained from the output of IDPCM, which is used as input to the subsequent operations. Similarly, the AC coefficient is retrieved from the AC code via decoding using the Huffman table [36]. The unzigzag operation restores the exact AC coefficient in 8×8 block size. After the dequantization, IDCT reconstructs replica of the original image that is inexact due to the quantization error. The chrominance and luminance images from IDCT are added with the offset value and converted into RGB format, as illustrated in Fig. 11.

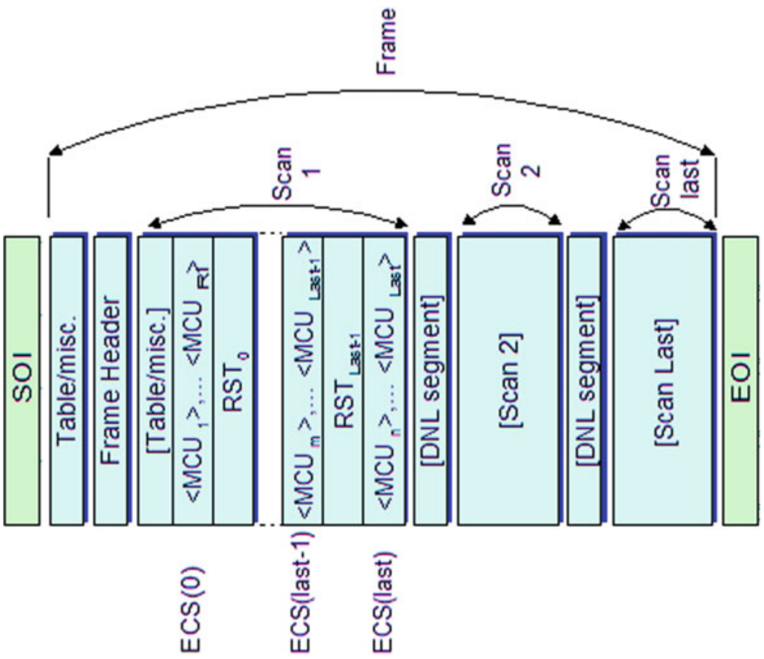


Fig. 9 Simple compressed image data format

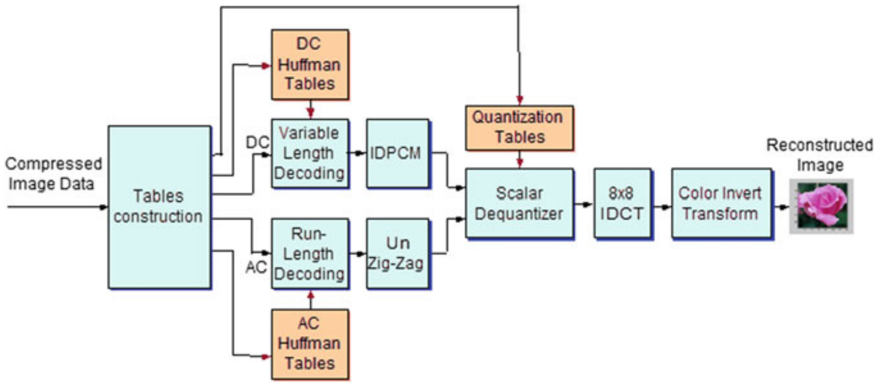


Fig. 10 JPEG decompression structure

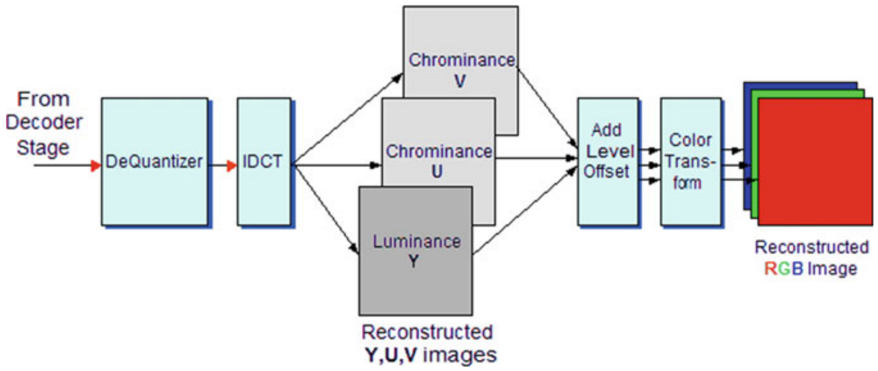


Fig. 11 Color inverse transformation at the decoder

4.3 JPEG 2000

The overall performance of the JPEG standard depends on the f bit rate that is employed for compression, the acceptable amount of data loss and the noisy nature of the communication channel. JPEG is most suitable at high and moderate bit rate, but performs poorly for low bit rates. The wavelet decomposition that is used in JPEG 2000 overcomes the shortcomings of the JPEG standard. It is a collaborative achievement of JPEG with ISO and IEC. The embedded block coding with optimized truncation (EBCOT) algorithm enhances the JPEG 2000 standard to realize superior compression compared to JPEG. The compressed bits stream has resolution and SNR scalability and random access. The main features are lossless and lossy compression, protective image security, region-of-interest coding, and, robustness to bit errors. JPEG 2000 provides higher quality compression using the lossy compression technique at low bit rates. The progressive decoding provides lossless compression. JPEG 2000 protects images with watermarking, labeling, stamping or encryption. The region-of-interest coding technique yields a higher degree of compression with low information loss. The coded bitstream is more robust to transmission errors due to the use of error-resilient tools.

4.3.1 JPEG 2000 Codec

The compression system for the JPEG 2000 standard is illustrated in Fig. 12. The encoding process consists of preprocessing, forward transform, quantization, and entropy coding, similarly to other compression standards. The unique feature of this standard is that each subband has a different step size for the process of quantization and we can skip quantization to achieve lossless compression in the case of medical images [37]. Another significant advantage of this standard is the use of bit plane coding in the process of entropy coding, which assigns high coding priority to the

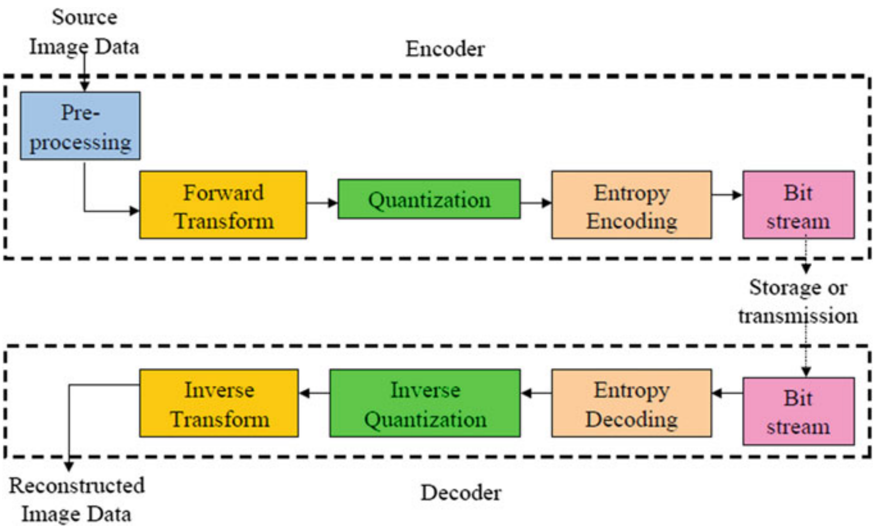


Fig. 12 JPEG 2000 compression engine

MSB plane compared to the other bit planes. In arithmetic coding, this standard encodes the data stream via a context-based approach. This provides higher quality scalability by decoding the bit pane partially. The decoding begins from the MSB bit plane. The quantization step is repeated after discarding the 1-bit plane from the decoding process.

The three preprocessing steps are image tiling, color transformation, and DC level shifting. Among these, image tiling and color transformation are optional for each image component [38]. The sample of each image component is subtracted from the same quantity in DC level shifting. The color transformation transforms RGB into Y, C_b, C_r.

Forward Transform

In this standard, the image is decomposed into multiresolution subbands via the discrete wavelet transform (DWT), which is also an important component of the JPEG 2000 standard. The low-pass samples represent a down-sampled, low-resolution version of the original set and the high-pass samples represent a down-sampled residual version of the original set (details). The 2D DCT decomposed the image into four frequency levels: LH, HL, HL, and HH. These spectral subbands differed from one another in terms of visual quality. The HH subband contains the low-frequency information and LH contains the high-frequency information of the horizontal details. The high-frequency information of the vertical details is available in the HL subband and HH contains the high-frequency information of the diagonal details.

Quantization

After transformation, all coefficients are quantized via scalar quantization. The quantization reduces the precision of the coefficients. The operation is lossy unless the quantization step size is 1 and the coefficients are integers. The quantization formula is presented below.

$$q_b(u, v) = sign(a_b(u, v))\frac{|a_b(u, v)|}{\Delta_b} \tag{3}$$

where

$q_b(u, v)$ is the quantized value
$a_b(u, v)$ is the transform coefficient of subband b
Δ_b is the quantization step size
$|a_b(u, v)|$ is the largest integer that does not exceed a_b.

Modes of Quantization

Quantization is achieved by two modes: the integer mode and the real mode. The integer-mode quantization uses integer-to-integer transforms, a fixed quantization step and bit plane discarding. The real mode uses real-to-real transforms, a rate control-based-quantization step and bit plane discarding [39].

The entropy encoder encodes each code block individually. As illustrated in Fig. 13, each subband is split into several precincts, which are rectangular. The three spatially consistent rectangles comprise a packet. The code block is created after further dividing the precinct into nonoverlapping rectangles. The code blocks in a packet are scanned in raster order. The values of a code block are converted to bit planes [40]. The MSB of each coefficient in the code block is coded via entropy coding. The code blocks, precincts, and packets that are used in JPEG 2000 and the entire process of a single code block are illustrated in Fig. 13.

JPEG 2000 Bitstream and Layers

JPEG 2000 generates an individual stream of codes for each code block, which is contained within a packet. In multilayer encoding of an image, all the encoded data that correspond to a code block are disseminated through packets that correspond to various layers. Figure 14 illustrates the JPEG 2000 bitstream and the structure of the layer [41].

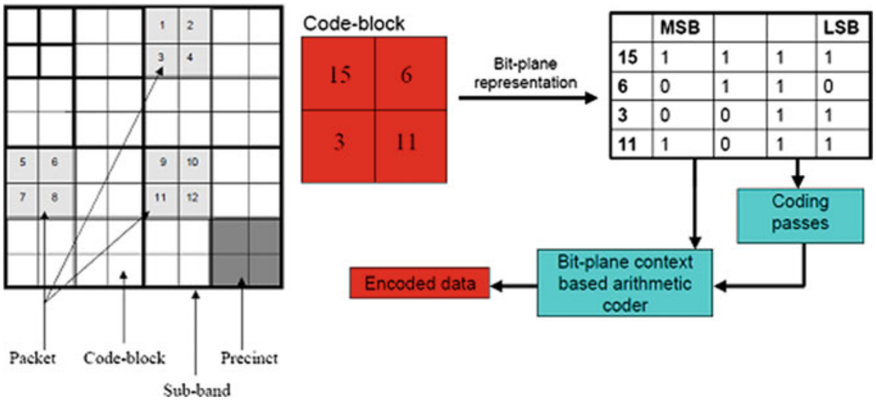


Fig. 13 Code blocks, precincts and packets and code block processing

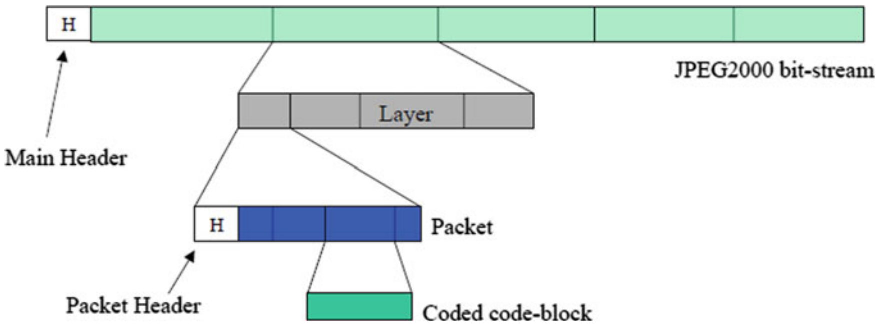


Fig. 14 JPEG 2000 bitstream and the structure of the layer

4.4 H.261 Standard

ITU-T recommended H261 for video compression in video conferencing and video telephony over ISDN lines, in combination with standards H221, H230 H242, and H320, during the period 1988–1990. It performs the decoding and encoding at the rate of p* 64 kbps (the range of p is 1–30). It defines the various components in a video conferencing system. It achieves a high compression ratio by removing both spatial and temporal redundancies of video sequences. It supports the common intermediate format (CIF) with an image resolution of 352 × 288 pixels and quarter CIF (QCIF) with an image resolution of 176 × 144 pixels, with a maximum frame rate of 30 frames per second. The frame rate can be reduced based on the application and bandwidth availability. The input images are coded as a luminance and two color difference components (Y, C_b, C_r). The luminance matrix is double the size of the C_b and C_r matrices [42]. The encoded sequence consists of one I-frame (intra-frame) followed by three consecutive P-frames (inter-frames), as illustrated in Fig. 15. JPEG is used for I-frames and P-frames use pseudo differences from previous frames.

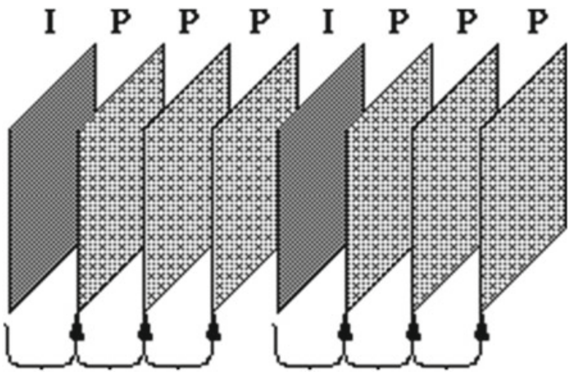


Fig. 15 Encoding sequence

The H.261 encoder is composed of three main steps: Prediction, Block transformation and Quantization & Entropy Coding.

In the prediction step, H261 uses both intra-coding and inter-coding. The resultant coded image is passed to the block transformation step, along with the prediction error. In the block Transformation step, the 8 × 8 block that is received from the prediction step, which consists of both coded frames and the prediction error, is subjected to the two-dimensional FDCT algorithm for conversion into a coefficient matrix. In the last step, quantization and entropy coding are applied to the DCT coefficients, which provides additional compression. Motion vector search, propagation of errors and bit-rate control are the three important, challenging issues in the H.261 standards.

4.4.1 Bitstream Structure

The bitstream structure of compression standard H.261 is illustrated in Fig. 16. The boundaries between pictures are designated with picture start code (PSC). The timestamp for each picture is indicated with a temporal reference (TR). The picture type (PType) is used to specify the type of frame as either P-frame or I-frame [37].

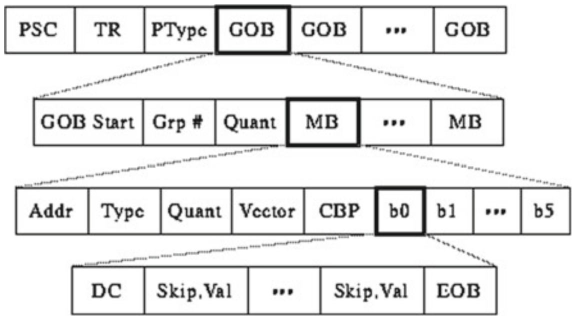


Fig. 16 Encoding sequence

The group of blocks (GOB) component of the bitstream indicates the separation of the picture into 11 $\times$ 3 macroblock regions. The group number (Grp #) indicates whole groups that should be skipped and the group quantization value (GQuant) indicates the single quantization value for the whole group.

5 Conclusions

The Internet of Things technology has generated a radical increase in the production of multimedia big data and is likely to transform enterprises, industries, and homes globally. The resultant tremendous amount of multimedia data has caused a scarcity of storage and a shortage in bandwidth for transmission, processing and receiving. It is essential and inevitable to develop methods for influencing the usage trajectory of the Internet of Things. The combined effort of researchers and scientists is necessary for the development of effective compression techniques for handling the tremendous amount of multimedia big data that is available in IoT.

This chapter, which is devoted to the compression of IoT multimedia big data, contains technical information that was obtained via the combined effort of many writers, academic investigators, technical investigators, scientific societies, software organizations, and professionals from various geographical areas. Exploring these technical details has motivated the recent research by the students and scientists who are working in this area. This effort also provides the opportunity to exchange knowledge between scientists and various technical societies.

In addition, this effort provides proper guidance for scholarly societies who are supporting the IoT industry and the processing and compression of multimedia big data from IoT applications. Almost all experts unanimously agree that compression of multimedia big data is an important and unavoidable requirement for the assurance of an excellent QoS in all multimedia applications. Almost all compression techniques that are used on multimedia big data currently can be applied to multimedia big data from a variety of IoT-enabled sources.

We have confidence that research on multimedia compression will yield optimal performance for all encoders that are applied to multimedia big data. Actions are already being taken by scientific societies on improving compression standards. IoT systems must accommodate the specifications of multimedia big data so that they can be incorporated into an international standard. This will provide assurance regarding the durability of multimedia big data. In addition, preprocessing and postprocessing techniques improve the quality of multimedia big data.

The compression of multimedia big data is now unanimously acknowledged as compulsory for enhancing the availability of bandwidth and storage requirements. Therefore, all compression standards and techniques can be applied to mono-dimensional, two-dimensional and multidimensional multimedia big data. It is also important to consider the security of multimedia big data during compression because IoT information is freely transmitted over the internet. We are confident that our presentation in this chapter properly guides scientists, scholars, and IoT professionals in conducting additional research in the area of multimedia big data compression.

References

1. L. Atzori, A. Iera, G. Morabito, The Internet of Things: a survey. Comput. Netw. **54**, 2787–2805 (2010)
2. D. Evans, The Internet of Things: how the next evolution of the internet is changing everything (2011)
3. S. Kumari, S. Tanwar, N. Tyagi, M. Kumar, K.K.R. Maasberg, Choo multimedia big data computing and Internet of Things applications: a taxonomy and process model. J. Netw. Comput. Appl. **124**, 169–195 (2018)
4. F.H. George, V. Jeffrey, W.M. Keith, The Internet of Things: a reality check. IEEE Comput. Soc. **14**(3), 56–59 (2012)
5. The Statistics Portal. Internet of Things (IoT) connected devices installed base worldwide from 2015 to 2025 (in billions) (2017), https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/
6. IDC. Internet of Things Market Statistics (2016), http://www.ironpaper.com/webintel/articles/internet-of-things-market-statistics/
7. A. Kumari, S. Tanwar, S. Tyagi, N. Kumar, Fog computing for healthcare 4.0 environment: opportunities and challenges. Comput. Electr. Eng. **72**, 1–13 (2018)
8. L. Jie et al., A survey on Internet of Things: architecture, enabling technologies, security and privacy, and applications. IEEE Internet of Things J. **99**, 1 (2017)
9. S. Tanwar, P. Patel, K. Patel, S. Tyagi, N. Kumar, M.S. Obaidat, An advanced internet of thing based security alert system for smart home, in *International Conference on Computer, Information and Telecommunication Systems (IEEE CITS-2017)*, Dalian University, Dalian, China, 21–23 July 2017, pp. 25–29 (2017)
10. D.X. Li, H. Wu, L. Shancang, Internet of Things in industries: a survey. IEEE Trans. Ind. Inform. **10**, 2233–2243 (2014)
11. L. Yu, Y. Lu, X. Zhu, Smart hospital based on Internet of Things. J. Netw. **7**, 1654–1661 (2012)
12. P. Mark, G. Eric, C. Ryan, F. Samantha, W. Leon, C. Hsinchun, Uninvited connections: a study of vulnerable devices on the Internet of Things (IoT), in *Proceedings of the 2014 IEEE Joint Intelligence and Security Informatics Conference (JISIC)*, The Hague, The Netherlands, pp. 232–235 (2014)
13. W. Zhu, P. Cui, Z. Wang, G. Hua, Multimedia big data computing. IEEE Multimedia **22**(3), 96–106 (2015)
14. S.-C. Chen, R. Jain, Y. Tian, H. Wang, Special issue on multimedia: the biggest big data. IEEE Trans. Multimed. **1** and **17**, 1401–1403 (2015)
15. M.K. Jeong, J.-C. Lu, X. Huo, B. Vidakovic, D. Chen, Wavelet based data reduction techniques for process fault detection. Technometrics **48**(1), 26–40 (2006)
16. M. Chen, S. Mao, Y. Liu, Big data: a survey. Springer Mob. Netw. Appl. J. (MONET) **19**(2), 171–209 (2014)
17. M. Chen, S. Mao, Y. Zhang, V.C. Leung, *Big data: related technologies, challenges and future prospects* (Springer, New York, NY, 2014)
18. K. Wang, J. Mi, C. Xu, L. Shu, D.-J. Deng, Real-time big data analytics for multimedia transmission and storage, in *Proceedings of IEEE/CIC International Conference on Communications in China (ICCC)*, Chengdu, China, pp. 1–6 (2016)
19. S.A. Hyder, R. Sukanesh, *An Efficient Algorithm for Denoising MR and CT Images Using Digital Curvelet Transform*. Springer Advances in Experimental Medicine and Biology—Software Tools and Algorithms for Biological Systems, vol. 696, Part 6, pp. 471–480 (2011)
20. A. Yassine, A.A.N. Shirehjini, S. Shirmohammadi, Bandwidth on demand for multimedia big data transfer across geo-distributed cloud data centers. IEEE Transa. Cloud Comput. **PP(**99), 1 (2016)
21. D. Ren, L. Zhuo, H. Long, P. Qu, J. Zhang, MPEG-2 video copy detection method based on sparse representation of spatial and temporal features, in *Proceedings of IEEE Second International Conference on Multimedia Big Data (BigMM)*, Taipei, Taiwan, pp. 233–236 (2016)

22. A. Paul, A. Ahmad, M.M. Rathore, S. Jabbar, Smartbuddy: defining human behaviors using big data analytics in social Internet of Things. IEEE Wirel. Commun. **23**(5), 68–74 (2016)
23. JPEG 2000 image coding system—Part 8: JPSEC Final Committee Draft—Version 1.0, ISO/IEC JTC1/SC29/WG1N 3480 (2004)
24. J. Yosef, S.A. Hyder, An efficient artifact free denoising technique for MR images relying on total variation based thresholding in wavelet domain. ICGST J. Graph. Vis. Image Process. **18**(1) (2018)
25. JPEG 2000 image coding system—Part 1: Core Coding System, ISO/IEC JTC 1/SC 29/WG 1 15444–1
26. T. Ebrahimi, C. Christopoulos, D.T. Lee, Special issue on JPEG-2000. Image Commun. J. **17**(1) (2002)
27. T. Ebrahimi, D.D. Giusto, Special section on JPEG2000 digital imaging. IEEE Trans. Consum. Electr. **49**(4), 771–888 (2003)
28. JPEG 2000 image coding system—Part 9: Interactivity tools, APIs and protocols, ITU-T Recommendation T.808, ISO/IEC 15444–9, July 2004
29. S. Pouyanfar, Y. Yimin, C. Shu-Ching, S. Mei-Ling, S.S. Iyengar, Multimedia big data analytics: a survey. ACM Comput. Surv. **51**(1), Article 10, 34 (2018), https://doi.org/10.1145/3150226
30. C.A. Bhatt, M.S. Kankanhalli, Multimedia data mining: state of the art and challenges. Multimed. Tools Appl. **51**(1), 35–76 (2011)
31. C. Min, A hierarchical security model for multimedia big data. Int. J. Multimed. Data Eng. Manage. **5**(1), 1–13 (2014)
32. S. Kaneriya, S. Tanwar, S. Buddhadev, J.P. Verma, S. Tyagi, N. Kumar, S. Misra, A range-based approach for long-term forecast of weather using probabilistic markov model, in *IEEE International Conference on Communication (IEEE ICC-2018)*, Kansas City, MO, USA, 20–24 May 2018, pp. 1–6 (2018)
33. C. Shu-Ching, Multimedia databases and data management: a survey. Int. J. Multimed. Data Eng. Manage. **1**(1), 1–11 (2010)
34. C. Ming, S. James, J. Zhanming, Connection discovery using big data of user-shared images in social media. IEEE Trans. Multimed. **17**(9), 1417–1428 (2015)
35. O.H. Ben, W. Matthew, JPEG compression, in *Student Projects in Linear Algebra*, ed. by D. Arnold (2005). Accessed 2009
36. P. Penfield, Chapter 3: compression, in *Notes* (MIT, 2004). Accessed 6 Sept 2009
37. P. Charles, Digital video and HDTV: algorithms and interfaces, in *The JPEG Still Picture Compression Standard*, ed. by G.K. Wallace. Communications of the ACM, 1 April 1991, pp. 30–44 (1991)
38. J. Yosef, Principal component analysis based multimodal medical image fusion of MRI and CT in wavelet domain, in *Transactions on Mass-Data Analysis of Images and Signals*, Vol. 9, no. 1, pp. 17–30, September (2018). ISSN: 1868–6451
39. T.P. Mahsa, D. Colin, A. Maryam, N. Panos: HEVC: the new gold standard for video compression. IEEE Consum. Electr. Mag. pp 36–46 (2012)
40. O.-R. Jens, J.S. Gary, S. Heiko, K.N. Thiow, W. Thomas, Comparison of coding efficiency of video coding standards—including high efficiency video coding (HEVC). IEEE Trans. Circuits Syst. Video Technol. **22**(12), 1669–1684 (2012)
41. K.R. Rao et al., *Video Coding Standards, Signals and Communication Technology* (Springer Science Business Media, Dordrecht, 2014), https://doi.org/10.1007/978-94-007-6742-3_2
42. W. Raymond, F. Borko, *Real-Time Video Compression—Techniques and Algorithms*, vol. 376, 1st edn. (Springer Science Business Media, Dordrecht, 1997)